Computational Experiments on Metaheuristic Techniques for Combinatorial Problems

Thesis submitted in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

By

Remya Kommadath

Roll No: 146107042

Department of Chemical Engineering

Indian Institute of Technology Guwahati

Guwahati-781039, India

Computational Experiments on Metaheuristic Techniques for Combinatorial Problems

Remya Kommadath

I

Dedicated to

Anitha Pattayil

Jinachandran Kommadath

Bhavya Kommadath

&

Debasis Maharana

Department of Chemical Engineering
Indian Institute of Technology Guwahati
Guwahati, Assam − 781039, India

Declaration

I hereby declare that the thesis is a presentation of my original research work done under the guidance of my thesis supervisor, Dr. Prakash Kotecha, Associate professor, Department of Chemical Engineering, Indian Institute of Technology Guwahati. Wherever contributions of others are involved, every effort is made to indicate this clearly, with due reference to the literature. The work documented in this thesis has not been submitted to any other University or Institute for the award of any degree or diploma.

Remya Kommadath
Roll No: 146107042
Department of Chemical Engineering
Indian Institute of Technology Guwahati
Guwahati 781039, India
Date: 04-11-2024

Department of Chemical Engineering
Indian Institute of Technology Guwahati
Guwahati, Assam − 781039, India

CERTIFICATE

It is certified that the work reported in the thesis entitled "**Computational Experiments on Metaheuristic Techniques for Combinatorial Problems**", by **Remya Kommadath** for the award of the degree of Doctor of Philosophy has been carried out under my supervision. To the best of my knowledge, the work documented in this thesis has not been submitted to any other University or Institute for the award of any degree or diploma.

Dr. Prakash Kotecha
Associate Professor
Department of Chemical Engineering
Indian Institute of Technology Guwahati
Guwahati 781039, India
Date:

Acknowledgment

I want to express gratitude to my research supervisor, **Dr. Prakash Kotecha**, for his constant support throughout the course of this work. I would also like to thank all my doctoral committee members, **Prof. Pallab Ghosh** (Department of Chemical Engineering), **Prof. Prabir Kumar Saha** (Department of Chemical Engineering), and **Prof. Deepak Sharma** (Department of Mechanical Engineering) for their valuable feedback on my research work. I am also thankful to all the faculty members and staff of the Chemical Engineering Department for their help at various stages.

I want to thank my friends at IIT Guwahati for making my stay enjoyable. I am deeply grateful to my friend **Dr. Varun Punnathanam** for his constant encouragement, which played a pivotal role in helping me uncover hidden potential within myself. I express my heartfelt gratitude to **Dr. R. Anandalakshmi**, **Dr. Deepti Nair**, and **Ms. Soumi Sarkar** for supporting me during my moments of hardship.

I wish to acknowledge that the journey of my Ph.D. work would not have been possible without the unwavering support of **my family**. Last but not least, I extend my gratitude and acknowledgment to my best friend turned husband, **Dr. Debasis Maharana**, for his unwavering support and optimistic perspective, which have been instrumental in helping me navigate through various challenges during this research journey.

Remya Kommadath

IIT Guwahati

Abstract

Metaheuristic techniques have emerged as a promising approach for addressing complex optimization models across diverse disciplines due to their capability to traverse vast and discrete search spaces, manage non-linearity, and accommodate multiple objectives. While these techniques do not mandate specific modeling prerequisites, it is advantageous to model problems efficiently to enhance their performance. Additionally, the precise implementation of these techniques is essential for achieving effective performance, even for identical problems where their superiority is asserted. This thesis provides strategies suitable to metaheuristic techniques for solving combinatorial problems and variants that can harness the benefits of parallel computing. This thesis highlights the importance of efficient solution structure by considering three combinatorial optimization problems.

Firstly, the production planning of the petrochemical industry is considered, where limitations of strategies employed in the literature to solve three models are critically analyzed, and concise solution structures along with efficient constraint handling methods are proposed. The efficiency of the proposed strategies is demonstrated by solving multiple cases of the production planning models using different metaheuristic techniques. The proposed strategies could determine similar or better solutions as compared to the literature while reducing the problem complexity. Further, a multi-objective multi-unit production planning model is solved by utilizing the proposed strategy to determine the trade-off between profit and investment cost. Multiple multi-objective metaheuristic techniques were used to determine the Parato solutions for the four cases of the production planning problem.

Secondly, the scheduling of jobs on dissimilar parallel machines is studied, and an optimization framework involving metaheuristic techniques and a novel no-wait time heuristic mechanism is proposed to solve the model efficiently. The performance of the proposed framework is

analyzed using multiple metaheuristic techniques by solving twelve instances of a scheduling problem. The rescheduling of jobs based on the proposed heuristic mechanism could minimize constraint violations, resulting in the identification of feasible solutions in the earlier stages of metaheuristic techniques. A multi-objective scheduling model using the proposed framework is proposed considering the minimization of total processing cost and makespan as two objectives. This inclusion of the heuristic mechanism was observed to be beneficial in determining better converged non-dominated solutions.

This thesis considers the optimization of a vapor compressor absorber cascaded refrigeration system as the third combinatorial problem. An efficient solution structure is proposed to determine the best absorbent solution and refrigerant pair along with the required operating parameters in a single optimization attempt. The proposed optimization strategy effectively determines the most suitable refrigerant, absorbent solution-refrigerant pair, and optimal operating conditions from a set of eleven refrigerants and three absorbent solution-refrigerant combinations. The modifications proposed in this thesis for solving combinatorial problems are generic and can be incorporated with the metaheuristic techniques without changing their innate structure. Furthermore, this study helped analyze the sensitivity of metaheuristic techniques toward the choice of decision variables and the strategies employed to handle constraints.

In the direction of exploring the possibilities of parallel computing with metaheuristic techniques, a critical analysis of teaching learning-based optimization (TLBO) is analyzed. Three variants of TLBO are proposed without incorporating the removal of duplicate solutions. These variants primarily differ in the stage at which new solutions are generated and evaluated in an iteration. Additionally, two variants of multi-objective improved teaching learning-based optimization (MO-ITLBO) are proposed, which provide insights into the ambiguities in its exact implementation.

Contents

List of tables

List of figures

Chapter 1

Introduction

Optimization aims to identify the best feasible solution among a set of choices and plays a crucial role across a wide range of fields and disciplines. A wide range of methods are used for solving optimization problems, which can be broadly classified into mathematical programming and metaheuristic techniques. Even though the metaheuristic techniques do not guarantee global optima, they are widely used to determine approximate solutions in a reasonable time for problems with complex search space. Most metaheuristic techniques imitate various optimization procedures followed in nature to converge on an optimal solution. Metaheuristic techniques possess several advantages, such as (i) the ability to handle nonlinearities, (ii) accommodating conflicting objectives, (iii) providing value-added solutions, (iv) solving black-box optimization problems, (v) not requiring information about gradients, (vi) can be tuned to accommodate problem specific operators, (viii) does not require the problem to be postulated in the conventional equality and inequality form, (ix) number of decision variables and constraints does not increase exponentially with the problem size, unlike formulations using Mixed-Integer Linear Programming (MILP) or Mixed-Integer Nonlinear Programming (MINLP) model, and (x) can be easily integrated with parallel and GPU computing, etc.

The metaheuristic techniques have attained wide acceptance from various fields (Li et al., 2024; Rezk et al., 2024), such as agri-food supply chain networks (Prajapati et al., 2022), refrigeration (Nagraj et al., 2022), solar collector design (Maharana et al., 2022), facility location (Latifi et al., 2017), cancer detection (Abdel-Basset et al., 2023), image processing (Ma et al., 2023), and biorefineries (Maharana et al., 2023). These techniques are also applied to address high-dimensional combinatorial problems (Abdel-Basset et al., 2022). It is observed that metaheuristic techniques are increasingly replacing the classical optimization methods in

solving real-life problems (Weinand et al., 2022), as these techniques are independent of the nature of the problem (Pinel et al., 2018). It usually converges to an approximate solution faster than its classical counterpart for high-dimensional combinatorial problems (He et al., 2021).

The combinatorial problems are significantly complex and often modeled using mathematical programming techniques. The use of such techniques necessitates the use of binary variables for realizing optimal decisions at various stages. The use of binary variables is observed for selecting transportation modes (Xie & Cao, 2020), representing the product choice, the change-over of one product to another, and the choice of machine to process in planning and scheduling (Liang et al., 2019). These variables also find their applicability in determining the location of various facilities among the available possibilities in the optimal design of a cooling grid (Al-Noaimi et al., 2019), in indicator constraint modeling for product tolerance design in a multiproduct assembly (Tsutsumi et al., 2020). It is also observed that using exact methods to solve combinatorial optimization problems formulated using mathematical programming techniques leads to an exponential increase in the problem dimension and requires significant computational resources for the solution (Rothlauf, 2011).

In combinatorial scheduling problems, using classical optimization techniques may guarantee an optimal schedule for small to medium-sized scheduling problems (Rossi et al., 2016). However, the increase in the problem complexity increases the consumption of resources (time and memory), leading to situations where the classical methods either converge to a sub-optimal solution or remain unsolved due to computational resource constraints (Jain & Grossmann, 2001). Such a scenario made the researchers strive for sub-optimal solutions by utilizing a reasonable amount of resources (Zhao et al., 2020), leading to the application of metaheuristic techniques to solve high-dimensional combinatorial problems. Various metaheuristic techniques are utilized to solve a transportation problem with fewer decision variables than its MILP model (Herrán et al., 2012). A three-stage variable neighborhood

search collaborated with the water wave optimization algorithm, is employed to solve a distributed flow shop scheduling problem (Zhao et al., 2021). Wang et al. (2022) used hybrid adaptive differential evolution to solve the fuzzy processing and completion time job shop scheduling problem. A review of green shop scheduling problems by Li and Wang (2022) reported metaheuristics to be the most popular choice for obtaining solutions.

Although metaheuristic techniques are good black-box optimization solvers, their effectiveness depends on the efficient modeling of the problem. These techniques do not mandate a canonical model of combinatorial problems, and the suitable transformation of constraints can help to avoid binary variables (Zhao et al., 2022). Though no modeling requisites are present for these techniques, it is mandatory to incorporate every feature of the problem while modeling it to obtain better performance (Daolio et al., 2017). Alfares and Al-Amer (2002) proposed a MILP model that considers binary variables to select processes and the production capacity of processes in the production planning of the petrochemical industry. Chauhan and Kotecha (2020) considered the same production planning model and significantly reduced the number of decision variables by proposing a model suitable for metaheuristic techniques. It should be noted that metaheuristic techniques lack inherent constraint-handling strategies and rely on additional methodologies (Coello Coello & A., 2002) to determine feasible solutions. The metaheuristic algorithms are devised to solve continuous real parameter optimization problems, while the combinatorial problems have discrete search space. Hence, the researchers have to encode specific strategies to the metaheuristic techniques for handling the discrete variables (Bhosale & Pawar, 2020).

Despite their advantages, metaheuristic techniques need to be judiciously applied to solve combinatorial optimization problems. These techniques might exhibit poor performance if they solve the model formulated in the conventional MILP or MINLP way that employs many artificial variables. In many instances, their success is critically dependent on the appropriate

choice of decision variables (Chauhan & Kotecha, 2018; Chauhan et al., 2018) and their bounds, modification, or incorporation of specific operators (Martínez et al., 2014) that exploit the structure of the problem. As metaheuristic techniques do not have any inherent constraint-handling capability, the modifications in the metaheuristic techniques are justified to identify a feasible solution. In this context, a relatively common practice is using a heuristic mechanism, which modifies the potential solution provided by the metaheuristic techniques to obtain a feasible or nearby feasible solution. Hossain and co-authors (Imran Hossain et al., 2019) have incorporated a repair mechanism to avoid invalid solutions to the university course scheduling problem. In an unrelated parallel machine scheduling (Yepes-Borrero et al., 2020), a repair mechanism is performed on the infeasible solution to satisfy the resource constraints.

In the past few decades, plenty of metaheuristic techniques have been proposed in the literature to solve a spectrum of optimization problems across various disciplines. In literature, researchers usually analyze the performance of these newly proposed algorithms on a set of classical benchmark functions (Faramarzi et al., 2020). Such a practice rarely ensures the better performance of these techniques while solving real-life problems in their innate form. In that instance, a variant of the technique with several modifications to incorporate the problem-specific characteristics is proposed to enhance its performance (Balakrishnan et al., 2022). A variant is also proposed to gain a better performance without getting trapped in local optima and to adapt the technique to different problem domains (Ahmed et al., 2023; Xu et al., 2020).

For problems with a computationally expensive objective function, evaluating the objective function for once might be excessively time-consuming. As metaheuristic techniques are population-based techniques that evaluate the objective function multiple times, their application for solving such problems becomes time expensive. In such a scenario, utilizing parallel computing helps to overcome this drawback. Popular metaheuristic techniques, such as GA (Arora et al., 2010), allow the evaluation of objective function parallelly, as in these

techniques, the generation of a new potential solution in an iteration is independent of updating any other solution. However, for techniques such as teaching-learning-based optimization (TLBO), artificial bee colony (ABC), etc., the population has to be updated after evaluating each solution in an iteration. Hence, such techniques could not harness the benefits of parallel computing apart from evaluating the initial population. Variants of the techniques necessitating the sequential evaluation of objective functions are available in the literature, where the modifications performed in the innate technique help to utilize the benefits of parallel computing (Alrezaamiri et al., 2020; Niu et al., 2018).

In view of the above discussions, this thesis traverses through the possibilities of metaheuristic techniques in solving combinatorial problems. The combinations of a better decision structure, efficient constraint handling, hybridization with heuristic mechanism, and other modifications are incorporated to achieve improved performance of the metaheuristic techniques. In addition, multi-objective optimization by utilizing efficient strategies was explored in this thesis. This work also explores the aspects of integrating parallel computing with metaheuristic techniques in solving computationally expensive problems. Additionally, the importance of providing the correct implementation of any metaheuristic technique to ensure their efficient performance is highlighted in this work.

Chapter 2

Literature review

This thesis focuses on demonstrating the benefits of efficient solution frameworks for metaheuristic techniques in solving combinatorial optimization problems as well as complex nonlinear optimization models. This chapter provides a review of metaheuristic techniques and their application to different types of optimization problems. Section 2.1 provides a brief history of metaheuristic techniques and their usage by various researchers. The application of metaheuristic techniques on two types of combinatorial problems, namely the production planning in the petrochemical industry and the job scheduling problems, are reviewed in Section 2.2 and Section 2.3, respectively. Section 2.4 presents a review of metaheuristic techniques on the various approaches for solving a combinatorial refrigeration model. Section 2.5 describes the significance of variants of metaheuristic techniques and the requirement for publicizing the exact implementation of the techniques to retain their effective performance. Section 2.6 discusses the research gaps, and finally, the objectives of the thesis are presented in Section 2.7.

2.1 Metaheuristic techniques

Metaheuristic techniques have emerged as promising methods for optimizing various engineering fields. Generally, these are population-based iterative techniques that imitate the phenomena occurring in nature to solve complex optimization problems. The metaheuristic techniques mostly begin with an initial population involving random solutions generated within the variable bounds to start the search for optima. The exploration and exploitation of the search space are performed in every iteration using the variation operators of each technique, and the population is updated with the specific strategy of the technique based on the objective function value of the solutions. One of the major advantages of metaheuristic techniques is their easiness

of applicability to solve any kind of problem without devising major structural changes to the framework of the technique (Fausto et al., 2020).

The history of these techniques can be traced back to the mid-20th century. One of the earliest and most well-known techniques is the genetic algorithm (GA) proposed by John Holland in the 1960s, which follows natural genetics and evolution in search of the optima (Holland, 1992). Unlike other metaheuristic techniques, simulated annealing (SA) proposed by Kirkpatrick et al. (1983) works with a single solution and is inspired by the annealing process in metallurgy. SA accepts a worse solution with a small probability to avoid getting trapped in a local optimum. Tabu search keeps a tabu list for tracking the previously visited solutions to avoid revisiting them (Glover & Laguna, 1997). Particle swarm optimization is one of the most popular swarm intelligence techniques, which mimics bird flocking or fish schooling in search of food (Kennedy & Eberhart, 1995). Differential evolution modifies the vectors in the population using mutation and crossover operators (Storn & Price, 1997) and is also a competitive candidate among various metaheuristic techniques. These techniques remain popular in the research community, and their variants and hybrid approaches are widely used to solve various optimization problems in engineering and other domains. The continuous development and application of these metaheuristic techniques demonstrate their effectiveness and versatility in addressing complex optimization challenges.

In addition to these established metaheuristic techniques, there has been a proliferation of new techniques in the past two decades, which also emulate several natural phenomena. These newly proposed techniques claimed better performance than the popular metaheuristic techniques on benchmark functions and some real-life optimization problems. However, it is necessary to consider the "no free lunch theorem of optimization." According to the "no free lunch theorem," no single optimization algorithm can be regarded superior to others across all

possible problem sets unless they are evaluated under the same criteria and the algorithm of interest has provided better performance than others. While new metaheuristic techniques may demonstrate improved performance on certain problem sets, it does not imply that they are universally superior. Therefore, rigorous evaluations and comparisons are required by considering multiple problem instances and performance metrics to understand the strengths and weaknesses of various metaheuristic techniques comprehensively.

Almost every engineering field involves nonlinear optimization problems, and the flexibility and robustness of metaheuristic techniques to handle such problems led to their wide acceptance as a solver. Usually, the search space of nonlinear problems is multimodal, making it challenging to determine their global optima. The variation operators and the selection strategy employed in metaheuristic techniques help to search the solution space effectively and lead to the determination of a better optimum. Some applications of metaheuristic techniques for solving nonlinear models in various fields include optimizing multi-reservoir hydroelectric energy system using monarch butterfly algorithm (Ehteram et al., 2017), designing heat exchangers using teaching learning-based optimization (Patel & Savsani, 2014a), genetic algorithm for deploying sensors in heterogeneous sensor network (Karatas, 2018), hybrid salp swarm Coot algorithm for optimal portfolio selection (Pahade & Jha, 2022), etc.

Combinatorial problems are another important type of problem that metaheuristic techniques are adept at handling. The resulting optimization models of such combinatorial problems comprise multiple discrete decisions (Tumbalam Gooty et al., 2019). Such optimization models are NP-hard that are difficult to solve and require substantial computational effort to get a feasible solution (Roy et al., 2018). The classical methods are often inefficient in solving such problems with larger dimensions (Silveira et al., 2021). Hence, most researchers utilize metaheuristic techniques or certain local search operators (Sallam et al., 2020) to reach an

acceptable solution within reasonable resources. Several such applications that have utilized metaheuristic techniques include assembly line design (Buyukozkan et al., 2016; Fathi et al., 2020; Li et al., 2018), load balancing (Homaee et al., 2019), facility location (Ferreira & de Queiroz, 2018; Latifi et al., 2017), scheduling (Abu-Marrul et al., 2021; Tozzo et al., 2020), order batching (Henn & Schmid, 2013; Žulj et al., 2018), routing (Alinaghian et al., 2021), etc.

Although metaheuristic techniques are quick in solving optimization problems as compared to classical techniques, the requirement of repeated evaluation of the objective function becomes computationally expensive in certain cases. However, parallel computing offers a potential solution to the issue where the evaluation of the objective function is computationally intensive. Many metaheuristic algorithms, including genetic algorithms (GA), particle swarm optimization (PSO), and others, are inherently parallelizable, which means the objective function of the whole population in every iteration is evaluated parallelly to reduce the computational burden effectively. The proposal of new variants, whether for improving their performance in identifying better optima or harnessing the benefits of parallel computing, is necessary to implement them accurately. Any mistakes in qualitative or quantitative aspects of an algorithm can deteriorate its performance (Črepinšek et al., 2012).

2.2 Metaheuristic techniques for production planning of the petrochemical industry

Petrochemical manufacturing plants are massive and complex structures with several processing steps, beginning with the procurement of feedstock to produce and distribute useful products. These superstructures require optimal decisions at various planning, scheduling, and execution levels (Kwon et al., 2022). Ehrenstein et al. (2019) proposed a method to optimize the supply chain by considering extreme weather events to analyze economic and risk performance. Ghamari and Sahebi (2017) used a stochastic lot-sizing inventory model to manage petrochemical products efficiently.

The role of production planning in manufacturing industries is inevitable as it helps to earn more profit by improving the infrastructure and production arrangement based on dynamic customer requirements. The production plan specifies how to execute the production processes by efficiently utilizing resources such as the workforce, raw materials, and investment budget. The capital intensiveness and usage of sophisticated technologies in petrochemical industries necessitate optimal production planning. Alfares and Al-Amer (2002) proposed an optimization model for petrochemical industries in the development stages that provides decisions related to the choice of process and quantity of products to be manufactured to gain maximum profit. Integrating and coordinating a multisite refinery and petrochemical network with an efficient production strategy for satisfying the demand for multiple products was studied by Al-Qahtani and Elkamel (2009). A decision framework for resource allocation and production planning in the petrochemical industry is studied by considering the limited inventory capacity (Kwon et al., 2020). Siwi et al. developed a model to obtain strategic planning for the downstream integrated petroleum and petrochemical refinery, aiming for improved economic performance and environmental impact (Siwi et al., 2018). The long-term assessment of capacity expansion to minimize total cost and risk in the petrochemical industry is studied using a multi-period multi-objective model (Boaventura et al., 2022).

The development of models for real-world problems is a complex task and is often observed to be solved using different approaches to improve the desired objectives (Guzman et al., 2022). The production planning model proposed by Alfares and Al-Amer (2002) provides decisions related to the selection of an optimum set of products and the associated technologies for guiding the expansion of the Saudi Arabian petrochemical industry for its development stage. The model has incorporated realistic production and investment costs as the function of the production capacity. The production and investment costs associated with three different production capacities (low level (l), medium level (m), and high level (h)) of the

processes available to produce petrochemical products are known. A process is allowed to implement only one production unit with a capacity between low-level and high-level production capacities. The problem is associated with constraints on the bounds of decision variables and resources such as investment budget and raw materials. Additionally, the addition of unique process constraints restricts the use of more than one process for manufacturing a product. This model has been extended in the literature to explore the possibilities of obtaining a production plan with maximum economic benefit, subject to all the associated constraints.

There have been multiple attempts to enhance the economic benefits by adding features to the model and using efficient solution strategies. The first modification to the model is observed by installing one process unit between the l-m and m-h production capacity of a process (Kadambur & Kotecha, 2015), which is termed the multi-level production planning strategy. The multi-level production planning problem is solved using elitist teaching learning-based optimization and is observed to obtain a better profit. The multi-level production planning model is further extended as a multi-unit production planning problem modeled using integer and continuous decision variables (MUIPP) by incorporating multiple units of the processes at its *l, m,* and *h* production capacities and one unit each at *l-m* and *m-h* production capacities to obtain more economic benefits (Chauhan et al., 2018). This model is also extended by considering the implementations of multiple units anywhere between *l-m* and *m-h* production capacities, and it is modeled using only continuous variables (Chauhan & Kotecha, 2018). The capacity expansion of processes using multiple levels of production capacity (Kadambur & Kotecha, 2015) and multiple units of profitable processes (Chauhan & Kotecha, 2018) were observed to improve the profit significantly.

2.3 Metaheuristic techniques for solving scheduling problem

Scheduling is an integral part of process industries with its applications in a multitude of industrial sectors, such as batch scheduling in brewing (Sáenz-Alanís et al., 2016), mold and die making (Roshanaei et al., 2013), manufacturing in the pharmaceutical firm (Costa, 2015), operation scheduling in integrated iron and steel enterprises (Hu & He, 2022), etc. Scheduling problems require assigning and sequencing multiple jobs or orders on the available machines. Each job has some parametric conditions such as release date, due date, resource requirement, processing duration, operation sequence, etc. Scheduling problems also involve certain constraints, such as no cancellations allowed and prohibition in splitting a job processing on multiple machines. A schedule possesses information regarding machine to job allocation, sequencing of jobs on the machines, starting time of the jobs, etc. Optimization of scheduling problems involves the identification of an optimum schedule to achieve certain objectives. The various objectives considered in scheduling problems includes minimization of cost (de Miranda, 2019), makespan (Shih-Wei et al., 2021), inventory cost (Mishra & Shrivastava, 2018), total weighted tardiness of jobs (Jen-Ya, 2020), etc.

There have been multiple approaches in the literature to solve the scheduling problems such as disjunctive graph representation (Błażewicz et al., 2000), heuristic methods (Rossi et al., 2016), branch and bound method (D'Ariano et al., 2007), constraint programming (Lunardi Willian T. et al., 2020), metaheuristic techniques (Roshanaei et al., 2009), a combination of mixed-integer and constraint logic programming (Harjunkoski et al., 2000), etc. The use of classical optimization techniques may guarantee an optimal schedule for small to medium-sized scheduling problems (Rossi et al., 2016). As the problem complexity grows, the demand for resources, including time and memory, also escalates. Consequently, classical methods encounter limitations in converging to an optimal schedule due to these computational resource constraints (Jain & Grossmann, 2001). This implies that for more intricate problems, the

traditional approaches may only reach a sub-optimal solution. On the other hand, metaheuristic techniques can determine near-optimal solutions within a finite amount of time and memory.

The optimization of the scheduling problems using classical methods often introduces additional tightening constraints for the better delineation of the search space to identify a solution quickly (Bing Yan et al., 2020); however, these constraints are redundant regarding the complete problem description. In the asymmetric traveling salesman problem, the reformulation-linearization technique has been utilized to tighten the nonlinear sub-tour constraints (Sherali & Driscoll, 2002). A tightening approach based on constraint-and-vertex conversion and vertex projection has been incorporated into unit commitment problems in power systems (Yan et al., 2020). A constraint programming-based tightening is applied on the bounds of the decision variables that were included in solving a mixed-integer nonlinear programming model of refinery scheduling (Mouret et al., 2009). An upper bound tightening has been exerted to improve the computational results of the scheduling problem in a smelting plant (Ewaschuk et al., 2018) and energy distribution network (Yao et al., 2019). However, in the production scheduling at multistage facilities (Merchan et al., 2016), the inclusion of tightening constraints is observed to be ineffective for cost minimization. To the best of our knowledge, the literature lacks any study exploring the effect of tightening constraints on the metaheuristic techniques that require investigation in this direction.

Several researchers have used metaheuristic approaches for solving scheduling problems, such as solving flexible job shop scheduling problems using quantum behaved particle swarm optimization (Singh & Mahapatra, 2016) and multi-micro swarm leadership hierarchy optimization algorithm (Zhu & Zhou, 2020), university course scheduling using particle swarm optimization (PSO) (Imran Hossain et al., 2019), genetic algorithm (GA) (Wang, 2003) and harmony search (Al-Betar et al., 2012), scheduling in power systems using adaptive mutation

sine cosine algorithm (Feng et al., 2020), hybrid grey wolf optimizer (Makhadmeh et al., 2021), and so on. Appropriate modification to the inherent metaheuristic techniques has been found beneficial for solving scheduling problems. New procedures are employed for the initial population, selection, and reproduction strategies used in GA to minimize the makespan of flexible job shop scheduling problems (Pezzella et al., 2008). Tasgetiren et al. (2007) solved the flow shop sequencing problem using PSO and variable neighborhood search. In their work, a heuristic rule of the smallest position value is utilized to convert the continuous positions to the discrete sequences of the jobs. Nearchou (2004) introduced a hybridized Simulated Annealing (SA) with GA and local search for solving flow shop scheduling problems in which sequencing of jobs is done by providing higher priority to the job with longer processing time.

Metaheuristic techniques, in their innate form, solve only unconstrained optimization problems. The metaheuristic techniques do not have the inherent capability to handle constraints and often rely on constraint handling techniques to identify a feasible solution. However, these techniques alone may not be sufficient in effectively addressing all the constraints, necessitating modifications to the metaheuristic techniques themselves. A commonly adopted approach in this context is the utilization of a repair operator. The repair operator modifies the solution generated by the metaheuristic technique to achieve a feasible or nearly feasible solution.

For instance, Hossain and co-authors (Imran Hossain et al., 2019) incorporated a repair mechanism specifically designed to avoid invalid solutions for the university course scheduling problem. Their approach ensures that the generated schedules adhere to the restrictions, such as room availability and instructor availability, and avoid conflicts between courses. Similarly, in an unrelated study on parallel machine scheduling conducted by Yepes-Borrero et al. (2020), a repair mechanism was employed to handle infeasible solutions that fail to meet the resource

constraints. The repair operator modifies the solution iteratively until it satisfies the resource constraints, thereby producing a feasible schedule. These examples highlight the significance of incorporating repair mechanisms within metaheuristic techniques to handle constraints effectively. By integrating repair operators, researchers aim to enhance the ability of metaheuristic approaches to produce feasible or near-feasible solutions, thus improving their overall performance in constraint-based optimization problems.

The optimization of multiple conflicting objectives in real-world scheduling problems increases the difficulties in solving them due to the discontinuities in the Pareto fronts. The metaheuristic approach is considered a desirable alternative for solving multi-objective optimization problems. They are independent of the shape and continuity of the Patero front and provide multiple points on the Pareto front in a single simulation (Coello et al., 2004). It is common to incorporate other techniques, such as heuristic algorithms, fuzzy logic, local search operators, etc., along with multi-objective metaheuristic techniques to improve their efficiency in solving scheduling problems.

He et al. (2022) have used an adaptive multi-objective differential evolution algorithm using the lexicographic method to solve the multi-objective open shop scheduling problem, minimizing economic and environmental objectives. Azadeh et al. (2016) modified the multi-objective genetic algorithm (NSGA II) to optimize the objectives, minimization of makespan, and the workload of machines in an open shop scheduling problem. An enhanced multi-objective particle swarm optimization with a new position update technique and neighborhood local search is proposed to optimize the joint scheduling of flexible open shop problems with automated guided vehicles (Tan et al., 2021)

2.4 Application of metaheuristic techniques in refrigeration systems

Space cooling energy consumption was reported to have increased three times in the past three decades, and it was rated for approximately 16% of the final electricity consumption of the building sector in 2021 (about 2000 TWh) (IEA 2022). Energy efficiency standards are advised to be implemented to increase the energy performance of refrigeration and air conditioning units towards Net Zero Emissions by 2050 (IEA 2022). Minimizing energy consumption and reducing CO2 emissions in the refrigeration system is important. The performance of refrigeration and air conditioning units can be augmented by optimizing the working conditions, choosing appropriate systems, and system design (Mondal, Sahana, and De 2022). Refrigeration is a vital energy-consuming thermal process, which needs detailed thermodynamic modeling and analysis of various refrigerant properties (Yilmaz and Selbaş 2019). Thermoeconomic optimization techniques based on the "Theory of Exergetic Cost" (Lozano & Valero, 1993) are considered an efficient technique with a few advantages, such as its ability to analyze real-time systems without any simulation programs and advanced numerical techniques, irrespective of its complex nature.

Thermoeconomic / exergoeconomic optimization combines the idea of exergy and economics to explore the compromise between the total cost of the system and the cost of exergy input to the system. Thermoeconomic analysis along with environmental analysis can be implemented as a cost-efficient method to recognize the optimal thermal system via exergy-environment based cost minimization by incorporating exergy analysis, environmental analysis, and economic principles.

In order to get reasonable evaporation pressure at a lower evaporation temperature and a moderate condensation pressure at ambient temperature, two-stage Cascade Refrigeration Systems (CRS), which work with two or more dissimilar refrigerants, are preferable when

compared to a single-stage refrigeration system. In the two-stage vapor Compression–Absorption Cascade Refrigeration System (CACRS), a compression system at a low-temperature loop (LTL) and an absorption system at a high-temperature loop (HTL) are operated by power and waste heat to reduce electricity consumption. Many recent studies have been proposed in the field of structural optimization of CRS (Patel et al. 2019; Tan, Wang, and Liang 2015; Boyaghchi, Mahmoodnezhad, and Sabeti 2016; Nasruddin et al. 2016; Lizarte, Palacios-Lorenzo, and Marcos 2017; Patel, Desai, and Kachhwaha 2017; Giannetti et al. 2017; Bai et al. 2021; Sun et al. 2021; Zhang et al. 2023; Hao et al. 2022) for further improvement in their performance.

In the recent past, system optimization has been used in refrigeration units to select the optimal operating conditions, system design, plant efficiency, operating cost, etc. It is observed that classical optimization techniques such as Mixed-Integer Nonlinear Programming (Chen, Luo, and Yuan 2020), Lagrange Multipliers Method (X. Li et al. 2019), Sequential Quadratic Programming (Fu, He, and Zhang 2018), and Direct Search Methods (Jain, Sachdeva, and Kachhwaha 2015a) were applied to refrigeration systems. On the other hand, due to the stochastic nature and the capability of handling a large amount of data and redundancy of comprehensive information about the system, evolutionary optimization techniques such as Genetic Algorithm (Dai et al. 2022), Particle Swarm Optimization (Rahman and Zhang 2019), Differential Evolution (Kong et al. 2021), Chemical Reaction Optimization (Hadidi 2017), Artificial Cooperative Search (Turgut and Turgut 2019) and NSGA-II (Jain et al. 2019) are being successfully applied to optimize refrigeration problems. As the operating conditions, system design, and cost functions are different for different refrigeration systems, it is hard to choose any particular optimization algorithm/technique. Accordingly, the No Free Lunch theorem for optimization states that all types of optimization problems cannot be solved using a particular optimization technique (Wolpert and Macready 1997). Recently, data-driven

techniques (ANN) are also being applied to refrigeration system modeling (F. Ahmed and Chen 2023; R. Ahmed et al. 2021)

Based on the thermoeconomic optimization, Rezayan and Behbahaninia (2011) reported that the total annual cost of the NH_3/CO_2 based CRS decreased by about 9.34% compared to the base case for a constant cooling capacity of 40 kW. Patel et al. (2019) compared NH_3/CO_2 and C_3H_8/CO_2 based CRS using a heat transfer search algorithm to minimize the total annual cost and exergy destruction of the system. Nasruddin et al. (2016) analyzed a CRS with C_3H_8 in the HTL and C_2H_8+CO_2 in the LTL using the thermoeconomic optimization approach. The minimum total annual cost and exergy destruction of the system were reported as $51070.59 and 39876.04 W, respectively. Agnew and Ameli (2004) used refrigerant pairs of NH_3 and R508b (46% R23 + 54% R116) in place of R12 and R13 in a three-stage CRS and concluded that the suggested pairs showed better performance. Bhattacharyya et al. (2005) performed an optimization study on CRS using C_2H_8/CO_2. Dopazo et al. (2009) optimized CRS using CO_2 and NH_3. Lee, Liu, and Chen (2006) performed a thermoeconomic optimization study of CRS using CO_2 and NH_3 to optimize cascade condenser for various evaporator temperatures, condenser temperature, and the temperature difference in the cascade-condenser to minimize the exergy destruction and to maximize the COP. Aminyavari et al. (2014) applied the multi-objective genetic algorithm to optimize exergetic efficiency and the total cost rate of the NH_3/CO_2 based CRS using the concepts of exergy, economics, and environment.

Arshad et al. (2019) employed the genetic algorithm to improve the efficiency of LiBr-H_2O based double effect absorption refrigeration system in the series and parallel configuration by optimizing its various operating conditions. Eini et al. (2016) performed multi-objective optimization based on 3E analysis and inherently safe design in a CO_2/NH_3 based CRS. The results were compared with CO_2/C_3H_8 based system. Mosaffa et al. (2016) investigated

CO_2/NH_3 based CRS with flash intercoolers using exergoeconomic and environmental factors to maximize COP and minimize the total cost rate. Advanced exergy, exergoeconomic analyses, and multi-objective optimization were performed by Asgari, Noorpoor, and Boyaghchi (2017) for a butane-based internal auto-CRS. It was found that condenser inlet temperature variation (32 ºC to 40 ºC) increases the total avoidable exergy destruction by 88.18%, and compressor mass flow rate (from 0.08 kg/s to 0.3 kg/s) and evaporator inlet temperature (from -5 ºC to 0 ºC) increase total avoidable investment cost rates by 126.92% and 3.68%, respectively.

2.5 Significance of variants of metaheuristic techniques

The primary intention behind proposing variants of metaheuristic techniques includes enhancing their performance (Garg et al., 2023), adapting to problem domains (Zapotecas-Martínez et al., 2019), refining the balance between exploration and exploitation (Mostafa et al., 2023), improving the convergence speed (Sarkhel et al., 2018), and so on. Metaheuristic techniques proposed in the past few decades have been modified appropriately to overcome their disadvantages and cope with the problem requirements. Among popular metaheuristic techniques, teaching-learning-based optimization has been reviewed in this thesis.

Teaching-Leaning-Based Optimization (TLBO) is one such metaheuristic technique that was developed by Rao and co-researchers (Rao & Patel, 2012a; R. V. Rao, V. J. Savsani, & J. Balic, 2012; R. V. Rao, V. J. Savsani, & D. P. Vakharia, 2012) in 2011, and its inspiration is the knowledge transfer in a classroom environment. Like many other metaheuristic techniques, it is also a derivative-free optimization technique. These techniques require reasonably low modeling effort as they do not require the constraints to be postulated in terms of equalities and inequalities. It is used to solve several categories of problems and is observed to be competitive

with other state-of-the-art algorithms. In addition to its competitive performance and simplicity of the algorithm, it requires the user to specify only two parameters, viz., population size or class size and the termination criterion (such as the maximum number of iterations, the maximum number of function evaluations, etc.). This is contrary to other metaheuristic techniques, which require the selection of different methods (such as type of reproductive operator and variation operator in Genetic Algorithm) and tuning parameters (such as inertia factor and acceleration coefficients in Particle Swarm Optimization).

The simplicity of TLBO has also contributed to its widespread use for diverse applications such as parameter tuning (Niu et al., 2014; Rao et al., 2014), size and shape optimization (Dede & Ayvaz, 2015), design of heat exchangers (Patel & Savsani, 2014a), determination of optimal production plans (Kadambur & Kotecha, 2015), energy systems (Injeti et al., 2015), data classification (Shukla et al., 2019), chemical dynamic system optimization (Chen et al., 2018), short-term hydrothermal scheduling (Kumar Roy et al., 2013) and design of trusses (Camp & Farshchin, 2014; Dede, 2014) among other applications. Some of the variants of TLBO include elitist TLBO (Rao & Patel, 2012b), improved TLBO (R. Venkata Rao & Vivek Patel, 2013), modified TLBO (Ma et al., 2021), dynamic group strategy in TLBO (Zou, Wang, Hei, Chen, & Yang, 2014), bare-bones TLBO (Zou, Wang, Hei, Chen, Jiang, et al., 2014), quasi-oppositional TLBO (Roy & Bhui, 2013), dynamic opposite learning enhanced TLBO (Xu et al., 2020) and so on. It has also found its application in multi-objective optimization (Hajabdollahi et al., 2021; Rao & Waghmare, 2015; Zou et al., 2013b) and combinatorial problems (Ji et al., 2017; Li et al., 2015; Mishra & Shrivastava, 2018), etc. Some of the hybrid works include k-means clustering with TLBO (Mummareddy & Satapaty, 2015), combining with robust Tabu search (Dokeroglu, 2015), a multi-class cooperative teaching-learning based optimization with simulated annealing (Chen et al., 2016), Chaotic Particle Swarm

Optimization with TLBO (Azad-Farsani et al., 2014), etc. Any interested researcher can obtain additional details on various aspects of TLBO from the literature (Črepinšek et al., 2012; Rao, 2016.; Waghmare, 2013).

An immense amount of research is available in the literature related to the applicability of TLBO in various fields in its innate form, as variants, and as hybrid techniques. However, its relevance in solving computationally expensive problems is not explored considerably. The primary reason for this might be the requirement for the sequential evaluation of objective function in every iteration of TLBO. In the case of binary and real-coded GA, the operators, such as random number generation, initialization, selection, and mutation, are also parallelized along with the parallel evaluation of the objective function to provide time benefits (Arora et al., 2010). In the Artificial Bee Colony (ABC), the interdependency in updating the solution restricts its application to computationally expensive problems. This drawback is tackled by its parallel implementation of shared memory architecture to provide substantial time benefits (Harikrishna, 2009). The parallel implementation of multi-objective ABC has also employed the shared memory architecture with a master-slave model to obtain the final Pareto front (Alrezaamiri et al., 2020). Similar approaches have also been used to determine the Pareto solutions while solving the multi-objective cascade power system model using multi-objective PSO (Niu et al., 2018). Cruz et al. (2017) divided the population of TLBO into multiple subpopulations and evaluated each in parallel for solving automatic heliostat aiming. However, identifying the teacher, calculating the population mean, and selecting the partner solution require the information of each subpopulation to be shared with every other subpopulation.

The research community has also explored the potential of TLBO to solve multi-objective optimization problems (MOOP) by proposing its multi-objective variants. The initial works for extending TLBO to solve MOOP involved combining the multiple objectives into a single

objective and required multiple runs to generate the entire Pareto front (R. V. Rao & Vivek Patel, 2013; V. R. Rao & V. Patel, 2013). However, Zou et al. (2013a) extended TLBO to solve MOOPs and was primarily based on the non-dominated sorting used in NSGA-II (Deb et al., 2002). The proposed algorithm by Zou et al. was applied to six multi-objective unconstrained benchmark problems and two constrained real-world multi-objective optimization problems. This work compared the performance of the non-dominated sorting based TLBO to NSGA- II and a few other multi-objective metaheuristic techniques. However, this work did not consider any of the CEC 2009 (Zhang et al., 2008) problems, which include constrained optimization problems. Other works that used TLBO to solve MOOP include the decomposition based algorithm of Medina et al. (2014). A θ-multi-objective TLBO has been proposed by Niknam et al. Niknam et al. (2012) that utilizes an external repository for storing the Pareto solutions.

Various intricacies in the implementation of the TLBO and computational issues can be understood from the work of Črepinšek et al. (2012) and (Waghmare, 2013). Črepinšek et al. (2012) performed code reviews and experiments with the TLBO proposed by Rao et al. (2011) and Rao et al. (2012). Črepinšek et al. (2012) identified three mistakes in TLBO articles (Rao et al., 2012; Rao et al., 2011) regarding the determination of the number of function evaluations, the claim of the parameterless algorithm, and the inclusion of duplicate removal step. Due to a lack of consistency in the explanation provided in the article and implementation failed to validate the performance of TLBO mentioned in Rao et al. (2011) and Rao et al. (2012). A similar study related to artificial bee colony is also found in Mernik et al. (2015). These studies emphasize the importance of providing the exact implementation and proper explanation of the newly proposed metaheuristic techniques to ensure their better performance.

2.6 Research gap

In the context of solving combinatorial problems effectively using metaheuristic techniques, three different problems – production planning, scheduling, and optimization of a refrigeration system are considered. In these problems, the primary objective was to model the problem well-suited to the metaheuristic techniques by incorporating efficient solution representation and constraint handling techniques to achieve the best possible objective value. Towards this direction, the production planning models of the petrochemical industry present in the literature (Chauhan & Kotecha, 2018, 2020; Chauhan et al., 2018) are critically analyzed, and proposed a new strategy to solve them. Similarly, the efficient optimization framework involving a metaheuristic technique and a heuristic mechanism for scheduling multiple jobs on dissimilar parallel machines is proposed. Additionally, an efficient solution structure for the CACRS model that reduces computational effort is proposed.

Integrating certain problem insights into the optimization framework is observed to improve the performance of metaheuristic techniques in determining better solutions (Talbi, 2009). Moreover, the performance of metaheuristic techniques is sensitive to the representation of the decision variables as well as the strategies employed to handle various constraints. The sensitivity of some metaheuristic techniques with different integer constraint handling techniques is analyzed by Punnathanam et al. (2016). Additionally, process industries require separate constraint-handling techniques for the production domain of each process unit, as these units are allowed to produce between lower and upper capacity levels. The use of domain correction approaches in the supply chain model of process industries has been observed to be beneficial compared to the standard penalty approaches (Maharana et al., 2022).

In this direction, three optimization models, namely single-level production planning (SLPP) (Chauhan & Kotecha, 2020), multi-unit production planning modeled using integer and

continuous variables (MUIPP) (Chauhan et al., 2018), and multi-unit production planning modeled using only continuous variables (MUCPP) (Chauhan & Kotecha, 2018), which are used to determine an optimal production plan for developing a petrochemical plant are reviewed in this thesis. The models in the literature have attempted to identify an optimal production plan by maximizing the profit while satisfying constraints on resources, the domain of the decision variables, and the number of processes used for producing each product (unique process constraints). In the SLPP model, a product can be produced using a process by implementing a single unit within the bounds of the capacity. At the same time, in the multi-unit production planning models, a product can be manufactured using multiple units of a process, where the production capacity bounds of each production unit are different.

A careful investigation of the strategy provided to solve the SLPP model in Chauhan and Kotecha (2020) and the MUIPP model in Chauhan et al. (2018) has identified the scope of efficient modeling and, thereby, the possibility of improved performance of metaheuristic techniques. The production planning considered in the literature possesses one or more processes to produce each product. Hence, the total number of processes is always more than the number of products. The decision variables based on strategies in the literature represent the production quantity of each process unit. In such a scenario, the number of decision variables and domain constraints increases with the number of processes. The unique process constraints in the model restrict the usage of more than one process to manufacture a product. Hence, formulating the problem based on each process could be avoided. In view of this, to solve SLPP and MUIPP models, strategies are proposed in this thesis by considering the production quantity of each product to be manufactured as decision variables. The proposed strategy aims to reduce the number of decision variables and the applicable constraints without compromising the rigor of the problem.

A critical review of the MUCPP model provided by Chauhan and Kotecha (2018) shows that the strategy employed to solve the multi-unit production planning (MUCPP) problem, modeled only using continuous variables, accounts for the quantity of the product produced by each of the available processes as the decision variable. However, the restriction on manufacturing a product using a single process (unique process constraints) makes most decision variables in the solution structure futile. Such a solution structure increases the problem dimension and requires additional constraint-handling techniques to satisfy the constraints related to the variable bounds. On considering the possibility of improvement in the strategy followed by Chauhan and Kotecha (2018), this thesis has proposed a strategy that approached the production planning problem from the perspective of the production quantity of each product. The manufacturing industries usually constitute more processes than the number of products, as multiple processes exist to produce a specific product. A solution structure based on the production quantity of each product instead of the production quantity from each process reduces the number of decision variables in the proposed strategy compared to the strategy in the literature. An appropriate approach has been employed in the proposed strategy to ensure the usage of only one process to manufacture the product. This resulted in the inherent satisfaction of unique process constraints and thus reduced the number of applicable constraints.

The role of the investment budget for the successful establishment and operation of the capital-intensive petrochemical industries cannot be neglected. The efficient allocation of the investment budget helps in long-term planning and strategic decision-making of the petrochemical industries. Even though the production planning model studied in the literature (Alfares & Al-Amer, 2002; Chauhan & Kotecha, 2018) considered the developing stage of the petrochemical industry, it never analyzed the impact of investment budget while attaining better profit. The studies performed in the literature have solved the production planning

problem as a single objective optimization problem, imparting restrictions on the investment budget in addition to other constraints. However, solving these literature models might lead to a solution that can have better profit with a larger investment cost, while a solution with a similar profit can be achieved with a lesser investment cost. To avoid such a scenario, and considering the vital nature of investment budget in the success of the petrochemical industry, this thesis explores the various possibilities of production plans by solving the MUCPP model with two conflicting objectives: maximization of profit and minimization of investment cost.

Another sort of combinatorial problem considered in this thesis is scheduling jobs on multiple parallel machines where each job possesses release and due dates. This study has solved multiple instances of a scheduling problem with varying complexity using different metaheuristic techniques. The objective of the scheduling problem is to determine an optimal schedule with minimum cost by satisfying all the overlapping, release date, and due date constraints. The scheduling model in Kotecha et al. (2009) is solved with metaheuristic techniques, and static penalties are used to prefer a feasible solution to an infeasible one. However, it has been observed that such a constraint-handling strategy is not efficient enough for most algorithms with an increase in the complexity of scheduling problems. This thesis has proposed a no-wait time heuristic mechanism to assist metaheuristic techniques by rescheduling the jobs assigned to the machines based on the potential schedule provided by the technique. The rescheduling of jobs based on the proposed heuristic mechanism attempts to minimize the constraint violation, resulting in the discovery of feasible solutions in the earlier stages of metaheuristic techniques. The metaheuristic technique does not require modifications to incorporate the heuristic mechanism. The model of the scheduling problem in (Kotecha et al., 2009) has incorporated the horizon constraints while solving it using metaheuristic techniques without addressing their effects. Hence, this work also analyzed the impact of tightening constraints on metaheuristic techniques with the proposed heuristic mechanism.

Apart from identifying the optimal schedule with minimum processing cost, it is also crucial for the scheduling problem to determine a schedule that completes every job as quickly as possible, measured as makespan. Minimizing the makespan might result in assigning jobs to a costlier machine, as processing jobs on faster machines is often expensive. In this direction, the multi-objective scheduling problem is solved in this thesis by considering the minimization of processing cost and minimization of makespan as the two objectives. It is inappropriate to neglect the need to determine an optimal schedule with a minimum makespan, as it plays a vital role in improving the overall efficiency and productivity of a firm. The benefits of optimizing the scheduling problem by considering minimization of makespan as the objective include efficient utilization of the available machines, faster completion of all the jobs, load balancing, and increased productivity, which results in improved system efficiency. This thesis also proposed a new multi-objective variant of phase-wise teaching learning-based optimization to solve the scheduling problem.

Researchers have solved the vapor compression absorption cascade refrigeration system (CACRS) optimization involving absorbent solution-refrigerant pairs with different refrigerants as individual optimization problems, and the best choice is selected manually. This process increases computational complexity and does not fully utilize the potential of many recent optimization algorithms. The optimization of the CACRS involves complex nonlinear models and is often to be solved using metaheuristic techniques. These are efficient solution techniques for nonlinear optimization problems; however, they cannot guarantee their efficiency in all types of optimization problems (Wolpert and Macready 1997). Additionally, it has been observed that the application of multiple algorithms could benefit in determining robust solutions to complex optimization problems (Lian, Yan, and Wang 2022; Pati and Verma 2021). In this regard, this thesis provides an efficient optimization solution strategy for

the CACRS that reduces the complexity of using optimization techniques to select the best absorbent-refrigerant-refrigerant pair along with other operating decisions.

A significant drawback of the metaheuristic techniques is that they require considerable computational effort and repeatedly evaluate the objective function. The use of metaheuristic techniques becomes prohibitive when the evaluation of objective function is computationally intensive, and the utilization of parallel computing usually overcomes it. Most metaheuristic techniques, such as Genetic Algorithm (GA), Particle Swarm Optimization (PSO), etc., are inherently parallelizable, implying that the objective function of the entire population in every iteration can be evaluated in parallel. In this direction, very few works related to the TLBO are discussed in the literature. The inherent nature of TLBO requires the sequential generation and immediate evaluation of new solutions to update the population. This feature restricts TLBO from utilizing parallel computing. For example, to solve an optimization problem, a single function evaluation requires one second, and the population size (N_P) and the number of generations (N_G) considered are 100. Then, the amount of time required by TLBO for evaluating the objective function (without duplicate removal and excluding the evaluation of the initial population) for N_G generations is 20,000 seconds. Despite the availability of parallel computational resources, the time required to evaluate the objective function cannot be reduced, as it requires the evaluation of $2N_PN_G$ population members, which are generated sequentially. To the best of our knowledge, no work discusses this vital aspect of TLBO, which can potentially limit its applicability to large-scale problems. This work addresses this lacuna by providing three alternate strategies that can substantially reduce computational time by harnessing the available resources. Each strategy modifies the TLBO algorithm to harness the benefits of parallel computing at varying levels.

It is a common practice to propose a variant to extend a single objective metaheuristic technique to solve multi-objective problems. A large number of multi-objective optimization problems (MOOPs) occur widely in nature, and researchers have explored the potential of TLBO to solve them. Rao and Waghmare (2014) presented a comparative study of TLBO on multi-objective unconstrained and constrained functions, but the article (Rao & Waghmare, 2014) did not explain the strategy employed in TLBO to solve MOOP. Rao and Patel (2014) proposed Multi-objective Improved Teaching Learning Based Optimization (MO-ITLBO) that extended the Improved Teaching Learning Based Optimization (ITLBO) algorithm to solve MOOP based on the epsilon domination method of Deb et al. (2005). Patel and Savsani (2014b) proposed and used MO-ITLBO to design a plate-fin heat exchanger. Patel and Savsani (2016) also proposed MO-ITLBO which also extended the ITLBO to solve MOOP based on the epsilon domination method of Deb et al. (2005) and also provided statistical analysis.

Though the three works (Patel & Savsani, 2014b; Patel & Savsani, 2016; Rao & Patel, 2014) seem to be utilizing the same MO-ITLBO algorithm, none of these works convincingly explain the extension of ITLBO to MO-ITLBO. Even though the proposed algorithm is to solve MOOPs, Rao and Patel (2014) explained the working of ITLBO for the Single Objective Optimization Problems (SOOP), which was already shown in their previous work (R. V. Rao & Vivek Patel, 2013) in which ITLBO was originally proposed. This thesis points out issues that have not been explicitly addressed in Patel and Savsani (2016). Since the MO-ITLBO presented by Patel and Savsani (2016) is similar to the algorithm of Rao and Patel (2014) and Patel and Savsani (2014b), comments in this study are valid for all three works.

2.7 Objectives of the thesis

The primary objective of this thesis is to analyze the effectiveness of metaheuristic techniques in solving a wide range of problems, including combinatorial problems, nonlinear problems,

computationally intensive problems, and multi-objective optimization problems. To summarize, the research objectives of this work are enumerated below:

i) Propose a better solution structure to solve production planning models.

ii) Analyze the trade-off between profit and investment cost.

iii) Propose an optimization framework that reduces the wait time among jobs to solve the scheduling of jobs on dissimilar parallel machines.

iv) Analyze the trade-off between total processing cost and makespan.

v) Propose an efficient solution structure for the CACRS model that reduces computational effort.

vi) Propose variants of the teaching-learning-based optimization that can harness the benefits of parallel computing.

vii) Critically analyze MO-iTLBO to identify the discrepancies in its implementation.

The thesis structure is depicted in Fig. 2.1.

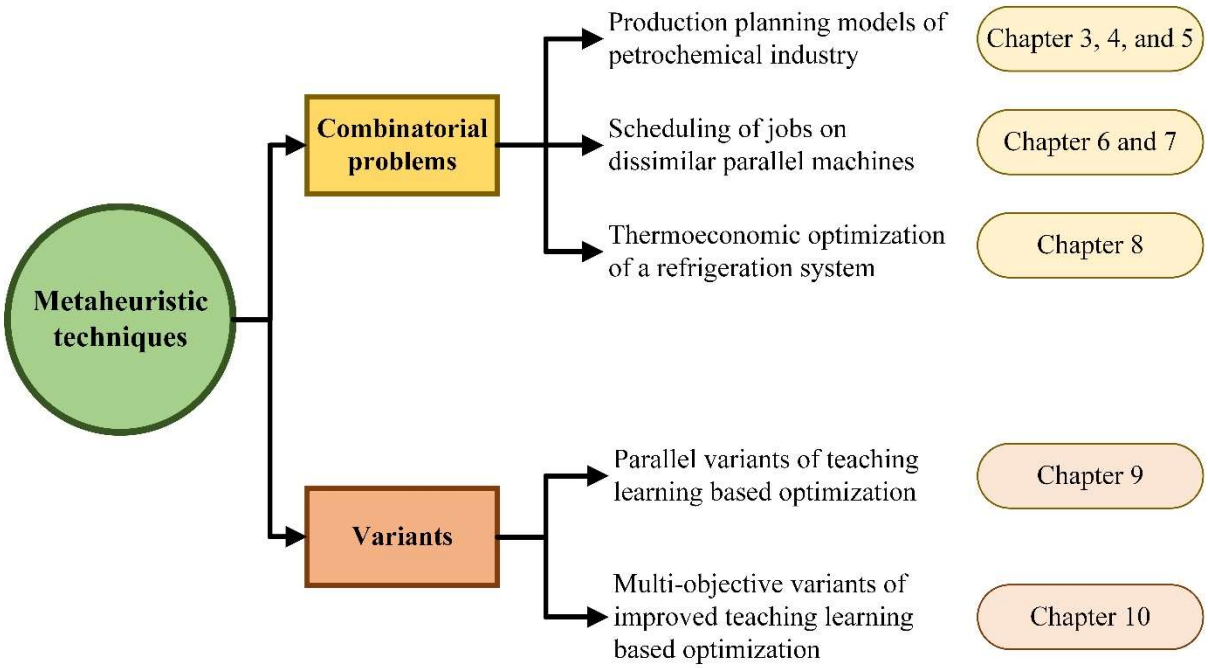


Fig. 2.1 Structure of the thesis

Chapter 3

An effective strategy for solving single and multi-unit production planning models with unique process constraints using metaheuristic techniques

In this chapter, production planning models presented in the literature, namely, single-level production planning (SLPP) in Chauhan and Kotecha (2020) and multi-unit production planning (MUIPP) in Chauhan et al. (2018), are thoroughly analyzed, and their limitations were identified. The possibilities of improving these models in terms of complexity and determining better optima are studied, and new solution strategies are proposed for both models. The efficacy of the proposed strategies is analyzed by solving the multiple cases of the production planning models using different metaheuristic techniques. This chapter is organized as follows: Section 3.1 briefly explains the production planning model studied in this work. Section 3.2 explains the strategy in the literature used for solving the SLPP and the MUIPP models. Section 3.3 of this article describes the proposed strategy for the SLPP model, followed by the proposed strategy for the MUIPP model. Section 3.4 provides all the experimental settings, and the results and discussions are provided in Section 3.5. Finally, the article is concluded in Section 3.6 by specifying the major findings of this study.

3.1. Problem statement

The production planning model studied in Alfares and Al-Amer (2002) constitutes I products that can be produced using J processes. There might be single or multiple predefined processes to manufacture a particular product. For each process, the production and the investment costs at three specific capacities, namely low-level (l), medium level (m), and high-level (h) are known. A piecewise linear relationship is assumed between the production capacities and costs, as depicted in Fig. 3.1. The production costs at l_j, m_j, and h_j of j^{th} process are C_{lj}, C_{mj}, and $C_{hj,}$ respectively, and their respective investment costs are V_{lj}, V_{mj}, and V_{hj}.

Kommadath, R., D. Maharana, and P. Kotecha, *An effective strategy for solving single and multi-unit production planning models with unique process constraints using metaheuristic techniques.* Expert Systems with Applications, 2023. **224**: p. 119813. DOI: https://doi.org/10.1016/j.eswa.2023.119813

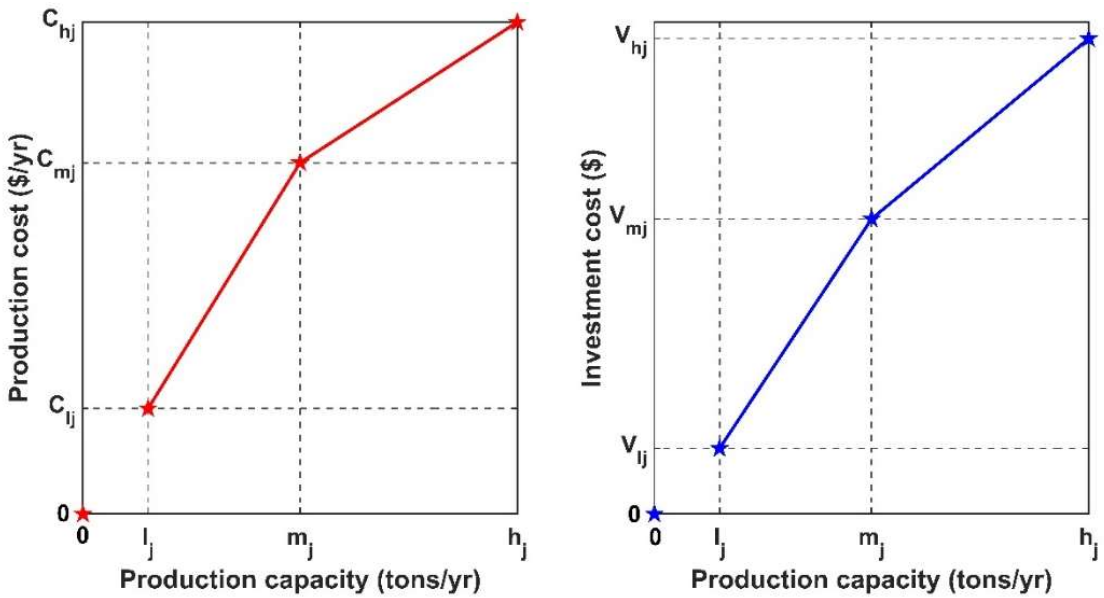


Fig. 3.1 Variation of production and investment costs with production capacity of j^{th} process

The objective is to determine an optimal production plan that maximizes the profit subject to the following constraints.

- *Investment budget constraint:* The total investment cost of a potential production plan should not exceed the available investment budget.
- *Raw material constraints:* The quantity of raw materials used to produce products specified in a production plan must be less than or equal to the quantity of raw materials available.
- *Domain hole constraints:* The production of a product by a process must occur between its low-level and high-level capacity. However, a process can remain unused, indicating the production from that process is zero. Any production quantity between zero and the low-level capacity of a process is invalid, and this discontinuous region acts as the domain hole. The domain hole constraints limit the production of any process in its invalid production capacities.
- *Unique process constraints:* The unique process constraint restricts the usage of more than one process to produce a product.

3.2. Literature models

Two production planning models, namely single-level production planning (SLPP) in Chauhan and Kotecha (2020) and multi-unit production planning model with integer and continuous decision variables (MUIPP) in Chauhan et al. (2018), primarily differ in number and valid implementation region of processing units. The details related to these models and their solution representation used in the literature are provided below.

3.2.1. Single-level production planning model

In the SLPP model presented in Chauhan and Kotecha (2020), a process is allowed to produce the product either in the *l-m* or *m-h* production capacity. The structure of the production plan, along with the lower bound (*lb*) and upper bound (*ub*) of the decision variables used in Chauhan and Kotecha (2020), are shown in Fig. 3.2.

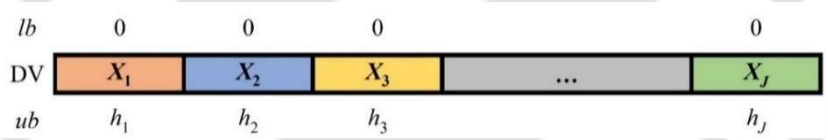


Fig. 3.2 Solution structure of the SLPP model used in the literature

The j^{th} decision variable (*DV*) in the production plan represents the amount of product that has to be manufactured using the j^{th} process. Hence, the number of decision variables is equal to the total number of processes. In Fig. 3.2, the parameter h_j indicates the high-level production capacity of the j^{th} process.

3.2.2. Multi-unit production planning model with integer and continuous variables

In the MUIPP model provided in Chauhan et al. (2018), a process is allowed to manufacture a product by implementing one processing unit anywhere between *l-m* and *m-h* capacity levels. Additionally, multiple process units are allowed to implement at *l*, *m*, and *h* capacities for manufacturing the product. Unlike the single-level, where a process is allowed to manufacture the product by implementing a single unit in the valid production domain, in the MUIPP model,

a process can produce the product by implementing multiple units of it. The solution structure of the MUIPP model used in Chauhan et al. (2018) and the lower bound (*lb*) and the upper bound (*ub*) of the decision variables are provided in Fig. 3.3. The parameter B indicates the total investment budget, m_j represent the medium-level and h_j is the high-level capacity of the j^{th} process. V_{lj}, $V_{mj,}$ and V_{hj} indicate the investment cost of j^{th} process at l, m, and h production capacity levels, respectively.

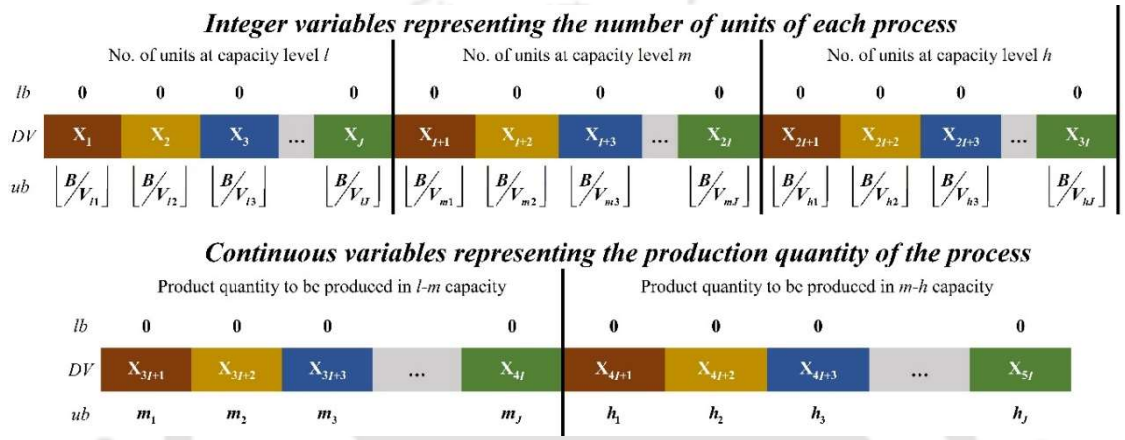


Fig. 3.3 Solution structure of the MUIPP model employed in the literature

The two models specified above differ in considering the number of processing units to be implemented in the permissible production capacity. In the MUIPP model, one unit each can be implemented in the *l*-*m* and *m*-*h* production capacity, while in the SLPP model, only one unit is permitted in the *l*-*h* production capacity. Additionally, the provision for implementing multiple units at *l*, *m*, and *h* production capacity of any process is incorporated in the MUIPP model. The comparison of the SLPP and MUIPP models, based on the permissible production units and the capacity, is provided in Table 3.1.

Table 3.1 Comparison of SLPP and MUIPP model

Model	**Maximum number of units**	**Permitted region for implementing production units**
SLPP	1	Anywhere in the region between *l*-*h* production capacity
MUIPP	Restricted by resource availability	Multiple units at *l*, *m*, and *h* capacities and single unit each in the regions between *l*-m and *m*-*h* capacities.

The number of decision variables in both models depends on the number of permissible production units and the total number of processes. As the possibilities of producing a product by the process are higher in MUIPP than in the SLPP model, the number of decision variables is higher in the MUIPP models. In MUIPP, one unit each is permitted to implement in *l-m* and *m-h* production capacity, due to which the number of continuous variables is twice that of the SLPP model. The number of domain hole constraints in the MUIPP model is higher than in the SLPP model, as a domain constraint exists corresponding to each continuous variable.

3.3. Proposed strategy for the production planning models

Fundamentally, unique process constraints restrict multiple processes in producing one type of product. However, the strategies in the literature inherently accept multiple processes for a single product and rely on penalty approaches for realizing the constraints. The proposed strategy addresses the above shortcoming by providing an efficient solution approach where the decision variables represent the amount of product to be produced and a strategy to select an appropriate process to manufacture the product. The strategies proposed to solve the two models with unique process constraints are explained in subsequent sections.

3.3.1. Single-level production planning model

The proposed strategy for solving the SLPP model with unique process constraints considers the amount of each product to be produced as the decision variable. The structure of a potential solution utilized by the proposed strategy to solve the SLPP model with unique process constraints is provided in Fig. 3.4.

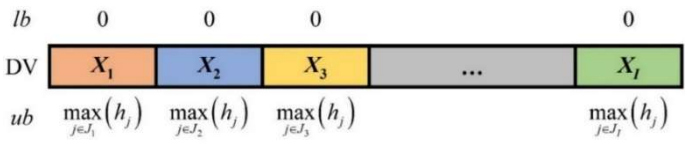


Fig. 3.4 Proposed solution structure for SLPP model

Each decision variable indicates the amount of product to be produced from any of the allowed processes between their lower and higher-level capacities. Hence, the total number of decision

variables corresponding to the model is equal to the maximum number of products (I) to be produced. The lower bound of all the decision variables is zero. The upper bound corresponding to each decision variable is the maximum capacity value among the high-level capacity of all the processes allowed to produce a product. The bounds of i^{th} variable are given in Eq. (3.1)

$$\begin{aligned} lb_i &= 0 \qquad && \forall i \in \{1,2,...,I\} \\ ub_i &= \max_{j \in J_i}\left(h_j\right) && \forall i \in \{1,2,...,I\} \end{aligned} \tag{3.1}$$

where J_i is the set of processes that are allowed to produce the i^{th} product, and I is the total number of products. As some of the products can be produced using multiple processes, the profit on producing the production quantity specified in the potential solution using all the processes is determined, and the most profitable process is selected. The proposed strategy avoids those potential processes where a lower bound violation could occur during the profit calculation and assigns zero production to such a process for the i^{th} product as Eq. (3.2).

$$Y_j = \begin{cases} 0 & if\ 0 < X_i < l_j \\ X_i & else \end{cases} \quad \forall j \in J_i,\ \forall i \in \{1,2,...,I\} \tag{3.2}$$

Each decision variable can possess any value between zero to the high-level production capacity as per the proposed strategy. However, a scenario may arise where the i^{th} decision variable can have a value greater than the high-level production capacity of the processes allowed to produce the i^{th} product. The decision variable violating high-level production capacities for any potential process is modified as per Eq. (3.3). The variable Y_j is used to indicate the production quantity of the product.

$$Y_j = \begin{cases} h_j & if\ X_i > h_j \\ X_i & else \end{cases} \quad \forall j \in J_i,\ \forall i \in \{1,2,...,I\} \tag{3.3}$$

The requirement of these correction approaches for rectifying the bounds is due to different valid production regions for the potential processes corresponding to each product. The purpose of modifying the decision variables using Eq. (3.2) or Eq. (3.3) is to ensure that their value is

within the valid region of the process used to manufacture the product. Without these bound correction approaches, selecting profitable processes (α_i) for the i^{th} product and determining total profit would be inaccurate. If the value of any decision variable is zero, then there is no need to determine the profitable process for that variable.

Determination of production cost and investment cost: The production cost (C_j) and the investment cost (V_j) between the known capacities of the j^{th} process are calculated using the piecewise linear relation given in Eq. (3.4) and Eq.(3.5), respectively.

$$C_j = \begin{cases} C_{lj} + \dfrac{\left(Y_j - l_j\right)\left(C_{mj} - C_{lj}\right)}{\left(m_j - l_j\right)} & if\ l_j \le Y_j \le m_j \\ C_{mj} + \dfrac{\left(Y_j - m_j\right)\left(C_{hj} - C_{mj}\right)}{\left(h_j - m_j\right)} & if\ m_j \le Y_j \le h_j \quad \forall j \in J_i,\ \forall i \in \{1,2,...,I\} \\ 0 & Y_j = 0 \end{cases} \tag{3.4}$$

$$V_j = \begin{cases} V_{lj} + \dfrac{\left(Y_j - l_j\right)\left(V_{mj} - V_{lj}\right)}{\left(m_j - l_j\right)} & if\ l_j \le Y_j \le m_j \\ V_{mj} + \dfrac{\left(Y_j - m_j\right)\left(V_{hj} - V_{mj}\right)}{\left(h_j - m_j\right)} & if\ m_j \le Y_j \le h_j \quad \forall j \in J_i,\ \forall i \in \{1,2,...,I\} \\ 0 & Y_j = 0 \end{cases} \tag{3.5}$$

The revenue (S_j) from the i^{th} product produced using j^{th} process is calculated using Eq. (3.6), where E_j represents the unit selling price of the product manufactured using j^{th} process.

$$S_j = Y_j E_j \quad \forall j \in J_i,\ \forall i \in \{1,2,...,I\} \tag{3.6}$$

The profit obtained for producing the i^{th} product using the j^{th} process is calculated using Eq. (3.7). The process that provides maximum profit (α_i) among all the potential processes of i^{th} product is selected for manufacturing it.

$$Profit_j = S_j - C_j \quad \forall j \in J_i,\ \forall i \in \{1,2,...,I\} \tag{3.7}$$

The penalties for violating the budget and raw material constraints are calculated after selecting the profitable process for each non-zero variable. The associated costs and raw material requirements are calculated based on the value of the modified decision variable.

Investment budget constraint: A penalty ($P^{Investment}$) for violating the investment budget constraint is determined using Eq. (3.8), where B indicates the investment budget and V_i is the investment cost required to implement the profitable process for producing the i^{th} product.

$$P^{Investment} = \begin{cases} \left(B - \sum_{i=1}^{I} V_{\alpha_i} \right)^2 & if \sum_{i=1}^{I} V_{\alpha_i} > B \\ 0 & else \end{cases} \tag{3.8}$$

Raw material constraints: The total amount of k^{th} raw material (r^k) required for a production plan is determined using Eq. (3.9).

$$r^k = \sum_{i=1}^{I} b_{\alpha_i}^k X_i \tag{3.9}$$

Here $b_{\alpha_i}^k$ is the amount of k^{th} raw material required for unit production of the i^{th} product using its profitable process. The penalty for the violation of raw material constraints (P^{Raw}) is calculated using Eq. (3.10).

$$\begin{aligned} P_k^{Raw} &= \begin{cases} \left(R^k - r^k \right)^2 & if\ r^k > R^k \\ 0 & else \end{cases} \quad \forall k \in \{1, 2, ..., K\} \\ P^{Raw} &= \sum_{k=1}^{K} P_k^{Raw} \end{aligned} \tag{3.10}$$

As the proposed strategy utilizes the profitable process to manufacture the products, it would inherently satisfy the unique process constraints and does not require separate constraint handling techniques for these constraints. The total penalty for violating the constraints is calculated using Eq. (3.11), and the fitness of a potential schedule is determined using Eq. (3.12), where (λ) is the penalty factor.

$$P^{Total} = P^{Investment} + P^{Raw} \tag{3.11}$$

$$f = \begin{cases} \sum_{i=1}^{I} Profit_{\alpha_i} & if\ P^{Total} = 0 \\ \sum_{i=1}^{I} Profit_{\alpha_i} - \lambda P^{Total} & else \end{cases} \quad (3.12)$$

A detailed pseudo code of the solution strategy is given in Fig. 3.5. The overall optimization formulation of the MUIPP model is provided in Eq. (3.13).

Input: Potential solution (X)
Output: Corrected solution (X_c) *and Fitness value* (f)

For i = 1 to I
  Identify the set of processes (J_i) that can produce the i^{th} product
  For n = 1 to $|J_i|$
    Set $j = J_i^n$
    Assign $Y_j = X_i$
    If $(0 < Y_j < l^j)$
      Set $Y_j = 0$
    Else
      If $(Y_j > h^j)$
        $Y_j = h^j$
      End If
      Identify the capacity region in which the Y_j has to be produced
      Determine the production cost (C_j) using Eq. (3.4)
      Determine the selling price (S_j) using Eq. (3.6)
      Determine the profit (P_j) using Eq. (3.7)
    End if
  End For
  Determine the profitable process (α_i) among J_i
  Assign $X_i = Y_{\alpha_i}$
  Assign the profit $Profit_i = Profit_{\alpha_i}$
  Determine investment cost for producing X_i with the process α_i, using Eq. (3.5)
  Determine raw materials utilized by the process α_i to produce X_i using Eq. (3.9)
End For
Calculate the penalty for violating investment budget constraint using Eq. (3.8)
Calculate the penalty for violating raw material constraints using Eq. (3.10)
Calculate the fitness value using Eq. (3.12)
Assign $X_c = X$ and **return** X_c and f

Fig. 3.5 Pseudocode of the proposed strategy for the SLPP model

It should be noted that the formulation provided in Eq. (3.13) should be used in conjunction with the proposed strategy discussed above. The binary variable Z_i is used to demonstrate the unique process constraints and is not necessary for the proposed strategy.

$$
\begin{aligned}
&\textit{Maximize Profit}\\
&\textit{s.t.},\\
&\sum_{i=1}^{I} V_{\alpha_i} \le B\\
&\sum_{i=1}^{I} b_{\alpha_i}^{k} X_i \le R^k \quad \forall k \in \{1,2,...,K\}\\
&(X_i = 0) \vee l_{\alpha_i} \le X_i \le h_{\alpha_i} \quad \forall i \in \{1,2,...,I\}\\
&\sum_{j \in J_i} Z_j \le 1 \quad \forall i \in \{1,2,...,I\}\\
&0 \le X_i \le ub_i \quad \forall i \in \{1,2,...,I\}
\end{aligned}
\tag{3.13}
$$

3.3.2. Multi-unit production planning model with integer and continuous variables

In the MUIPP model, a process can produce the product from multiple units at the known capacities and a single unit between *l-m* and *m-h* capacity levels. In the proposed strategy, a potential solution provides information about the number of units of the process implemented at its known capacities and the amount of product produced using its single unit installed in the *l-m* and *m-h* capacity levels. The structure of a potential solution is provided in Fig. 3.6.

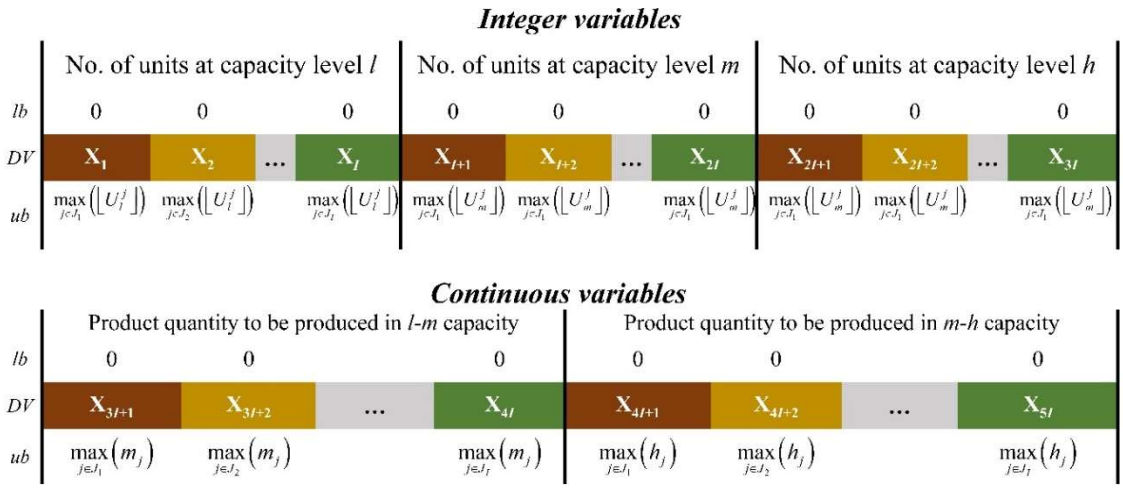


Fig. 3.6 Proposed solution structure for the MUIPP model

The profitable process for producing the product specified in the production plan is unknown a priori and has to be identified for each product. There would be five variables corresponding

to a product. The first set of I variables indicates the number of units of the selected process that have to be implemented at its capacity to produce each I product. Similarly, the following two sets with I variables indicate the number of units of the selected process implemented at its m and h capacities, respectively. The last two sets of decision variables represent the production quantity of each product to be manufactured using the profitable process in its l-m and m-h production capacities, respectively.

The lower bound of all the decision variables is equal to zero, and if the value of any decision variable is zero, it implies that the product is not produced. The first $3I$ decision variables are integers indicating the number of units of the profitable process implemented at known production levels. The maximum number of process units for all the processes of the i^{th} product that can be implemented at their known capacities is determined using Eq. (3.14).

$$\left.\begin{aligned} U_l^j &= \min\left(\left\lfloor B/V_{lj} \right\rfloor, \min_{k\in\{1,2,..,K\}}\left(\left\lfloor \frac{R_k}{b_j^k l_j} \right\rfloor\right)\right) \\ U_m^j &= \min\left(\left\lfloor B/V_{mj} \right\rfloor, \min_{k\in\{1,2,..,K\}}\left(\left\lfloor \frac{R_k}{b_j^k m_j} \right\rfloor\right)\right) \\ U_h^j &= \min\left(\left\lfloor B/V_{hj} \right\rfloor, \min_{k\in\{1,2,..,K\}}\left(\left\lfloor \frac{R_k}{b_j^k h_j} \right\rfloor\right)\right) \end{aligned}\right\} \quad \forall j \in J_i,\ \forall i \in \{1,2,...,I\} \tag{3.14}$$

The upper bound of the first $3I$ integer variables is calculated as per Eq.(3.15).

$$ub_t = \begin{cases} \max_{j\in J_i}\left(\left\lfloor U_l^j \right\rfloor\right) & if\ 1 \le t \le I \\ \max_{j\in J_i}\left(\left\lfloor U_m^j \right\rfloor\right) & elseif\ I+1 \le t \le 2I \\ \max_{j\in J_i}\left(\left\lfloor U_h^j \right\rfloor\right) & else\ 2I+1 \le t \le 3I \end{cases} \quad \forall t \in \{i, I+i, 2I+i\}, \forall i \in \{1,2,...,I\} \tag{3.15}$$

The upper bound for the last $2I$ continuous decision variables indicating the production quantity at l-m and m-h production levels is determined using Eq. (3.16).

$$ub_t = \begin{cases} \max_{j\in J_i}\left(m_j\right) & if\ 3I+1 \le t \le 4I \\ \max_{j\in J_i}\left(h_j\right) & else\ 4I+1 \le t \le 5I \end{cases} \quad \forall t \in \{3I+i, 4I+i\}, \forall i \in \{1,2,...,I\} \tag{3.16}$$

The profit for producing the product quantity specified in the potential solution must be calculated for each allowed process. The process providing maximum profit is selected to manufacture the product, and the rest of the processes are neglected. Suppose the production quantity provided in the potential solution violates the bounds of any process that produces the i^{th} product. In that case, it is modified based on the lower and upper production limits of that process. This bound rectified value (Y) is used to determine the profitable process. The bound violation of an integer variable with respect to the j^{th} process is corrected as given in Eq. (3.17) where N_i is the total number of processes for producing the i^{th} product.

$$\left.\begin{aligned} Y_j &= \begin{cases} U_l^j & if\ X_i > U_l^j \\ X_i & else \end{cases} \\ Y_{j+N_i} &= \begin{cases} U_m^j & if\ X_i > U_m^j \\ X_i & else \end{cases} \\ Y_{j+2N_i} &= \begin{cases} U_h^j & if\ X_i > U_h^j \\ X_i & else \end{cases} \end{aligned}\right\}, \forall j \in J_i, \quad \forall i \in \{1,2,...,I\} \tag{3.17}$$

Similarly, the domain violation of the continuous decision variable for the j^{th} process representing the product quantity to be manufactured in the *l-m* capacity level is handled using Eq. (3.18), and that of the *m-h* capacity level is modified using Eq. (3.19).

$$Y_{j+3N_i} = \begin{cases} 0 & if\ 0 \le X_i \le l_j \\ m_j & else\, if\ X_i > m_j \\ X_i & else \end{cases} \quad \forall j \in J_i, \quad \forall i \in \{1,2,...,I\} \tag{3.18}$$

$$Y_{j+4N_i} = \begin{cases} 0 & if\ 0 \le X_i \le m_j \\ h_j & else\, if\ X_i > h_j \\ X_i & else \end{cases} \quad \forall j \in J_i, \quad \forall i \in \{1,2,...,I\} \tag{3.19}$$

The production cost of the i^{th} product manufactured using the j^{th} process at its *l*, *m,* and *h* capacities are calculated using Eq. (3.20).

$$\left.\begin{aligned} C_j^l &= Y_j C_{lj} \\ C_j^m &= Y_{j+N_i} C_{mj} \\ C_j^h &= Y_{j+2N_i} C_{hj} \end{aligned}\right\} \quad \forall j \in J_i, \quad \forall i \in \{1,2,...,I\} \tag{3.20}$$

The production cost for producing the i^{th} product in the *l*-*m* and *m*-*h* production levels is calculated using Eq. (3.21).

$$C_j^{lm} = \begin{cases} C_{lj} + \dfrac{\left(Y_{j+3N_i} - l_j\right)\left(C_{mj} - C_{lj}\right)}{\left(m_j - l_j\right)} & \text{if } l_j \le Y_{j+3N_i} \le m_j \quad \forall j \in J_i, \ \forall i \in \{1,2,...,I\} \\ 0 & \text{else} \end{cases}$$

$$C_j^{mh} = \begin{cases} C_{mj} + \dfrac{\left(Y_{j+4N_i} - m_j\right)\left(C_{hj} - C_{mj}\right)}{\left(h_j - m_j\right)} & \text{if } m_j \le Y_{j+4N_i} \le h_j \quad \forall j \in J_i, \ \forall i \in \{1,2,...,I\} \\ 0 & \text{else} \end{cases} \tag{3.21}$$

The parameter N_i in Eq. (3.20) and Eq. (3.21) indicates the total number of processes allowed to produce the i^{th} product. The total production cost required to manufacture the i^{th} product using the j^{th} process is determined using Eq. (3.22).

$$C_j = C_j^l + C_j^m + C_j^h + C_j^{lm} + C_j^{mh} \qquad \forall j \in J_i, \ \forall i \in \{1,2,...,I\} \tag{3.22}$$

The revenue on producing i^{th} product with the j^{th} process is calculated using Eq.(3.23).

$$S_j = E_j\left(Y_j + Y_{j+N_i} + Y_{j+2N_i} + Y_{j+3N_i} + Y_{j+4N_i}\right) \qquad \forall j \in J_i, \ \forall i \in \{1,2,...,I\} \tag{3.23}$$

The profit obtained from producing the i^{th} product using the j^{th} process is determined as per Eq. (3.7). In the proposed strategy, a potential production plan in the MUIPP model inherently satisfies the unique process constraints. The investment cost for the multiple units implemented at the *l*, *m*, and *h* capacity levels $\left(V_i^{MU}\right)$ of the profitable process (α) for producing the i^{th} product is determined using Eq. (3.24).

$$V_i^{MU} = X_i V_{l\alpha_i} + X_{I+i} V_{m\alpha_i} + X_{2I+i} V_{h\alpha_i} \quad \forall i \in \{1,2,...,I\} \tag{3.24}$$

The investment cost corresponding to the i^{th} product using a single unit $\left(V_i^{SU}\right)$ in the *l-m* and *m-h* production capacities of the profitable process is calculated using Eq.(3.25). The total investment cost (V_i) required to manufacture the i^{th} product is determined using Eq. (3.26).

$$V_i^{SU} = V_{(lm)_{\alpha_i}} + V_{(mh)_{\alpha_i}} \qquad \forall i \in \{1,2,...,I\}$$

$$where \quad V_{(lm)_{\alpha_i}} = V_{l\alpha_i} + \frac{\left(X_i - l_{\alpha_i}\right)\left(V_{m\alpha_i} - V_{l\alpha_i}\right)}{\left(m_{\alpha_i} - l_{\alpha_i}\right)} \tag{3.25}$$

$$V_{(mh)_{\alpha_i}} = V_{m\alpha_i} + \frac{\left(X_i - m_{\alpha_i}\right)\left(V_{h\alpha_i} - V_{m\alpha_i}\right)}{\left(h_{\alpha_i} - m_{\alpha_i}\right)}$$

$$V_i = V_i^{MU} + V_i^{SU} \tag{3.26}$$

The total amount of the i^{th} product manufactured $\left(X_i^{total}\right)$ using the profitable process is calculated using Eq. (3.27).

$$X_i^{total} = l_{\alpha_i} X_i + m_{\alpha_i} X_{I+i} + h_{\alpha_i} X_{2I+i} + X_{3I+i} + X_{4I+i} \qquad \forall i \in \{1,2,...,I\} \tag{3.27}$$

The amount of each raw material required to produce the total amount of i^{th} product is calculated using Eq. (3.28), where $b_{\alpha_i}^k$ is the amount of i^{th} product to be produced using the most profitable process.

$$r_i^k = b_{\alpha_i}^k X_i^{total} \qquad \forall i \in \{1,2,...,I\} \tag{3.28}$$

The penalty for violating constraints on the investment budget and raw materials is determined using Eq. (3.8) and Eq.(3.10), respectively. The total penalty assigned to a solution is calculated using Eq. (3.11), and its fitness value is determined using Eq. (3.12). Fig. 3.7 provides the stepwise procedure of the proposed strategy for the MUIPP model with unique process constraints. The representation of decision variables based on the proposed strategy has helped in reducing the number of decision variables, domain hole constraints, and unique process constraints in the MUIPP model. The overall optimization formulation of the MUIPP model is provided in Eq. (3.29). It should be noted that the formulation provided in Eq. (3.29) should be

used in conjunction with the proposed strategy discussed above. The binary variable Z_i is used to demonstrate the unique process constraints and is not necessary for the proposed strategy.

$$
\begin{aligned}
&\textit{Maximize Profit} \\
&\textit{s.t.}, \\
&\sum_{i=1}^{I} V_i^{MU} + V_i^{SU} \le B \\
&\sum_{i=1}^{I} b_{\alpha_i}^{k} X_i^{total} \le R^k \quad \forall k \in \{1,2,...,K\} \\
&\left(X_{3I+i} = 0\right) \vee l_{\alpha_i} \le X_{3I+i} \le m_{\alpha_i} \quad \forall i \in \{1,2,...,I\} \\
&\left(X_{4I+i} = 0\right) \vee m_{\alpha_i} \le X_{4I+i} \le h_{\alpha_i} \quad \forall i \in \{1,2,...,I\} \\
&\sum_{j \in J_i} Z_j \le 1 \quad \forall i \in \{1,2,...,I\} \\
&0 \le X_i \le ub_t, \quad \forall t \in \{i, I+i, 2I+i, 3I+i, 4I+i\}, \forall i \in \{1,2,...,I\}
\end{aligned}
\tag{3.29}
$$

The primary difference between the strategies proposed in this work and the strategy in the literature is the representation of decision variables. In the SLPP model, the decision variables selected in the literature represent the amount of product to be produced by each potential process. The availability of multiple processes for a single product will need the number of decision variables equivalent to the number of processes available for the product. In contrast, the decision variables in the proposed strategy indicate the amount of products to be produced and thus will need only one decision variable per product irrespective of the number of processes available for each product.

The choice of decision variables helps to satisfy the unique process constraints inherently. However, the strategy in the literature utilized the penalty-based approach to handle the unique process constraints. The penalty assigned for violating the constraint corresponding to a particular product is $\left(1000^{n_t}\right)$, where n_t is the number of additional processes employed to produce the product. The proposed strategy employed a correction approach for handling the domain hole constraints. In contrast, the strategy in the literature assigned a hard penalty of 10^5 for violating each constraint corresponding to a process.

Input: Potential solution (X)

Output: Corrected solution (X_c) *and fitness value* (f)

For i = 1:I

Identify the set of processes (J_i) that can produce the i^{th} product

Set $N = |J_i|$

For n = 1:N

Set $j = J_i^n$

Assign $Y_j = X_i, Y_{N+n} = X_{I+i}, Y_{2N+n} = X_{2I+i}, Y_{3N+n} = X_{3I+i}$ and $Y_{4N+n} = X_{4I+i}$

Rectify the bound violated variables using Eq. (3.17) – Eq. (3.19)

Calculate the production cost using Eq. (3.22)

Calculate the selling price using Eq. (3.23)

Calculate the profit using Eq. (3.7)

End For

Determine the profitable process (α_i) among J_i

Assign $X_i = Y_{\alpha_i}, X_{I+i} = Y_{N+\alpha_i}, X_{2I+i} = Y_{2N+\alpha_i}, X_{3I+i} = Y_{3N+\alpha_i}$ and $X_{4I+i} = Y_{4N+\alpha_i}$

Calculate the total amount of i^{th} product using Eq. (3.27).

Assign the profit from i^{th} product produced using process α_i as $Profit_i = Profit_{\alpha_i}$

Calculate the total investment cost of profitable process α_i to produce X_i^{total} amount of product using Eq. (3.26)

Determine amount of raw materials utilized by process α_i to produce X_i^{total} as Eq. (3.28)

End For

Calculate the penalty for violating the budget constraint using Eq. (3.8)

Calculate the penalty for violating the raw material constraints using Eq. (3.10)

Calculate the fitness f as Eq. (3.12)

Assign $X_c = X_i$ and **return** X_c and f

Fig. 3.7 Pseudocode of the proposed strategy for the MUIPP model

The strategy in the literature and the proposed strategy have utilized continuous as well as integer variables to formulate the MUIPP model. The strategy in the literature has selected integer variables as the number of units of a process that can be implemented at their known

capacity levels. Meanwhile, the integer variables in the proposed strategy considered the number of units of a beneficial process at its known levels. Similar to the SLPP model, continuous variables of the strategy in the literature specify the amount of product to be produced by each potential process. In the case of the proposed strategy, these variables indicate the amount of product to be produced. The proposed strategy of both SLPP and MUIPP models selects the most profitable process to manufacture a product. The decision structure followed in the proposed strategy of the MUIPP model helps to satisfy the unique process constraints inherently.

The solutions (X) determined by the metaheuristic techniques do not guarantee the satisfaction of all the associated constraints. In such a scenario, the use of a correction approach could be beneficial in avoiding the infeasible search space. The correction approach employed for handling the bound constraints ensures a solution that satisfies those constraints. However, the corrected solution should be communicated to the metaheuristic technique, which in turn will avoid the use of the infeasible solution in generating new solutions. The proposed strategy of both models uses correction approaches and communicates the modified solutions (X_c) with metaheuristic techniques. The communication between the metaheuristic techniques and the fitness function is represented in Fig. 3.8.

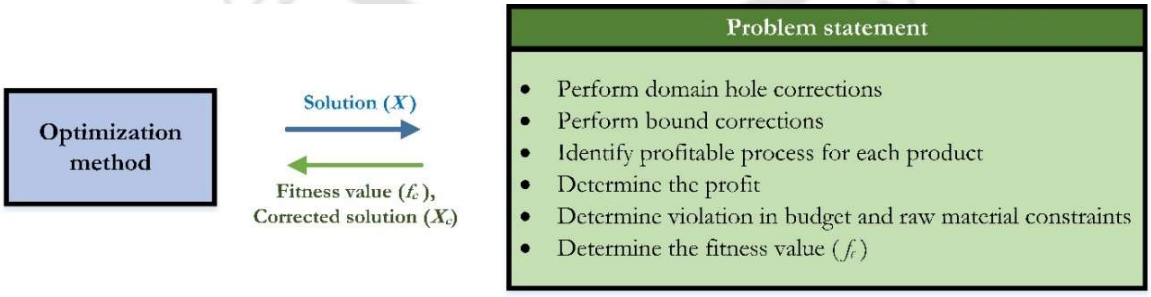


Fig. 3.8 Communication between metaheuristic technique and the fitness function

3.4. Experimental Settings

The case studies of the two models consider 24 products, which can be manufactured using 54 processes. The data regarding (i) the set of processes available to produce each product, (ii) investment budget, (iii) availability of raw materials, (iv) the amount of each raw material required to produce one unit of product using a particular process, and (v) the per unit selling price of each product is provided in the Appendix B (Table B1 to Table B4). The current work considers four cases on each model, as specified in Table 3.2. The maximum units that can be implemented at the given capacities are calculated based on the investment budget.

Table 3.2 Details of production planning case studies

Cases	Investment Budget ($ billion)	Raw Material 1 (million tons/year)	Raw material 2 (million tons/year)
Case 1	1	0.5	0.5
Case 2	1	1	1
Case 3	2	0.5	0.5
Case 4	2	1	1

In view of the difference in the investment budget, Case 1 has a different upper bound for the integer variables in the MUIPP model compared to Case 3. This inference is also valid in comparing the upper bounds of integer variables in Case 2 and Case 4. However, the bounds of the continuous variable for SLPP and MUIPP are independent of the investment budget. The penalty factor λ is set to 10^{15}. The metaheuristic techniques considered in this work are designed to solve optimization problems with continuous decision variables. Hence, the integer variables are handled separately using the round-off strategy.

The explanations for single-level and multi-unit production planning models presented in subsections 3.1.1 and 3.1.2, respectively, are obtained by carefully studying their respective literature. The models in the literature can be formulated by following the detailed explanation of the decision variables and applicable constraints. These models can be validated by testing

the solutions provided in the literature with the formulated model and comparing them with the objective function value. The solutions presented in the literature were tested with the developed model, and the results match the literature.

'No free lunch theorem for optimization' (Wolpert & Macready, 1997) states that for any algorithm, the improved performance in a set of problems is counterbalanced by the performance over another set of problems. This fact necessitates the use of multiple metaheuristic techniques for solving an optimization problem. We have considered five recently proposed metaheuristic techniques, namely Coyote Optimization Algorithm (COA) (Pierezan & Coelho, 2018), Chicken Swarm Optimization (CSO) (Meng et al., 2014), Grey Wolf Optimization (GWO) (Mirjalili et al., 2014), Jaya (Rao, 2016), and Simultaneous Heat Transfer Search (SHTS) (Maharana & Kotecha, 2016) to solve the production planning models. These techniques are selected due to their popularity and the availability of their implementation in the public platform, thereby avoiding any conflict in their implementation (Chinta et al., 2016; Črepinšek et al., 2012). In addition to these techniques, the authors have also considered the best-performing algorithm in the literature for solving SLPP and MUIPP, which are Moth Flame Optimization (MFO) (Mirjalili, 2015) and Sanitized Teaching Learning Based Optimization (sTLBO), respectively. A brief description of the metaheuristic techniques is given in Appendix A.

All the experiments regarding the SLPP and MUIPP models are simulated using the MATLAB 2019a environment installed on a desktop computer with Windows 10, a 64-bit operating system, i7 4.20 GHz processor, and 16 GB RAM. On considering the stochastic nature of the techniques, each case of the models is solved 25 times using the metaheuristic techniques. In all the instances, the population size for all the metaheuristic techniques except SHTS is 100. In SHTS, the population size (N) must be a multiple of six (Maharana & Kotecha, 2016) and

is set to 102. The termination criteria for all the techniques are consistent with the literature for the SLPP model (Chauhan & Kotecha, 2020) and the MUIPP model (Chauhan et al., 2018). The average computational time required by each technique to solve all cases of the SLPP and MUIPP models in a single run is approximately 10 seconds and 21 seconds, respectively.

This study analyses the efficacy of the proposed strategies using seven different techniques. It is common practice among researchers to evaluate the performance of metaheuristic techniques in solving various optimization problems using the parameter settings provided by their inventors (Faramarzi et al., 2020; Li et al., 2020; Shen et al., 2023). This study also employed the algorithm-specific parameters provided by the inventors to analyze the performance of metaheuristic techniques in solving the proposed strategies of both models. The algorithm-specific parameters in CSO, such as the number of roosters, hens, and mother hens, are taken as $0.15N$, $0.7N$, and $0.5N_h$, respectively, where N_h is the number of hens. The parameter of COA specifying the probability of leaving the pack is $\left(0.005N_c^{\,2}\right)$, where N_c is the number of coyotes, and the probability of a coyote birth is set as $\left(1/D\right)$, where D is the problem dimension. The exact parameter values of COA and CSO used to solve SLPP and MUIPP cases are provided in Table 3.3. Apart from COA and CSO, other techniques used in this study do not have any algorithm-specific parameters other than population size and termination criterion.

Table 3.3 Parameter values of different metaheuristic techniques

Algorithm	Parameters	Value
COA	Probability of leaving the pack	0.5
	Probability of coyote birth	0.04 for SLPP and 0.008 for MUIPP
CSO	Percentage of roosters in population	15
	Percentage of hens in population	70
	Percentage of mother hen among hens	50

3.5. Results and Discussion

This section discusses (i) the benefits of the proposed strategy in problem size, (ii) statistical analysis of the results provided by the five metaheuristic techniques on solving the model with the strategy provided in the literature, as well as using the proposed strategy, (iii) analysis of the best production plan obtained from each case of both the SLPP and MUIPP models using the proposed strategy.

3.5.1. Benefits of proposed strategies in problem size

The models employing the proposed strategies require less than half the number of decision variables compared to the strategies in the literature. The reduction in the problem size is due to the employed strategy, which utilizes the amount of product to be produced as the decision variable rather than the amount of product produced by each process. The comparison of problem dimensions in both SLPP and MUIPP case studies with 24 products and 54 processes is specified in Table 3.4.

Table 3.4 Comparison of problem dimension based on the literature and proposed strategy

		Literature		**Proposed**	
		SLPP (Chauhan & Kotecha, 2020)	**MUIPP** (Chauhan et al., 2018)	**SLPP**	**MUIPP**
Variables	Continuous	54	108	24	48
	Integer	–	162	–	72
Constraints	Domain hole	54	108	24	48
	Unique process	24	24	–	–

All the continuous variables have to satisfy the domain hole constraints, and the number of such variables in the model using the literature strategy is a multiple of the number of processes. However, as per the proposed strategy, the continuous variable represents the production quantity of the product from the selected process, and the total number of such variables would be a multiple of the number of products. Hence, the number of decision variables and the

domain hole constraints in the model utilizing the strategy provided in the literature is higher compared to the proposed strategy.

Like the strategies in the literature, an explicit constraint handling technique for unique process constraints is not required for the proposed strategy. The choice of decision structure (Fig. 3.4 and Fig. 3.6) and selecting the profitable process for producing a product cause the proposed strategy to inherently satisfy the unique process constraints. The number of unique process constraints in SLPP and MUIPP with the strategies in the literature is equal to the total number of products that can be produced. It is primarily due to the solution structure of the strategy in the literature, where the production capacity of each process is treated as the decision variable. The implicit handling of unique process constraints and the reduced number of domain constraints and decision variables in the proposed strategy have reduced the dimension of the SLPP and MUIPP models compared to the strategies employed in the literature.

3.5.2. Statistical analysis of results

The four case studies are solved using the strategy in the literature and five recently proposed metaheuristic techniques. The statistical results obtained by these techniques on solving both SLPP and MUIPP models are analyzed. The statistical results obtained by solving the case studies using the proposed strategies and the seven metaheuristic techniques are also studied. The best and mean objective value in each case are also compared to examine the efficacy of the proposed strategies with their respective strategy in the literature.

Statistical analysis of the results obtained using the strategy in the literature

The four case studies employing the strategy in the literature have been solved using five recently proposed metaheuristic techniques. The *best* (B), *mean* (M), and *median* (Md) statistical values obtained in solving the SLPP cases using the strategy in the literature by the metaheuristic techniques in 25 runs are provided in Table 3.5. The techniques, namely Moth

Flame Optimization (MFO) and Sanitized Teaching Learning Optimization (sTLBO) have determined the best results for SLPP in Chauhan and Kotecha (2020), which are also provided in Table 3.5. The best value among each statistic is indicated in bold typeface font, and the negative value in Table 3.5 indicates an infeasible production plan.

Table 3.5 Statistical results obtained on solving the SLPP cases using strategy in the literature

	Statistics	COA	CSO	GWO	Jaya	SHTS	Literature (Chauhan & Kotecha, 2020)	
							Best	Technique
Case 1	B	**678.38**	672.18	562.10	-1.0E+18	387.71	636.43	sTLBO
	M	**632.07**	557.84	369.91	-2.1E+20	95.53	499.54	sTLBO
	Md	**637.05**	554.95	328.71	-1.5E+20	54.85	512.35	MFO
Case 2	B	**759.74**	713.55	627.20	-1.0E+18	520.26	745.14	sTLBO
	M	**691.36**	610.32	-1.0E+20	-1.7E+20	134.60	571.59	sTLBO
	Md	**692.08**	612.80	408.35	-1.4E+20	96.30	596.44	MFO
Case 3	B	**870.57**	829.86	751.07	661.74	453.60	834.73	sTLBO
	M	**824.66**	730.10	-1.8E+20	-7.9E+19	162.34	695.18	sTLBO
	Md	**823.99**	729.59	494.90	-2.8E+19	103.40	699.51	sTLBO
Case 4	B	**1077.44**	1006.5	936.50	742.54	549.00	1047.69	MFO
	M	**1030.75**	905.01	-2.0E+20	-7.3E+19	185.20	888.35	sTLBO
	Md	**1037.16**	915.15	637.90	-1.5E+19	124.90	881.85	sTLBO

In all cases of SLPP model, COA determined better value in *best*, *mean*, and *median* than the other four metaheuristic techniques. It also provided better statistical measures than the value reported in the literature. It can be observed that the *mean* and the *median* determined by CSO are also better than the results determined by sTLBO in the literature. Even though the *best* determined by SHTS is not better than other techniques except Jaya, its feasible *mean* implies the determination of a feasible production plan in all the runs. In Case 1 and Case 2, Jaya is the only technique that is unable to determine even a single feasible solution among all the runs. In all the SLPP cases except Case 1, the *mean* obtained by GWO is infeasible, indicating GWO is unable to determine feasible solutions in all runs. However, the feasible *median* provided by GWO represents the determination of feasible solutions in more than half of the total runs. The number of runs that GWO reported infeasible solutions are two, two and five for Case 2, Case

3, and Case 4, respectively. The performance of Jaya is not satisfactory as it is unable to determine feasible solutions in Case 1 and Case 2 and provided feasible solutions in only two and five runs in Case 3 and Case 4 respectively.

The four cases of the MUIPP model are also solved using the five metaheuristic techniques, and the statistical results are compared with the results reported

(Chauhan et al., 2018). The *best*, *mean*, and *median* results obtained by all the techniques for solving MUIPP cases are given in Table 3.6. The details regarding the best reported in the literature for each statistic measure and the respective metaheuristic technique are also provided in Table 3.6.

Table 3.6 Statistical results obtained on solving MUIPP cases using strategy in the literature

	Statistics	COA	CSO	GWO	Jaya	SHTS	Value in literature (Chauhan et al., 2018) Best	Technique
Case 1	**B**	-1.0E+33	654.20	563.23	-1.0E+48	679.96	**709.95**	sTLBO
	M	-1.2E+56	527.33	-4E+58	-8.0E+73	594.28	**652.85**	sTLBO
	Md	-1.0E+45	523.80	403.68	-1.0E+60	606.34	**655.23**	sTLBO
Case 2	**B**	-1.0E+30	727.10	689.70	-1.0E+48	803.05	**839.25**	sTLBO
	M	-1.2E+56	631.03	-4E+58	-8.0E+73	707.27	**766.61**	sTLBO
	Md	-1.0E+45	674.08	503.95	-1.0E+60	717.40	**769.32**	sTLBO
Case 3	**B**	-1.0E+42	1122.90	965.30	-1.0E+57	1215.55	**1262.05**	sTLBO
	M	-4.0E+64	934.32	-4.0E+94	-4.0E+82	989.36	**1118.94**	sTLBO
	Md	-1.0E+54	959.00	-1.0E+63	-1.0E+75	993.93	**1111.23**	sTLBO
Case 4	**B**	-1.0E+42	1464.00	1223.73	-1.0E+57	1490.25	**1509.15**	sTLBO
	M	-4.0E+64	1234.00	-4.0E+94	-4.0E+82	1314.23	**1402.23**	sTLBO
	Md	-1.0E+54	1276.75	-1.0E+63	-1.0E+75	1303.95	**1398.41**	sTLBO

The statistical values of CSO and SHTS specify that these two techniques are able to determine feasible production plans in every run of all the cases. Among the five techniques, SHTS has determined better solutions followed by CSO. GWO determined a feasible solution corresponding to the *best* in all four cases of the MUIPP model, whereas the *mean* of GWO in every case is infeasible. GWO has determined a feasible *median* in Case 1 and Case 2, which indicates the determination of a feasible production plan in at least half of the total runs. The

total number of runs where GWO determined feasible solution is 18 and 19 for Case 1 and Case 2, respectively. However, the infeasible *median* of GWO in Case 3 and Case 4 implies that the technique has provided a feasible solution in less than half of the runs and which is equal to seven runs each. Unlike SLPP, where COA has determined better solutions in most cases, it is unable to identify even one feasible solution in any case of the MUIPP. The analysis of statistical results showed that the performance of Jaya in solving both of the production planning models was not satisfactory.

In all the cases where the MUIPP model was solved using the strategy presented in the literature, sTLBO identified better values in all the statistical measures. While solving MUIPP cases using the strategy in the literature, the metaheuristic techniques used in this work were unable to determine a better solution than the value reported in the literature. Hence, we have incorporated sTLBO in the rest of our analysis, along with five other techniques.

Statistical analysis of the results obtained using the proposed strategy

The four cases of SLPP and MUIPP models are solved by utilizing the proposed strategy with seven metaheuristic techniques, including MFO and sTLBO, along with five other techniques. The statistical results, *best*, *mean,* and *median*, of all these techniques while solving SLPP and MUIPP cases using the proposed strategies are given in Table 3.7. The best obtained by the techniques in solving every case of both production planning models using the strategy in literature is also provided in Table 3.7. It should be noted that the best results of the literature are obtained by considering the results of all seven techniques.

On considering the statistical results of SLPP cases, it is observed that the *best* identified by COA in all the cases is either better (Case 1, Case 3, and Case 4) or equal (Case 2) to the results given in the literature. In Case 2, the *best* determined by all techniques except GWO using the proposed strategy is equal to the *best* reported in the literature. In Case 4, all techniques except CSO and GWO determined a *best* better than the value reported in the literature. In all cases of

the SLPP model, the *best* obtained on solving the cases using the proposed strategy is better than or equal to the *best* identified using the strategy in the literature. For the SLPP, it is observed that the *mean* of all the techniques is feasible by solving using the proposed strategy, while it is not true with the techniques using the strategy in the literature. The feasible *mean* of all the techniques indicates the determination of feasible solutions in all the runs. It should be noted that the best solution determined by sTLBO and COA in all cases of SLPP model using the proposed strategy is in close proximity to those reported in Alfares and Al-Amer (2002), which uses the branch and bound technique.

Table 3.7 Statistical results of SLPP and MUIPP cases solved using proposed strategy

			Proposed Strategy							Literature Strategy	
			COA	CSO	GWO	Jaya	SHTS	MFO	sTLBO	Best	Technique
SLPP model	Case 1	B	**692.83**	690.82	683.74	692.83	692.76	692.83	**692.83**	678.38	COA
		M	675.29	668.27	593.50	669.00	632.30	673.89	**683.80**	632.07	COA
		Md	683.13	675.41	577.54	679.42	640.03	679.42	**688.01**	637.05	COA
	Case 2	B	**759.74**	**759.74**	759.73	**759.74**	**759.74**	**759.74**	**759.74**	**759.74**	COA
		M	746.19	746.82	654.55	729.67	704.92	750.36	**753.62**	691.36	COA
		Md	753.22	753.21	650.07	730.28	708.86	759.74	**759.74**	692.08	COA
	Case 3	B	**893.45**	892.39	880.67	893.45	886.45	893.17	**893.45**	870.57	COA
		M	**887.69**	854.45	810.00	874.15	805.00	868.26	879.95	824.66	COA
		Md	**890.91**	866.16	824.52	876.41	805.47	867.29	879.33	823.99	COA
	Case 4	B	**1111.48**	1110.49	1104.11	**1111.48**	**1111.48**	**1111.48**	**1111.48**	1077.44	COA
		M	**1104.79**	1084.91	1041.15	1097.45	1043.20	1102.29	1098.95	1030.75	COA
		Md	**1111.48**	1092.02	1041.30	1098.48	1058.08	1107.50	1098.48	1037.16	COA
MUIPP model	Case 1	B	**724.22**	668.90	644.30	707.75	671.25	621.51	707.75	709.95	sTLBO
		M	638.80	568.90	518.86	596.20	620.21	-9.5E+16	**663.50**	652.85	sTLBO
		Md	653.97	555.40	508.88	619.30	620.85	454.10	**676.85**	655.23	sTLBO
	Case 2	B	834.05	796.90	789.90	802.40	837.05	758.86	**844.35**	839.25	sTLBO
		M	761.81	702.03	643.50	668.68	759.07	571.60	**813.16**	766.61	sTLBO
		Md	780.45	720.80	665.45	730.33	757.85	549.80	**813.20**	769.32	sTLBO
	Case 3	B	1249.92	1200.20	1117.48	1234.15	1241.85	1140.28	**1265.15**	1262.05	sTLBO
		M	1152.50	990.90	895.07	-5.4E+19	1108.40	835.61	**1193.95**	1118.94	sTLBO
		Md	1169.77	1034.80	912.30	1077.77	1118.90	853.55	**1202.68**	1111.23	sTLBO
	Case 4	B	1510.85	1453.05	1348.15	**1514.55**	1488.75	1364.90	**1514.55**	1509.15	sTLBO
		M	1351.65	1203.33	1114.71	1327.86	1371.57	1075.09	**1423.66**	1402.23	sTLBO
		Md	1357.10	1195.30	1121.85	1388.75	1412.75	1132.92	**1416.45**	1398.41	sTLBO

Considering all the cases of the MUIPP model, sTLBO determined a better solution than other techniques except in Case 1. Moreover, the solutions determined by sTLBO for solving MUIPP

cases with the proposed strategy are better than the *best* reported in the literature. In Case 4, along with sTLBO, Jaya also determined the best production plan. The *mean* and the *median* of sTLBO on solving all MUIPP cases using the proposed strategy are better than the corresponding values obtained using the strategy in the literature. On using the proposed strategy, the *mean* of MFO in Case 1 and Jaya in Case 3 is infeasible, whereas the *median* of these techniques is feasible. Hence, it is evident that these techniques are able to determine a feasible production plan in at least half of the total runs. The bound correction as per the proposed strategies and the efficient handling of unique process constraints helped the metaheuristic techniques determine better solutions than the literature strategies.

Comparison of profit obtained using both strategies

The comparison of the *best* determined by each technique on SLPP cases using the literature and proposed strategy is provided in Fig. 3.9, and the same for MUIPP cases in Fig. 3.10. It should be noted that the profit of an infeasible solution is excluded from Fig. 3.9 and Fig. 3.10.

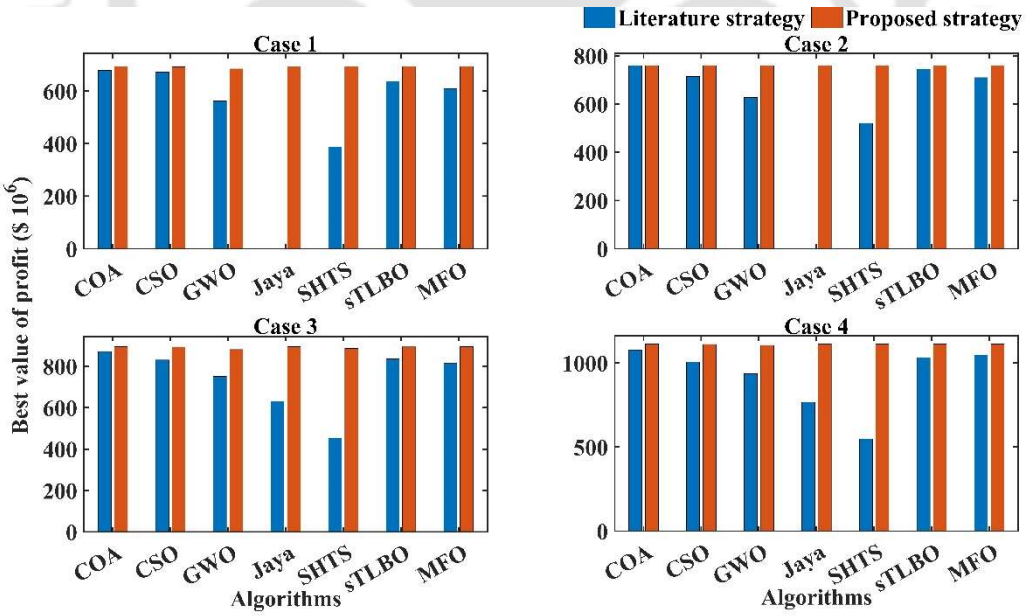


Fig. 3.9 Best profit in the SLPP cases

All the techniques are able to determine a feasible production plan with better profit in solving the SLPP cases using the proposed strategy than the strategy in the literature. Moreover, Jaya and SHTS demonstrated significant improvement in determining the best solution while using

the proposed strategy. In solving the SLPP model with the literature strategy, Jaya is unable to determine a feasible solution in Case 1 and Case 2. In contrast, it is able to determine the feasible production plan in these cases with the proposed strategy.

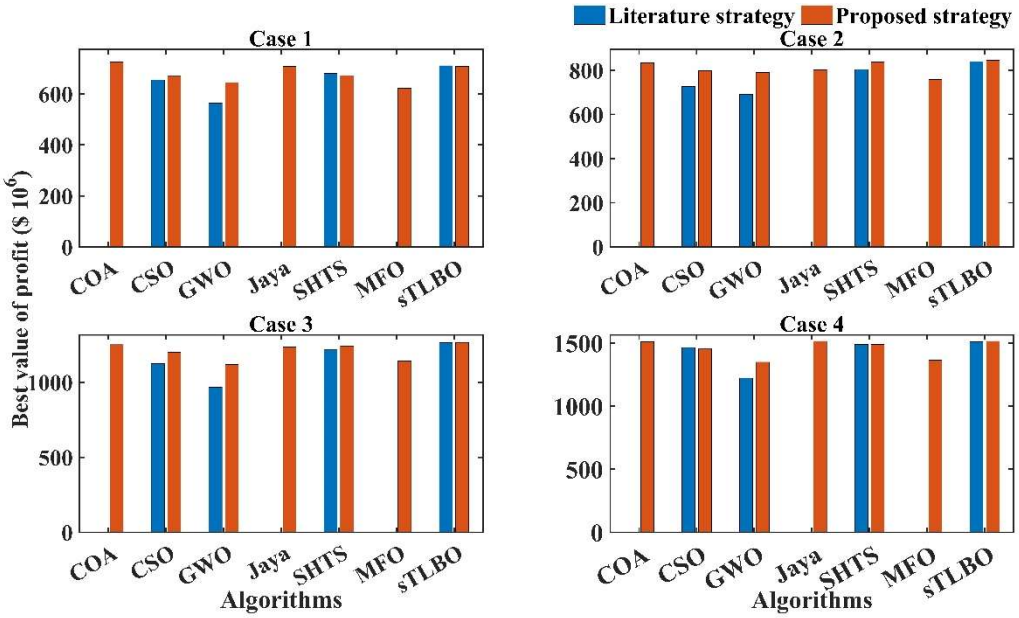


Fig. 3.10 Best profit in the MUIPP cases

In none of the MUIPP cases, techniques such as COA, Jaya, and MFO could determine a feasible production plan using the strategy in the literature. In contrast, all these techniques determined a feasible production plan using the proposed strategy.

The percentage improvement in *mean* obtained using the proposed strategy with respect to the strategy in the literature in every case of both models is provided in Table 3.8. As the *mean* depicts the overall performance of the metaheuristic techniques, it would help to analyze the benefits of the proposed strategy compared to the literature. The term 'Feasible' in Table 3.8 indicates the cases where the infeasible *mean* obtained using strategy in the literature is improved as a feasible *mean* with the proposed strategy. Similarly, 'Infeasible' represents the cases where none of the techniques are able to identify a feasible *mean* using both strategies.

On considering the SLPP cases, the *mean* obtained by Jaya using the proposed strategy is feasible, while it is infeasible with the strategy in the literature, which implies Jaya benefited

significantly from the proposed strategy. The other metaheuristic technique that has most benefited from the proposed strategy is GWO. SHTS has obtained a mere feasible *mean* in solving the SLPP cases using the strategy in the literature, whereas the *mean* has improved significantly with the help of the proposed strategy. Thus it can be observed that all metaheuristic techniques are able to improve their performance in all the cases of the SLPP model with the proposed strategy.

Table 3.8 Percentage improvement in *mean* using the proposed strategy over literature

	Technique	Case 1	Case 2	Case 3	Case 4
SLPP	COA	6.84	7.93	7.64	7.18
	CSO	19.79	22.37	17.03	19.88
	GWO	60.44	Feasible	Feasible	Feasible
	Jaya	Feasible	Feasible	Feasible	Feasible
	SHTS	561.86	423.70	395.88	463.28
	MFO	37.90	31.63	26.81	29.74
	sTLBO	36.89	31.85	26.58	23.71
MUIPP	COA	Feasible	Feasible	Feasible	Feasible
	CSO	7.88	11.25	6.06	-2.49
	GWO	Feasible	Feasible	Feasible	Feasible
	Jaya	Feasible	Feasible	Infeasible	Feasible
	SHTS	4.36	7.32	12.03	4.36
	MFO	Infeasible	Feasible	Feasible	Feasible
	sTLBO	1.63	6.07	6.70	1.53

In the MUIPP model, COA, as well as GWO, have utilized the benefits of the reduced problem dimension of the model using the proposed strategy and are able to determine the feasible *mean*, whereas these techniques were unable to obtain a feasible *mean* using the strategy in the literature. The infeasible *mean* of Jaya in Case 3 and MFO in Case 1 indicates that these techniques are unable to determine a feasible production plan in all the runs of the specified cases. CSO provided a better *mean* using the proposed strategy in all the cases except Case 4, where it determined a better *mean* with the strategy in the literature.

3.5.3. Analysis of the best production plan

The best production plan determined by every technique for the SLPP and MUIPP models with the proposed strategy is analyzed to examine the efficiency of the obtained plan based on the resource utilization and product portfolio.

Resource utilization

From a user perspective, the efficient utilization of resources plays an important role in identifying the quality of a production plan. The details regarding the investment budget and raw materials utilization of the best production plan corresponding to each case of SLPP and MUIPP models are provided in Table 3.9.

In the SLPP model, the investment budget is completely utilized for all the cases except Case 4. As per the case study, the amount of raw material available in Case 1 and Case 3 is less than that of Case 2 and Case 4 (Table 3.2). In the SLPP model, it is evident from the best production plan of Case 1 and Case 4 that both the raw materials are entirely consumed, while such a trend is not observed in Case 2 and Case 3. The maximum residual amount of raw materials is observed in Case 2 of the SLPP model.

Table 3.9 Percentage of resources utilized

	Cases	Investment Budget	Raw material 1	Raw material 2
SLPP	**Case 1**	100.00	100.00	100.00
	Case 2	100.00	66.05	84.72
	Case 3	100.00	100.00	100.00
	Case 4	99.47	97.95	100.00
MUIPP	**Case 1**	97.33	98.87	99.94
	Case 2	99.30	88.26	95.02
	Case 3	99.90	78.96	96.25
	Case 4	99.68	88.26	96.66

In all the MUIPP cases, it is observed that the best production plan does not lead to the complete utilization of the available resources. In all the cases except Case 1 of the MUIPP model, more than 99% of the available investment budget is utilized in implementing the best production

plan. The best production plan obtained in all the cases of both the production planning models utilized the second raw material more than the first one. Hence, it can be inferred that the products manufactured using the second raw material are more profitable.

Product portfolio

For the SLPP model, the contribution of each product to the total profit using the proposed strategy is provided in Fig. 3.11. The legend corresponding to each figure specifies the product number and the process number used to produce it. For example, i_1-j_3 represents the first product produced using the third process. It is observed that the 1st, 5th, 15th, and 21st products are commonly manufactured in all cases, as these could be the most profitable products within the available resources. In all cases, the 21st product is observed to be the most profitable as its profit contribution is the highest compared to other selected products. The investment budget in Case 1 and Case 2 is less than in the other cases. Hence the number of products manufactured in Case 1 and Case 2 is lesser compared to Case 3 and 4. It should be noted that Case 4 has the maximum availability of all the resources, and it is reflected in the best production plan as it selects the maximum number of products to be manufactured.

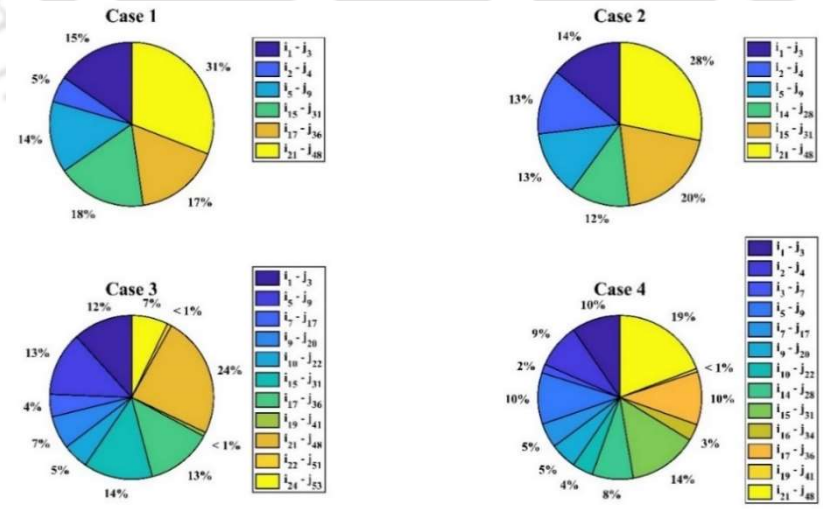


Fig. 3.11 Profit contribution of products in the SLPP model

The profit contribution of each product in all cases of the MUIPP model is provided in Fig. 3.12. The legend specifies the product number, the process number used to produce it, and the

number of units of the selected process to be implemented. For example, i_2-j_4 (2U) represents the first product produced using the third process by implementing two units.

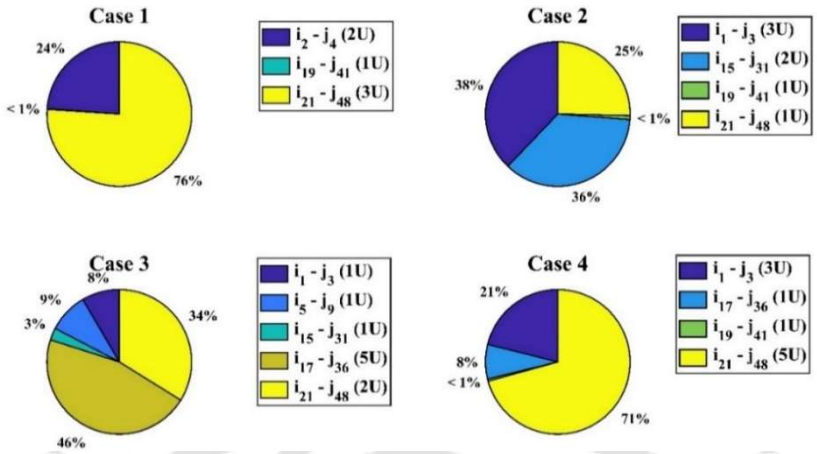


Fig. 3.12 Profit contribution of products in the MUIPP model

On analyzing the profit contribution of the selected products in the MUIPP model, it is found that the 21st product is produced as per the best production plan of all the cases. In all MUIPP cases except in Case 2, multiple units of the 48th process are implemented to produce the 21st product. In Case 2, the 1st product is observed to be more profitable than the 21st product. Due to the restriction on the number of processes, more products have been produced in the SLPP model than in the MUIPP model.

3.6. Conclusion

This study has analyzed two models, namely the SLPP and the MUIPP, to guide production planning in the petrochemical industries. This study solved both models using five recently proposed metaheuristic techniques, where COA is able to determine a better production plan in all the cases of the SLPP model. The shortcomings of the strategy in the literature are identified, and a compact and efficient strategy is proposed. In literature, the problem dimension is primarily a function of the total number of processes. However, in the proposed strategy, the problem dimension varies with the number of products, and the most profitable process is selected to manufacture each product. For a case study involving 24 products and 54

processes, the proposed strategy helped reduce the number of decision variables of both models by approximately 56% compared to the literature. The proposed strategy is also able to reduce the total number of constraints by about 67% for the SLPP model and up to 62% for the MUIPP model. The performance analysis of the proposed strategy using seven metaheuristic techniques demonstrated the determination of better production plans for the SLPP and MUIPP models as compared to the literature. The proposed strategy helped sTLBO provide up to a 9% improvement in profit. Out of seven metaheuristic techniques, three of the techniques, namely COA, Jaya, and MFO, were able to obtain a feasible solution in all cases, while they were unable to determine even one feasible solution in the MUIPP model with the strategy in the literature.

Chapter 4

A metaheuristic-based efficient strategy for multi-unit production planning using only continuous variables and unique process constraints

This chapter carefully investigated the drawbacks of the strategy available in the literature for solving the multi-unit production planning problem (MUCPP) with unique process constraints, which employs only continuous decision variables to model the problem. The critical analysis of the strategy used by Chauhan and Kotecha (2018) identified a few limitations, and a new solution strategy is proposed in this chapter to overcome them. The proposed strategy inherently satisfies the requirement of implementing multiple units at *l*, *m*, and *h* production capacities as well as between *l*-*m* and the *m*-*h* production capacity levels, as in the multi-unit production planning model (MUIPP) studied in Chapter 3. Moreover, the proposed strategy does not require integer variables and thus reduces the number of decision variables.

Four different case studies are solved using eight metaheuristic techniques to analyze the effectiveness of the proposed strategy. The structure of this chapter is as follows: Section 4.1 describes the multi-unit production planning model, along with its strategy used in the literature. Section 4.2 discusses the proposed strategy for the multi-unit production planning problem with continuous variables. Section 4.3 provides the experimental settings employed for the performance evaluation of the proposed strategy, followed by the results and discussions in Section 4.4. Section 4.5 concludes this chapter by consolidating the significant findings of this work.

4.1. Strategy in literature for multi-unit production planning with continuous variables

The production planning problem explained in Section 3.1 is extended to accommodate the facility of implementing multiple units of each process at its *l*-*m* and the *m*-*h* production

Kommadath, R., D. Maharana, and P. Kotecha, *A metaheuristic-based efficient strategy for multi-unit production planning with unique process constraints.* Applied Soft Computing, 2023. **134**: p. 109871. DOI: https://doi.org/10.1016/j.asoc.2022.109871

capacity level in Chauhan and Kotecha (2018). In this, the maximum number of processing units that can be implemented at *l-m* and *m-h* production capacities of a process is limited by the availability of the resources such as investment budget and raw materials and the investment cost and amount of raw material required at low level production capacity. The maximum number of units of the j^{th} process that can be implemented in its *l-m* capacity range and *m-h* capacity range is determined using Eq. (4.1) and Eq. (4.2), respectively.

$$N_{lm}^{j} = \min\left(\left\lfloor B/V_{lj} \right\rfloor, \min_{k\in\{1,2,..,K\}}\left(\left\lfloor \frac{R_k}{b_j^k l_j} \right\rfloor\right)\right) \tag{4.1}$$

$$N_{mh}^{j} = \min\left(\left\lfloor B/V_{mj} \right\rfloor, \min_{k\in\{1,2,..,K\}}\left(\left\lfloor \frac{R_k}{b_j^k m_j} \right\rfloor\right)\right) \tag{4.2}$$

It should be noted that the maximum number of units corresponding to the processes is different as the lower production capacity, the investment cost, and the raw material requirements of each process are different. In the strategy followed in the literature, there would be a decision variable corresponding to each unit of the process, and its value represents the amount of product to be manufactured using that particular process unit. Hence, corresponding to the j^{th} process, there would be N_{lm}^{j} decision variables indicating the production quantity of the product in the *l-m* production capacity and N_{mh}^{j} decision variables in the *m-h* production capacity. The total number of decision variables (D) as per the strategy available in the literature is calculated using Eq. (4.3).

$$D = \sum_{j=1}^{J} N_{lm}^{j} + N_{mh}^{j} \tag{4.3}$$

The decisions to be made for the production planning problem are (i) the products to be produced, (ii) the choice of process to manufacture the selected product, (iii) the operating capacity of the process, and (iv) the number of processing units to be implemented at the

operating capacity for determining an optimal production plan with maximum profit by satisfying all the associated constraints. The solution structure based on the strategy in the literature is provided in Fig. 4.1, where, corresponding to each process, there would be $\left(N_{lm}^{j} + N_{mh}^{j}\right)$ decision variables representing the production capacity of each unit of the process. In Fig. 4.1, the decision variables corresponding to the 1st and j^{th} processes for *l-m* and *m-h* production capacities are detailed, and a similar structure is used for the rest of the processes.

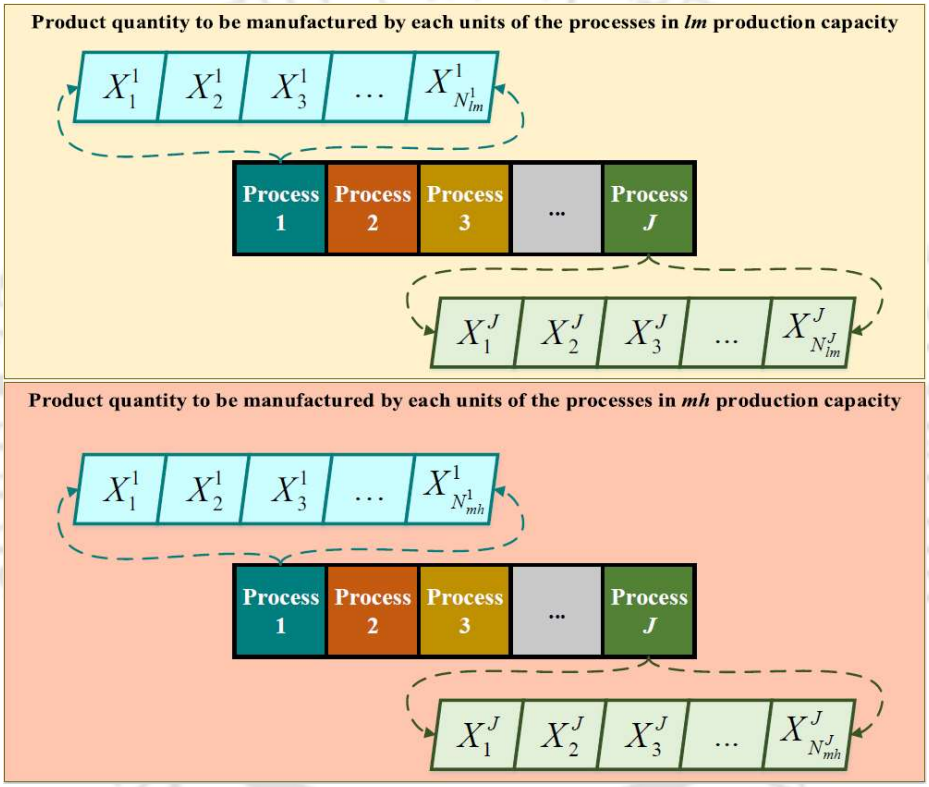


Fig. 4.1. Solution structure of the strategy in the literature

Bounds of the decision variables: The lower bound is equal to zero for all the *D* variables. The decision variable with a value of zero infers no production of the respective process unit. The upper bound of the variables representing the production quantity of a specific unit of the j^{th} process in its *l-m* and *m-h* production capacity is equal to the medium-level and high-level production capacity, respectively. The bounds of the decision variables are given in Eq. (4.4).

$$\left.\begin{aligned} 0 \le X_{u_{lm}}^{j} \le m_j, \quad \forall u_{lm} \in \left\{1,2,...,N_{lm}^{j}\right\} \\ 0 \le X_{u_{mh}}^{j} \le h_j, \quad \forall u_{mh} \in \left\{1,2,...,N_{mh}^{j}\right\} \end{aligned}\right\}, \quad \forall j \in \left\{1,2,...,J\right\} \tag{4.4}$$

Determination of production and investment costs: The determination of production and investment costs of a processing unit of a process depends on the production region in which it has to be implemented. The production cost of the j^{th} process having any production capacity in the *l-m* and *m-h* production capacity range is determined using Eq. (4.5) and Eq. (4.6), respectively. Similarly, the investment cost of any unit of the j^{th} process implemented in *l-m* and *m-h* production region is determined using Eq. (4.7) and Eq. (4.8), respectively.

$$C_j^{u_{lm}} = \begin{cases} C_{lj} + \dfrac{\left(X_{u_{lm}}^{j} - l_j\right)\left(C_{mj} - C_{lj}\right)}{\left(m_j - l_j\right)} & if\ l_j \le X_{u_{lm}}^{j} \le m_j \\ 0 & else\ 0 \le X_j < l_j \end{cases}, \tag{4.5}$$
$$\forall u_{lm} \in \left\{1,2,...,N_{lm}^{j}\right\}, \qquad \forall j \in \left\{1,2,...,J\right\}$$

$$C_j^{u_{mh}} = \begin{cases} C_{mj} + \dfrac{\left(X_{u_{mh}}^{j} - m_j\right)\left(C_{hj} - C_{mj}\right)}{\left(h_j - m_j\right)} & if\ m_j \le X_{u_{mh}}^{j} \le h_j \\ 0 & else\ 0 \le X_j < m_j \end{cases}, \tag{4.6}$$
$$\forall u_{mh} \in \left\{1,2,...,N_{mh}^{j}\right\}, \qquad \forall j \in \left\{1,2,...,J\right\}$$

$$V_j^{u_{lm}} = \begin{cases} V_{lj} + \dfrac{\left(X_{u_{lm}}^{j} - l_j\right)\left(V_{mj} - V_{lj}\right)}{\left(m_j - l_j\right)} & if\ l_j \le X_{u_{lm}}^{j} \le m_j \\ 0 & else\ 0 \le X_j < l_j \end{cases}, \tag{4.7}$$
$$\forall u_{lm} \in \left\{1,2,...,N_{lm}^{j}\right\}, \qquad \forall j \in \left\{1,2,...,J\right\}$$

$$V_j^{u_{mh}} = \begin{cases} V_{mj} + \dfrac{\left(X_{u_{mh}}^{j} - m_j\right)\left(V_{hj} - V_{mj}\right)}{\left(h_j - m_j\right)} & if\ m_j \le X_{u_{mh}}^{j} \le h_j \\ 0 & else\ 0 \le X_j < m_j \end{cases}, \tag{4.8}$$
$$\forall u_{mh} \in \left\{1,2,...,N_{mh}^{j}\right\}, \qquad \forall j \in \left\{1,2,...,J\right\}$$

In Eq. (4.5) – Eq. (4.8), C_{lj}, C_{mj}, and C_{hj} represent the production cost of the j^{th} process at its l, m, and h production capacities, respectively. Similarly, V_{lj}, V_{mj}, and V_{hj} indicate the investment costs required to implement a processing unit of j^{th} process in l, m, and h production capacity, respectively.

Investment budget constraint: The total investment cost of a particular production plan is calculated using Eq. (4.9). The total investment cost required to implement a potential production plan must be less than or equal to the total investment budget. Any potential solution that violates the investment budget constraint is penalized based on the magnitude of the violation, and the corresponding penalty is determined using Eq. (4.10).

$$V^{Total} = \sum_{j=1}^{J} \left(\sum_{u_{lm}=1}^{N_{lm}^{j}} V_{j}^{u_{lm}} + \sum_{u_{mh}=1}^{N_{mh}^{j}} V_{j}^{u_{mh}} \right) \tag{4.9}$$

$$P^{Investment} = \begin{cases} \left(B - V^{Total} \right)^2 & if\ V^{Total} > B \\ 0 & else \end{cases} \tag{4.10}$$

Raw material constraints: The total k^{th} raw material required for implementing a production plan is calculated using Eq. (4.11).

$$r^{k} = \sum_{j=1}^{J} b_{j}^{k} \left(\sum_{u_{lm}=1}^{N_{lm}^{j}} X_{u_{lm}}^{j} + \sum_{u_{mh}=1}^{N_{mh}^{j}} X_{u_{mh}}^{j} \right) \quad \forall k \in \{1,2,...,K\} \tag{4.11}$$

The total amount of raw materials used to manufacture all the products using the selected processes in a potential production plan should be less than or equal to the total availability of the raw materials. The violation of these constraints is penalized, and the penalty is determined as per Eq. (4.12).

$$P_k^{Raw} = \begin{cases} \left(R^k - r^k\right)^2 & if\ r^k > R^k \\ 0 & else \end{cases}, \quad \forall k \in \{1,2,...,K\}$$
$$P^{Raw} = \sum_{k=1}^{K} P_k^{Raw} \tag{4.12}$$

Domain hole constraints: Any decision variables that violate the domain hole constraints are replaced with zero, which is mathematically represented in Eq. (4.13).

$$X_{u_{lm}}^j = \begin{cases} X_{u_{lm}}^j & if \quad l_j \le X_{u_{lm}}^j \le m_j \\ 0 & else \end{cases}, \forall u_{lm} \in \{1,2,...,N_{lm}^j\}, \forall j \in \{1,2,...,J\}$$
$$X_{u_{mh}}^j = \begin{cases} X_{u_{mh}}^j & if \quad m_j \le X_{u_{mh}}^j \le h_j \\ 0 & else \end{cases}, \forall u_{mh} \in \{1,2,...,N_{mh}^j\}, \forall j \in \{1,2,...,J\} \tag{4.13}$$

Unique process constraints: These constraints allow the usage of only one process for producing a particular product. Hence, all the processing units producing a product in both valid regions must use the same process. The total quantity of a product manufactured by multiple units in *l-m* and *m-h* production capacities of processes is calculated as Eq. (4.14).

$$X_j^{Total} = \sum_{u_{lm}=1}^{N_{lm}^j} X_{u_{lm}}^j + \sum_{u_{mh}=1}^{N_{mh}^j} X_{u_{mh}}^j \tag{4.14}$$

The number of processes that have been utilized to manufacture each product is identified using Eq. (4.15), where J_i indicates the set of processes allowed to produce the i^{th} product. The variable y_j corresponding to j^{th} process possesses a value of one when any unit of j^{th} process manufactures the product; otherwise, it would be zero. The number of processes used to produce the i^{th} is represented in Eq. (4.15). The penalty corresponding to the violation of unique process constraints is determined using Eq. (4.16).

$$n_i = \sum_{j \in J_i} y_j, \quad \forall i \in \{1,2,...,I\}$$
$$where \quad y_j = \begin{cases} 0 & if \quad X_j^{Total} = 0 \\ 1 & if \quad X_j^{Total} > 0 \end{cases}, \quad \forall j \in \{1,2,...,J\} \tag{4.15}$$

$$P^{Unique} = \sum_{i=1}^{I} P_i^{Unique}$$
$$where\ P_i^{Unique} = \begin{cases} 1000^{n_i} & if\ n_i > 1 \\ 0 & else \end{cases}, \quad \forall i \in \{1,2,...,I\} \tag{4.16}$$

Objective function: The selling price of the product manufactured by the j^{th} process on both *l-m* and *m-h* production capacity ranges is determined using Eq. (4.17), and the corresponding total production cost is determined using Eq. (4.18).

$$S_j = E_j X_j^{Total}, \quad \forall j \in \{1,2,...,J\} \tag{4.17}$$

$$C_j = \sum_{u_{lm}=1}^{N_{u_{lm}}^j} C_j^{u_{lm}} + \sum_{u_{mh}=1}^{N_{u_{mh}}^j} C_j^{u_{mh}}, \quad \forall j \in \{1,2,...,J\} \tag{4.18}$$

In MUCPP, the profit corresponding to a potential production plan, irrespective of its feasibility, is determined using Eq. (4.19).

$$Profit = \sum_{j=1}^{J} \left(S_j - C_j \right) \tag{4.19}$$

Fitness function: The fitness value helps to prefer a feasible solution to an infeasible one and is determined using Eq. (4.20). Here, λ is a penalty factor.

$$Maximize\ f = \begin{cases} Profit & if\ P^{Total} = 0 \\ Profit - \lambda P^{Total} & else \end{cases}$$
$$where\ P^{Total} = P^{Investment} + P^{Raw} + P^{Unique} \tag{4.20}$$

4.2. Proposed strategy for multi-unit production planning with continuous variables

The solution strategy followed by Chauhan and Kotecha (2018) uses multiple processes per product, which adversely affects the satisfaction of unique process constraints. For example, consider a product *i* that can be produced by two processes, j_1 and j_2. Let process j_1 implement up to two units in *l-m* production capacity and three units in *m-h* production capacity. Similarly, process j_2 can implement a maximum of three units in *l-m* production capacity and one unit in

m-h production capacity. Based on the strategy in the literature, there would be nine decision variables (2+3+3+1). These variables are free to possess any value within their bounds. However, to satisfy the unique process constraints, the product should be manufactured either by using the processing units of j_1 or by using j_2 in both production levels.

The optimization technique searches for a solution that manufactures beneficial products using their most profitable process. As per the strategy in the literature, the decision variables corresponding to the processing units of the less profitable processes of the products must be zero. Similarly, it is also not mandatory that all the production units of the profitable process have to be utilized to manufacture the selected product. In that instance, some of the decision variables indicating the production quantity of the selected process would be equal to zero. In summary, most of the decision variables in the production plan would have a value of zero while it satisfies the unique process constraints. In order to satisfy all the unique process constraints, the strategy in the literature relies on the penalty approach, which might cause difficulty for the metaheuristic technique to identify a feasible solution.

The systematic analysis of the strategy employed by Chauhan and Kotecha (2018) helped us identify an efficient modeling strategy for the multi-unit production planning problem. Unlike the strategy in the literature, where the decision variables represent the production quantity of each processing unit of all the processes, the proposed strategy considers only the production quantity of each product at the *l-m* and *m-h* production capacities of a process as the decision variables. The solution structure is categorized as two sections representing the quantity of product produced using the multiple units of the selected process in its *l-m* and *m-h* production capacity ranges. The pictorial representation of the solution structure followed in the proposed strategy is provided in Fig. 4.2.

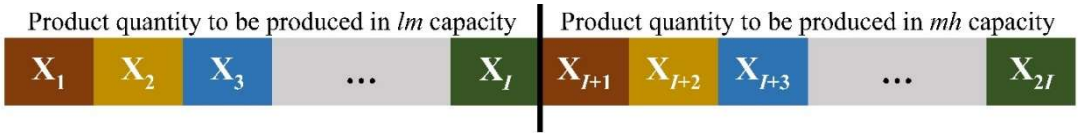


Fig. 4.2 Solution structure followed in the proposed strategy

Bounds of the decision variables: The lower bound of all the decision variables is zero, where a zero value indicates the corresponding product is not being manufactured. The maximum number of units of the j^{th} process that can be implemented in the *l-m* and *m-h* production capacities is determined using Eq. (4.1) and Eq. (4.2), respectively. The maximum product quantity produced by a process in its *l-m* and *m-h* capacities is determined using Eq. (4.21).

$$\begin{aligned} Xmax_{lm}^{j} &= N_{lm}^{j} \max\left(l^{j}, m^{j}\right) \\ Xmax_{mh}^{j} &= N_{mh}^{j} \max\left(m^{j}, h^{j}\right) \end{aligned} \tag{4.21}$$

The profitable process for a product is unknown in advance, and it is necessary to identify the profitable process based on the production quantity specified in a potential production plan. Hence, identifying the maximum quantity of the product that can be manufactured using any of the potential processes is crucial to determine the upper bound of the decision variables. The upper bound for each decision variable can be determined using Eq. (4.22), where J_i is the set of processes allowed to produce the i^{th} product.

$$ub_i = \begin{cases} \max\limits_{j \in J_i}\left(Xmax_{lm}^{j}\right) & \forall i \in \{1,2,...,I\} \\ \max\limits_{j \in J_i}\left(Xmax_{mh}^{j}\right) & \forall i \in \{I+1, I+2,..., 2I\} \end{cases} \tag{4.22}$$

The value of the decision variable representing the product quantity need not necessarily satisfy the upper capacity of all the potential processes. This is mainly due to the difference in the maximum production capacity of all the processes used to produce a particular product. Suppose the decision variable violates the upper capacity of any potential processes; the profit corresponding to that process is determined based on its maximum production capacity. The bound correction of decision variables with respect to the *l-m* production capacity of each

potential process is performed as in Eq. (4.23). Similarly, the bound correction of decision variables in *m-h* production capacity of the processes is carried out as provided in Eq. (4.24). Here, δ_j^{lm} and δ_j^{mh} represent the production quantities using the j^{th} potential process in the *l-m* and *m-h* production capacities, respectively.

$$\gamma_j^{lm} = \begin{cases} X_i & if \;\; l^j \le X_i \le Xmax_{lm}^j \\ Xmax_{lm}^j & else \;\; X_i > Xmax_{lm}^j \end{cases} \qquad \forall j \in J_i \tag{4.23}$$

$$\gamma_j^{mh} = \begin{cases} X_i & if \;\; m^j \le X_i \le Xmax_{mh}^j \\ Xmax_{mh}^j & else \;\; X_i > Xmax_{mh}^j \end{cases} \qquad \forall j \in J_i \tag{4.24}$$

The decision variables in the proposed strategy provide the total amount of product to be manufactured in the *l-m* and *m-h* production capacities of the profitable process. The selection of the profitable process and the number of processing units in its *l-m* and *m-h* capacities have to be determined for each potential solution. Hence, after the bound correction of the decision variable with respect to the j^{th} process $\left(j \in J_i\right)$, the number of units $\left(z_j\right)$ and per-unit capacity $\left(\omega_j\right)$ of the j^{th} process have to be determined to identify the profitable process among the set of processes $\left(J_i\right)$ available to produce the product. The total production cost required to implement z_j units of the j^{th} process with a production capacity of ω_j^{lm} in *l-m* and ω_j^{mh} in *m-h* production capacities can be determined as per Eq. (4.25) and Eq. (4.26), respectively.

$$C_j^{lm} = z_j^{lm} c_j^{lm}$$
$$where \; c_j^{lm} = \begin{cases} C_{lj} + \dfrac{\left(\omega_j^{lm} - l_j\right)\left(C_{mj} - C_{lj}\right)}{\left(m_j - l_j\right)} & if \; l_j \le \omega_j^{lm} \le m_j, \quad \forall j \in J_i \\ 0 & else \; 0 \le X_i < l_j \end{cases} \tag{4.25}$$

$$C_j^{mh} = z_j^{mh} c_j^{mh}$$
$$where \; c_j^{mh} = \begin{cases} C_{mj} + \dfrac{\left(\omega_j^{mh} - m_j\right)\left(C_{hj} - C_{mj}\right)}{\left(h_j - m_j\right)} & if \; m_j \le \omega_j^{mh} \le h_j, \quad \forall j \in J_i \\ 0 & else \; 0 \le X_i < l_j \end{cases} \tag{4.26}$$

The total amount of product produced by the j^{th} processes can be determined using Eq. (4.27), and the revenue from the product is calculated using Eq. (4.28) where the E_j is the unit selling price of the product produced from the j^{th} process. The profit corresponding to each process can be determined using Eq. (4.29).

$$\rho_j = z_j^{lm}\omega_j^{lm} + z_j^{mh}\omega_j^{mh}, \quad \forall j \in J_i, i \in \{1,2,...,I\} \tag{4.27}$$

$$S_j = E_j \rho_j, \quad \forall j \in J_i, i \in \{1,2,...,I\} \tag{4.28}$$

$$Profit_j = S_j - C_j \quad \forall j \in J_i, i \in \{1,2,...,I\} \tag{4.29}$$

The process that provides maximum profit among the potential processes is identified as the profitable process (α_i) for the i^{th} product. The investment cost required to implement the multiple units of the selected profitable process at *l-m* and *m-h* production capacity is calculated as Eq. (4.30) and Eq. (4.31), respectively.

$$V_i^{lm} = v_i^{lm} z_{\alpha_i}^{lm}$$
$$where\ v_i^{lm} = \begin{cases} V_{l\alpha_i} + \dfrac{\left(x_{lm}^i - l_{\alpha_i}\right)\left(V_{m\alpha_i} - V_{l\alpha_i}\right)}{\left(m_{\alpha_i} - l_{\alpha_i}\right)} & if\ l_{\alpha_i} \le \omega_{\alpha_i} \le m_{\alpha_i}, \forall \alpha_i \in J_i, i \in \{1,2,...,I\} \\ 0 & else\ 0 \le X_i < l_{\alpha_i} \end{cases} \tag{4.30}$$

$$V_{I+i} = v_i^{mh} z_{\alpha_i}^{mh}$$
$$where\ v_i^{mh} = \begin{cases} V_{m\alpha_i} + \dfrac{\left(x_{mh}^i - l_{\alpha_i}\right)\left(V_{h\alpha_i} - V_{m\alpha_i}\right)}{\left(h_{\alpha_i} - m_{\alpha_i}\right)} & if\ m_{\alpha_i} \le \omega_{\alpha_i} \le h_{\alpha_i}, \forall \alpha_i \in J_i, i \in \{1,2,...,I\} \\ 0 & else\ 0 \le X_i < m_{\alpha_i} \end{cases} \tag{4.31}$$

Here, $z_{\alpha_i}^{lm}$ and $z_{\alpha_i}^{mh}$ indicate the number of units of the selected process for manufacturing the i^{th} product in its *l-m* and *m-h* production capacity ranges. The total investment cost corresponding to the implementation of multiple units of the profitable process for manufacturing all the products is determined using Eq. (4.32). The penalty due to the violation of the investment budget constraint $\left(P^{Investment}\right)$ can be calculated using Eq. (4.10).

$$V^{Total} = \sum_{i=1}^{I} \left(V_i + V_{I+i} \right) \tag{4.32}$$

The amount of k^{th} raw material required to manufacture the i^{th} product using its profitable process $\left(\alpha_i \right)$ is calculated using Eq. (4.33).

$$r_i^k = b_{\alpha_i}^k \left(X_i + X_{I+i} \right), \quad \forall i \in \{1,2,...,I\} \tag{4.33}$$

The total requirement of k^{th} raw material for manufacturing all the products specified in a potential schedule is determined as Eq. (4.34). The penalty assigned for violating the raw material constraints $\left(P^{Raw} \right)$ can be calculated using Eq. (4.12).

$$r^k = \sum_{i=1}^{I} r_i^k, \forall k \in \{1,2,...,K\} \tag{4.34}$$

The fitness value of a potential schedule can be determined using Eq. (4.35).

$$f = \sum_{i=1}^{I} Profit_i + \lambda \left(P^{Investment} + P^{Raw} \right) \tag{4.35}$$

The overall optimization formulation of the MUIPP model is provided in Eq. (4.36). It should be noted that the formulation provided in Eq. (4.36) should be used in conjunction with the proposed strategy discussed above. The binary variable Z_i is used to demonstrate the unique process constraints and is not necessary for the proposed strategy.

$$\begin{aligned}
&\textit{Maximize Profit} \\
&\textit{s.t.}, \\
&\sum_{i=1}^{I} \left(V_i + V_{I+i} \right) \le B \\
&\sum_{i=1}^{I} b_{\alpha_i}^k \left(X_i + X_{I+i} \right) \le R^k \quad \forall k \in \{1,2,...,K\} \\
&\left(X_i = 0 \right) \vee l^{\alpha_i} \le X_i \le Xmax_{lm}^{\alpha_i} \quad \forall i \in \{1,2,...,I\} \\
&\left(X_{I+i} = 0 \right) \vee m^{\alpha_i} \le X_i \le Xmax_{mh}^{\alpha_i} \quad \forall i \in \{1,2,...,I\} \\
&\sum_{j \in J_i} Z_j \le 1 \quad \forall i \in \{1,2,...,I\} \\
&0 \le X_i \le ub_i, \quad \forall i \in \{1,2,..,2I\}
\end{aligned} \tag{4.36}$$

The proposed strategy guarantees a production plan satisfying the domain constraints and the unique process constraints. Hence, when there is a violation of investment budget constraints or raw material constraints, the fitness value of a potential plan is penalized. The procedure for determining the fitness function value of a potential production plan is provided in Fig. 4.3.

Input: $X, l, m, h, C_l, C_m, C_h, V_l, V_m, V_h, N_{lm}, N_{mh}, Xmax_{lm}, Xmax_{mh}$

Output: X_c *and* f

For i = 1:I

Identify the set of processes (J_i) that can produce the i^{th} product

Determine the profitable process using the function *Determination of profitable process*: $\left[\alpha_i, z^{lm}_{\alpha_i}, z^{mh}_{\alpha_i}, x^i_{lm}, x^i_{mh}, X_i, X_{I+i}, Profit_i\right]$ =

Determination of profitable process $\left(i, J_i, l, m, h, X_i, X_{I+i}, N^i_{lm}, N^i_{mh}, Xmax^i_{lm}, Xmax^i_{mh}\right)$

Determine the investment cost (V_i *and* V_{I+i}) for implementing the process units of α_i to manufacture x^i_{lm} and x^i_{mh} quantity of the i^{th} product using Eq. (3.30) and Eq. (3.31), respectively and the total investment cost using Eq. (3.32).

Determine the amount of k^{th} raw material using Eq. (3.33)

End For

Calculate penalty for violating investment budget constraint $\left(P^{Investment}\right)$ using Eq. (3.10)

Calculate the penalty for violating the raw material constraint $\left(P^{Raw}\right)$ using Eq. (3.12)

Calculate the fitness value using Eq. (3.35)

Assign $X_c = X_i$ and **return** X_c and f

Fig. 4.3 Pseudocode for the determination of fitness value

In the proposed strategy, the decision variable does not directly provide information regarding the number of units, the choice of process to produce the product, or the production capacity at which the multiple units of the process have to be implemented. For a given production quantity of a product, the most profitable process among the available processes is selected as per Fig. 4.4. Instead of implementing multiple units having different production capacities, as per the strategy followed in the literature, this strategy allows each process to implement a maximum number of units at a specific production capacity. Before determining a profitable process,

bound violation of the product quantity specified in the potential solution must be examined. The process that has provided maximum profit by producing the quantity of the product specified in the *l-m* and *m-h* production capacities is selected as the process for producing the product.

Input: $i, J_i, l, m, h, X_i, X_{I+i}, N_{lm}, N_{mh}, Xmax_{lm}, Xmax_{mh}, E_\phi$

Output: $\alpha_i, z_{\alpha_i}^{lm}, z_{\alpha_i}^{mh}, x_{lm}^i, x_{mh}^i, X_i, X_{I+i}, Profit_i$

Set the number of processes to produce the i^{th} product as $N = |J_i|$

Set $Profit = -\infty$ as the initial value of profit obtained from the i^{th} product

For n = 1 to N

Set $\phi = J_i^n$

Assign $L = lm,\ \rho_n = X_i, \delta = l^\phi, \Delta = m^\phi$

Determine the number of units, production quantity of individual units, and the updated ρ_n using the function *Profit calculation*:

$[Profit_n, \rho_n, z_n, \omega_n] =$ ***Profit calculation***$(L, \rho_n, \delta, \Delta, N_L^\phi, Xmax_L^\phi, E_\phi)$

Assign $L = mh,\ \rho_{N+n} = X_{I+i}, \delta = m^\phi, \Delta = h^\phi$

Determine the number of units, production quantity of individual units, and the updated ρ_{N+n} using the function *Profit calculation*:

$[Profit_{N+n}, \rho_{N+n}, z_{N+n}, \omega_{N+n}] =$ ***Profit calculation***$(L, \phi, \rho_{N+n}, \delta, \Delta, N_L^\phi, Xmax_L^\phi, E_\phi)$

Determine total profit with respect to the n^{th} process as $Profit_n^{Total} = Profit_n + Profit_{N+n}$

End For

Set $Profit_i = \max_{n \in \{1,2,..,N\}} \left(Profit_n^{Total}\right)$ and the corresponding process in J_i as n_s,

Assign the profitable process as $\alpha_i = J_i^{n_s}$

Assign $X_i = \rho_{n_s}$ and $X_{I+i} = \rho_{N+n_s}$

Assign $z_{\alpha_i}^{lm} = z_{n_s}$, $z_{\alpha_i}^{mh} = z_{N+n_s}$, $x_{lm}^i = \omega_{n_s}$, *and* $x_{mh}^i = \omega_{N+n_s}$

Fig. 4.4 Pseudocode for the determination of profitable process

After the bound correction, the number of units and the per-unit capacity of the process in both *l-m* and *m-h* production capacities has to be determined. Details regarding the determination of

multiple units, their production capacity, and the profit calculation are given in Fig. 4.5. In the proposed strategy, multiple units are implemented at the same production capacity, while in

Input: $L, \phi, \rho, \delta, \Delta, N_L^{\phi}, Xmax_L^{\phi}, E_{\phi}$

Output: $Profit, \rho, z, \omega$

If $(0 < \rho) \wedge (\rho < \delta)$

Set $\rho = 0$, $Profit = 0$

Else

If $(\rho > Xmax_L^{\phi})$

Set $\rho = Xmax_L^{\phi}$

End If

If $(\rho \% \Delta) == 0$

Set the number of units of process ϕ in L production capacity as $z = \rho / \Delta$ and the corresponding per-unit production quantity as $\omega = \rho / z$

Else

Set $z = N_L^{\phi} - 1$

While $z \geq 0 \wedge \rho > 0$

If $[\rho \geq (z+1)\delta \wedge \rho \leq (z+1)\Delta]$

Set the number of units of the process ϕ in L production capacity $z = z + 1$ and the per-unit production quantity as $\omega = \rho / (z+1)$

Break While

Else if $[\rho \geq z\Delta \wedge \rho \leq (z+1)\delta]$

Set the number of units of the process ϕ in *l-m* production capacity as z, the per-unit production quantity as $\omega = \Delta$ and $\rho = z\omega$

Break While

End If

$z = z - 1$

End While

End If

End If

Determine the total production cost (C) for producing ρ the amount of the product with z units of the process ϕ using Eq. (3.25) or Eq. (3.26) and the selling price $S = E_{\phi}\rho$

Determine the profit, $Profit = S - C$

Fig. 4.5 Pseudocode for the profit calculation

the strategy in the literature, the value of the decision variable specifies the production capacity at which a unit is implemented. The symbol '%' in Fig. 4.5 indicates the remainder after the division operation (mod).

4.3. Experimental Settings

The same four case studies of Chapter 3 are utilized to analyze the efficacy of the proposed strategy of the MUCPP model. In the proposed strategy, the upper bound of the decision variables depends on the investment budget and is provided in Fig. 4.6. As the investment budget for Case 3 and Case 4 are higher than Case 1 and Case 2, the upper bound of the decision variables of former cases is more compared to the latter cases.

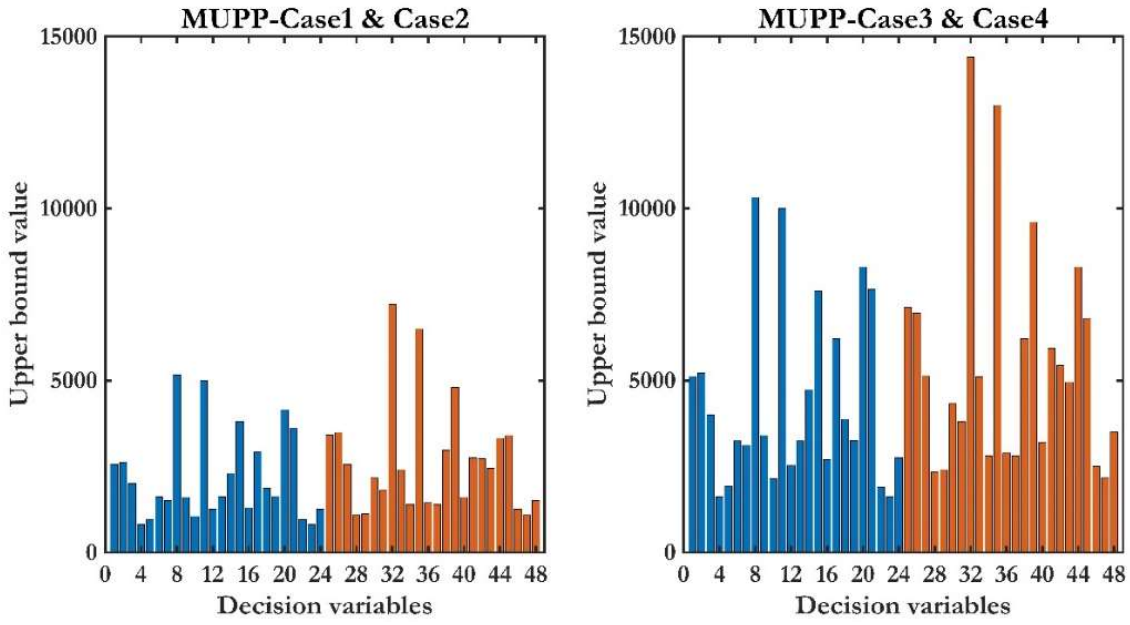


Fig. 4.6 Upper bound of the decision variables in the proposed strategy

The objective function and all metaheuristic techniques are implemented in Matlab 2020a on a desktop PC with Intel i7 4.2 GHz processor and 16GB RAM. Each case study is solved with eight metaheuristic techniques by using the proposed strategy and the strategy in the literature. On considering the stochastic nature of the techniques, each case of the MUCPP problem is

executed 26 times. The population size and the maximum number of function evaluations are set to 100 and 60100, respectively, for all techniques. The average computational time required by each technique to solve all cases of the MUCPP model in one run is approximately 40 seconds.

Four of the eight metaheuristic techniques have been used to solve the strategy in Chauhan and Kotecha (2018), whereas the other four are recently proposed techniques. The eight metaheuristic techniques include Multi-Population based Ensemble of mutation strategies Differential Evolution (MPEDE) (Wu et al., 2016), Sanitized Teaching Learning Based Optimization (sTLBO), Artificial Bee Colony (ABC) (Karaboga & Basturk, 2007), Dynamic neighborhood learning particle swarm optimizer (DNLPSO) (Nasir et al., 2012), Sine Cosine Algorithm (SCA)(Mirjalili, 2016), Symbiotic Organisms Search (SOS) (Cheng & Prayogo, 2014), Single Phase Multi-Group Teaching Learning Based Optimization with Lévy Flight (SPMGLFTLO) and Harris Hawk Optimization (HHO) (Heidari et al., 2019). A brief description of the metaheuristic techniques is given in Appendix A. The sensitivity of algorithm-specific parameters of ABC, DNLPSO, MPEDE, SCA, and SPMGLFTLO are analyzed using multiple parameter values, and a detailed study is provided in Appendix C. As techniques such as HHO, SOS, and sTLBO do not possess any tunable parameters, these are not considered in the study regarding parameter tuning. The value taken for algorithm-specific parameters of each technique is given in Table 4.1.

Table 4.1 Settings for the algorithm-specific parameters

Technique	Parameter	Values
ABC	Limit (l)	$l = 0.1 n_e D$
DNLPSO	Refreshing gap (m) and regrouping period (g)	$(m, g) = (3, 5)$
MPEDE	Ratio of indicator population to the whole population (λ_1) and generation gap (n_g).	$(\lambda_1, n_g) = (0.2, 20)$
SCA	Constant (a)	$a = 2$
SPMGLFTLO	Number of groups (N_g)	$N_g = 25$

In Chauhan and Kotecha (2018), authors have also solved the four cases with unique process constraints by incorporating a feasible solution in the initial population of all metaheuristic techniques. Hence, all the cases of MUCPP are solved by incorporating a feasible solution in the initial population to analyze the efficacy of the proposed strategy. This initial solution is the optimal solution reported in Kadambur and Kotecha (2016) for multi-level production planning, which allows a process to implement only one unit in its *l-m* and *m-h* production capacities.

4.4. Results and discussion

The efficacy of the proposed strategy in solving the MUCPP problem using metaheuristic techniques is demonstrated in this section using various studies that include (i) a comparison of problem dimension based on the proposed strategy and the strategy available in the literature, (ii) the statistical analysis of the results in solving the case studies using both strategies with and without an initial feasible solution, and (iii) the analysis of the best production plan.

4.4.1. Benefits of proposed strategies in problem size

A comparison of problem dimensions based on the proposed strategy and the strategy employed in Chauhan and Kotecha (2018) for MUCPP problems is provided in Table 4.2. According to the strategy followed in the literature, the number of decision variables depends on the investment budget. The investment budget considered in Case 1 and Case 2 is less than in Case 3 and Case 4. Therefore, the number of decision variables in Case 1 and Case 2 is lesser than in the other two cases for the MUCPP problem. However, as per the proposed strategy, the number of decision variables is independent of the investment budget, due to which an equal number of decision variables are present in all cases. It should be noted that according to the strategy used in the literature, a larger budget results in a greater number of decision variables, which is not the case with the proposed strategy. As per the strategy in the literature,

the number of decision variables in Case 1 and Case 2 is about 27 times more than the proposed strategy, while it is approximately 55 times more in Case 3 and Case 4. In both strategies, the number of constraints on investment budget and raw materials remains the same. The number of domain hole constraints in the strategy followed in the literature is equal to the number of decision variables, whereas, in the proposed strategy, the domain hole constraints are applicable for each process at their *l-m* and *m-h* production capacity. Hence, the number of domain constraints would be twice the number of processes.

Table 4.2 Comparison of problem dimension of strategy in the literature and proposed strategy

Cases	**Decision variables**		**Domain hole constraints**		**Unique process constraints**	
	Literature	Proposed	Literature	Proposed	Literature	Proposed
Cases 1 & 2	1287	48	1287	108	24	0
Cases 3 & 4	2624	48	2624	108	24	0

The number of unique process constraints as per the strategy in the literature is equal to the number of products, while the solution structure used in the proposed strategy inherently ensures the satisfaction of the unique process constraints. The reduction in the number of decision variables, domain hole, and unique process constraints led to an overall decrease in the problem dimension of the proposed strategy compared to the strategy in the literature.

4.4.2. Statistical analysis of the results

The statistical study of both strategies is analyzed by solving the case studies with and without providing an initial feasible solution (IFS) to the metaheuristic techniques. The best value corresponding to each statistic of a technique is highlighted using the bold-faced font in the statistical table. The statistical table constitutes the results corresponding to the *best* (B), *worst* (W), *mean* (M), *median* (Md), and *standard deviation* (SD) of the results obtained in all runs using each algorithm with both strategies.

Without the initial feasible solution

The statistical analysis of the fitness values obtained by solving both strategies without the initial feasible solution is provided in Table 4.3.

Table 4.3 Statistical analysis of the results obtained using both strategies

		ABC		DNLPSO		MPEDE		sTLBO	
		Literature	Proposed	Literature	Proposed	Literature	Proposed	Literature	Proposed
Case 1	B	-1.0E+93	**512.83**	427.62	**713.76**	639.97	**736.21**	683.03	**736.19**
	W	-1.0E+99	**138.16**	**267.25**	-9.1E+22	446.53	**462.71**	518.62	**524.98**
	M	-2.0E+98	**299.55**	**333.33**	-1.3E+22	541.12	**599.11**	624.53	**684.29**
	Md	-1.0E+96	**279.41**	**333.50**	137.20	541.42	**589.61**	631.25	**716.69**
	SD	4.1E+98	**106.61**	**40.01**	2.5E+22	**49.56**	74.68	**39.25**	**55.51**
Case 2	B	-1.0E+93	**568.24**	465.50	**792.37**	762.98	**810.40**	820.49	**826.65**
	W	-1.0E+99	**189.19**	**267.25**	-8.3E+22	538.46	**638.21**	673.58	**720.38**
	M	-2.0E+98	**360.11**	**342.35**	-1.1E+22	649.65	**725.63**	761.81	**768.65**
	Md	-1.0E+96	**360.42**	**337.80**	180.69	646.11	**732.56**	758.16	**775.12**
	SD	4.1E+98	**112.93**	**47.61**	2.1E+22	56.82	**44.71**	34.17	**33.87**
Case 3	B	-1.0E+102	**787.57**	667.60	**1188.68**	947.21	**1204.89**	1024.56	**1212.63**
	W	-1.0E+105	**252.56**	**-9.2E+18**	-5.6E+23	**731.50**	593.46	780.12	**936.33**
	M	-3.2E+104	**485.48**	**-3.5E+17**	-7.4E+22	817.05	**1008.74**	927.4	**1091.51**
	Md	-1.0E+102	**520.88**	**569.97**	-7.7E+20	801.74	**1041.30**	934.31	**1089.61**
	SD	4.8E+104	**144.69**	**1.8E+18**	1.4E+23	**57.93**	125.79	**66.95**	**83.09**
Case 4	B	-1.0E+102	**1004.58**	720.63	**1413.41**	1074.97	**1437.79**	1292.25	**1437.78**
	W	-1.0E+105	**214.33**	**-9.2E+18**	-5.5E+23	744.93	**1007.09**	1056.49	**1177.42**
	M	-3.2E+104	**501.47**	**-3.5E+17**	-8.4E+22	946.63	**1246.90**	1186.42	**1370.84**
	Md	-1.0E+102	**467.60**	**611.46**	-1.0E+21	960.69	**1220.25**	1189.17	**1401.72**
	SD	4.8E+104	**182.88**	**1.8E+18**	1.7E+23	**85.48**	128.65	**55.44**	**96.55**
		HHO		SCA		SOS		SPMGLFTLO	
		Literature	Proposed	Literature	Proposed	Literature	Proposed	Literature	Proposed
Case 1	B	540.75	**636.77**	622.30	**648.62**	600.30	**721.74**	666.87	**735.99**
	W	**268.79**	142.80	410.12	**416.83**	441.31	**516.84**	489.89	**534.61**
	M	384.13	**387.29**	525.61	**537.10**	546.27	**637.15**	590.76	**646.18**
	Md	**384.66**	357.76	**524.19**	518.28	561.87	**634.96**	590.89	**638.45**
	SD	**55.31**	119.68	**57.91**	58.44	**45.33**	63.75	**48.20**	66.24
Case 2	B	591.62	**732.31**	771.13	**789.27**	751.61	**810.40**	771.22	**825.38**
	W	**326.36**	206.52	**562.36**	446.38	464.33	**637.56**	**605.55**	593.49
	M	429.11	**453.06**	**649.28**	631.50	632.22	**741.08**	716.59	**764.90**
	Md	422.77	**431.09**	**651.76**	614.40	629.12	**732.87**	723.55	**775.02**
	SD	**72.48**	129.18	**52.69**	101.31	73.25	**44.48**	**45.96**	58.13
Case 3	B	749.87	**1005.01**	821.01	**1065.51**	887.55	**1203.86**	1030.41	**1204.30**
	W	**428.89**	205.86	**526.28**	387.23	512.30	**767.91**	640.29	**971.51**
	M	**620.65**	476.29	680.49	**881.58**	721.39	**1009.30**	849.99	**1084.52**
	Md	**628.27**	396.26	673.38	**941.01**	726.06	**1028.22**	848.32	**1061.56**
	SD	**89.91**	227.23	**63.08**	175.90	**88.27**	130.09	100.50	**77.10**
Case 4	B	1160.07	**1299.11**	1071.94	**1390.36**	1119.04	**1437.78**	1235.24	**1437.41**
	W	507.58	**522.84**	**718.15**	575.20	669.15	**1053.04**	853.62	**1013.27**
	M	701.58	**769.52**	870.22	**1022.23**	876.99	**1274.58**	1092.93	**1296.24**
	Md	690.28	**719.52**	849.62	**1027.21**	883.37	**1212.37**	1114.92	**1387.13**
	SD	**137.19**	195.72	**98.11**	185.45	**124.67**	141.43	**99.50**	141.12

The best value among both strategies in each statistic is indicated with bold typeface font. On Considering the *best*, *worst*, *mean*, and *median* values, the techniques SOS and SPMGLFTLO provided better performance in solving all the cases with the proposed strategy than the strategy employed in the literature. The *best* value obtained using the proposed strategy is better in all cases than the strategy used in the literature. However, *mean* and *median* values reported by DNLPSO indicate that its performance in all cases of the proposed strategy is not consistently better than values determined using the strategy in the literature where it has identified feasible solutions in fourteen runs in Case 1 and Case 2 and thirteen runs in Case 3 and Case 4.

The performance of ABC in solving the MUCPP cases using the strategy in the literature was not satisfactory, as it was unable to determine at least one feasible solution in all the cases. However, using the proposed strategy, ABC has determined feasible solutions in every run of all four cases. Even though SCA has determined feasible solutions using the strategy in the literature, based on the best value, the performance of SCA is better in all the cases with the proposed strategy. MPEDE and sTLBO provided better values in the best, mean, and median statistical measures while using the proposed strategy to solve all the cases. On solving the MUCPP case studies using the proposed strategy, MPEDE determined the best value in Case 1 and Case 4, whereas in Case 2 and Case 3, the best value is provided by sTLBO.

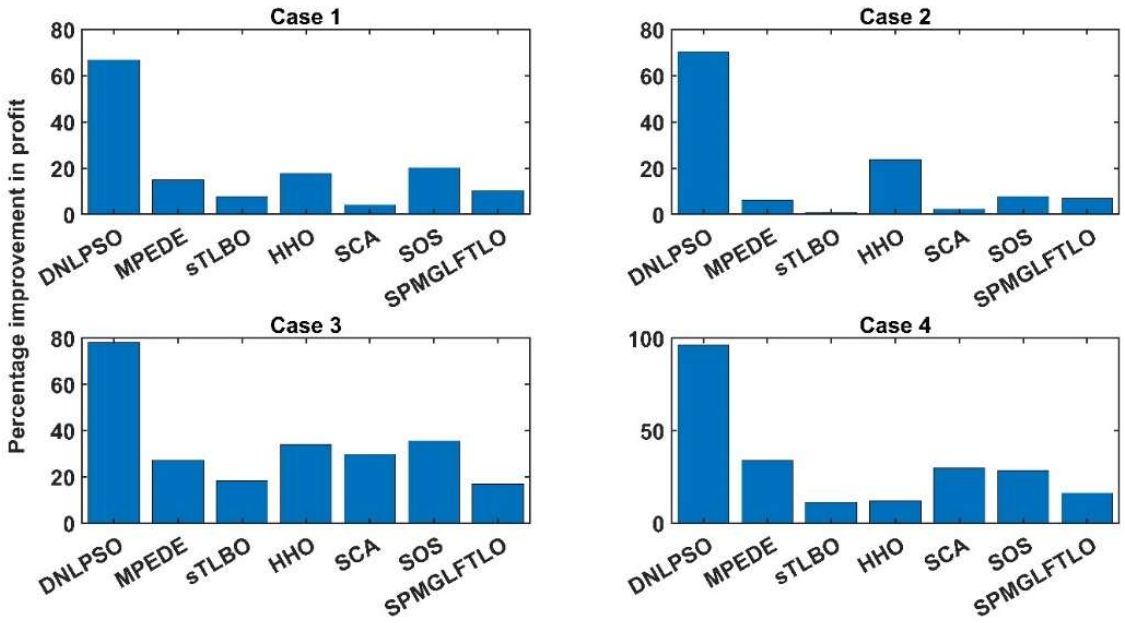


Fig. 4.7 Improvement in profit obtained using the proposed strategy over Chauhan and Kotecha (2018)

The best value determined by all the eight metaheuristic techniques using the proposed strategy is better than using the strategy in the literature. On using the proposed strategy, HHO in Case 3 and SCA in Case 2 are unable to determine a better value in any statistical measures other than best, compared to the strategy in the literature. The percentage improvement of the profit determined by all the techniques except ABC is provided in Fig. 4.7. ABC is able to determine feasible solutions in all cases with the proposed strategy. In contrast, the best fitness reported by ABC in all the cases using the strategy available in the literature is infeasible. Hence, the proposed strategy is found to be more beneficial for ABC than other techniques. Among the rest of the techniques, DNLPSO has benefited the most from the proposed strategy in all cases. STLBO demonstrated the least improvement in the profit in Case 2. It is evident from Fig. 4.7 that the techniques are able to obtain a better solution in solving the MUCPP cases using the proposed strategy.

With the initial feasible solution

The practice of providing a feasible solution in the randomly generated initial population is observed to be helpful for the exploration and exploitation operators of the metaheuristic

techniques in determining better solutions in the early stages of its iterations (Chauhan & Kotecha, 2018). This work has provided an IFS to their initial population to analyze the efficacy of the techniques in determining the better solution for the MUCPP cases using the proposed strategy. The profit corresponding to the initial feasible production plan available in Kadambur and Kotecha (2016) is given in Table 4.4.

Table 4.4 Profit of the initial feasible production plan

	Case 1	**Case 2**	**Case 3**	**Case 4**
Profit ($\$10^6$)	715.85	796.47	1040.15	1287.68

The initial feasible solution considered in this study is the optimal solution to the production planning problem, allowing only one unit each between *l*-*m* and *m*-*h* production capacities. It should be noted that without including the IFS, the proposed strategy has obtained a better solution than the IFS in all the cases.

Table 4.5 Statistical analysis of results obtained with and without IFS

		ABC		**DNLPSO**		**MPEDE**		**sTLBO**	
		With	**Without**	**With**	**Without**	**With**	**Without**	**With**	**Without**
Case 1	**B**	**715.85**	512.83	**728.69**	713.76	**736.21**	**736.21**	736.18	**736.19**
	W	**715.85**	38.11	**715.85**	-9.1E+22	**728.91**	462.71	**726.85**	524.98
	M	**715.85**	273.45	**719.02**	-1.3E+22	**735.18**	599.11	**731.97**	684.29
	Md	**715.85**	254.08	**715.89**	137.20	**736.21**	589.61	**730.52**	716.69
	SD	**0.00**	106.61	**5.07**	2.5E+22	**2.09**	74.68	**3.75**	55.51
Case 2	**B**	**796.47**	500.92	**826.76**	792.37	**826.76**	810.40	**826.75**	826.65
	W	**796.47**	101.64	**796.47**	-8.3E+22	**826.76**	638.21	**812.27**	720.38
	M	**796.47**	303.42	**807.21**	-1.1E+22	**826.76**	725.63	**826.00**	768.65
	Md	**796.47**	300.04	**796.48**	180.69	**826.76**	732.56	**826.65**	775.12
	SD	**0.00**	112.93	**13.43**	2.1E+22	**0.00**	44.71	**2.82**	33.87
Case 3	**B**	**1040.15**	787.57	**1209.01**	1188.68	**1224.55**	1204.89	**1224.50**	1212.63
	W	**1040.15**	178.79	**1040.15**	-5.6E+23	**1212.69**	593.46	**1167.29**	936.33
	M	**1040.15**	434.64	**1106.59**	-7.4E+22	**1222.85**	1008.74	**1213.52**	1091.51
	Md	**1040.15**	429.85	**1040.15**	-7.7E+20	**1224.30**	1041.30	**1214.13**	1089.61
	SD	**0.00**	144.69	**79.62**	1.4E+23	**3.74**	125.79	**11.11**	83.09
Case 4	**B**	**1287.69**	1004.58	**1434.20**	1413.41	**1437.79**	**1437.79**	1437.77	**1437.78**
	W	**1287.68**	214.33	**1287.68**	-5.5E+23	**1437.02**	1007.09	**1391.83**	1177.42
	M	**1287.68**	496.03	**1318.91**	-8.4E+22	**1437.68**	1246.90	**1429.26**	1370.84
	Md	**1287.68**	467.60	**1287.68**	-1.0E+21	**1437.79**	1220.25	**1437.11**	1401.72
	SD	**0.00**	182.88	**55.17**	1.7E+23	**0.20**	128.65	**13.47**	96.55
		HHO		**SCA**		**SOS**		**SPMGLFTLO**	
		With	**Without**	**With**	**Without**	**With**	**Without**	**With**	**Without**
Case 1	**B**	**715.85**	636.77	**720.47**	648.62	**734.05**	721.74	**735.97**	727.90
	W	**715.85**	142.80	**715.85**	416.83	**719.22**	516.84	**715.85**	497.08

	M	**715.85**	387.29	**716.03**	537.10	**728.25**	637.15	**727.79**	629.50
	Md	**715.85**	357.76	**715.85**	518.28	**728.45**	634.96	**728.81**	627.42
	SD	**0.00**	119.68	**0.91**	58.44	**2.70**	63.75	**4.23**	73.05
Case 2	**B**	**796.47**	732.31	**813.40**	789.27	**826.57**	810.40	**826.66**	816.94
	W	**796.47**	206.52	**796.47**	446.38	**796.47**	637.56	**796.47**	574.20
	M	**796.47**	453.06	**797.71**	631.50	**819.57**	741.08	**818.97**	730.41
	Md	**796.47**	431.09	**796.47**	614.40	**825.18**	732.87	**821.82**	733.23
	SD	**0.00**	129.18	**3.87**	101.31	**9.45**	44.48	**8.32**	60.32
Case 3	**B**	**1108.48**	1005.01	**1190.45**	1065.51	**1224.33**	1203.86	**1222.23**	1212.62
	W	**1040.15**	205.86	**1040.15**	387.23	**1136.74**	767.91	**1087.86**	659.88
	M	**1042.78**	476.29	**1045.93**	881.58	**1207.47**	1009.30	**1189.80**	1038.98
	Md	**1040.15**	396.26	**1040.15**	941.01	**1210.84**	1028.22	**1193.89**	1076.39
	SD	**13.40**	227.23	**29.48**	175.90	**17.86**	130.09	**29.95**	129.88
Case 4	**B**	**1342.65**	1299.11	**1392.15**	1390.36	1437.75	**1437.78**	1437.77	**1437.78**
	W	**1287.68**	522.84	**1287.68**	575.20	**1287.68**	1053.04	**1394.90**	941.02
	M	**1289.79**	769.52	**1291.70**	1022.23	**1405.42**	1274.58	**1430.75**	1210.49
	Md	**1287.68**	719.52	**1287.68**	1027.21	**1424.39**	1212.37	**1437.51**	1143.10
	SD	**10.78**	195.72	**20.49**	185.45	**41.84**	141.43	**13.76**	154.14

However, the results reported in Chauhan and Kotecha (2018) are inferior to the IFS in Case 1 and Case 3. The statistical analysis of the fitness value obtained in solving the MUCPP case studies based on the proposed strategy with and without including an IFS in the initial population is provided in Table 4.5.

All techniques except ABC have benefited from including the feasible production plan in the initial population while solving the MUCPP cases based on the proposed strategy. The mean and median values obtained by the techniques in all cases are improved by including the IFS. In all the cases, the standard deviation value of the results obtained by the techniques using the IFS is lesser than without an IFS. It is observed from Table 4.5 that the standard deviation value obtained in all cases of ABC and Case 1 and Case 2 of HHO are equal to zero. Moreover, the value corresponding to all other statistics is also equal to the fitness value of the initial feasible production plan. These statistical values indicate the inefficiency of the search operators of ABC and HHO in the specified cases to identify a better profit than that of the feasible solution. MPEDE determined the best production plan with the highest profit on solving the proposed strategy with the IFS in all the cases. The mean and the median values of the MPEDE algorithm

are close to the best fitness value, which indicates the high efficiency of MPEDE in solving MUCPP problems compared to the other techniques.

The convergence of the *best* and *mean* fitness values corresponding to the best technique for solving the case studies using the proposed strategy with and without IFS is provided in Fig. 4.8. The best and mean fitness of the best technique is plotted against the function evaluation count only when the values are feasible. Hence, in the cases without IFS, the best and mean fitness values corresponding to the initial function evaluations are not shown in Fig. 4.8. In all cases, the deviation of the best fitness and the mean fitness value corresponding to each function evaluation is high in solving the proposed strategy without an IFS compared to the instance with IFS. The mean fitness value and the best value are closer by including the IFS for the best algorithm, which indicates its improved performance in all the runs. A quick convergence to the final fitness value is observed in all the cases with the proposed strategy, while it is not found in the cases with IFS. In all cases except Case 3, the best solution reported by the proposed strategy with and without IFS is approximately equal. This indicates that the proposed strategy for the MUCPP problem can determine the best production plan even without the initial feasible solution.

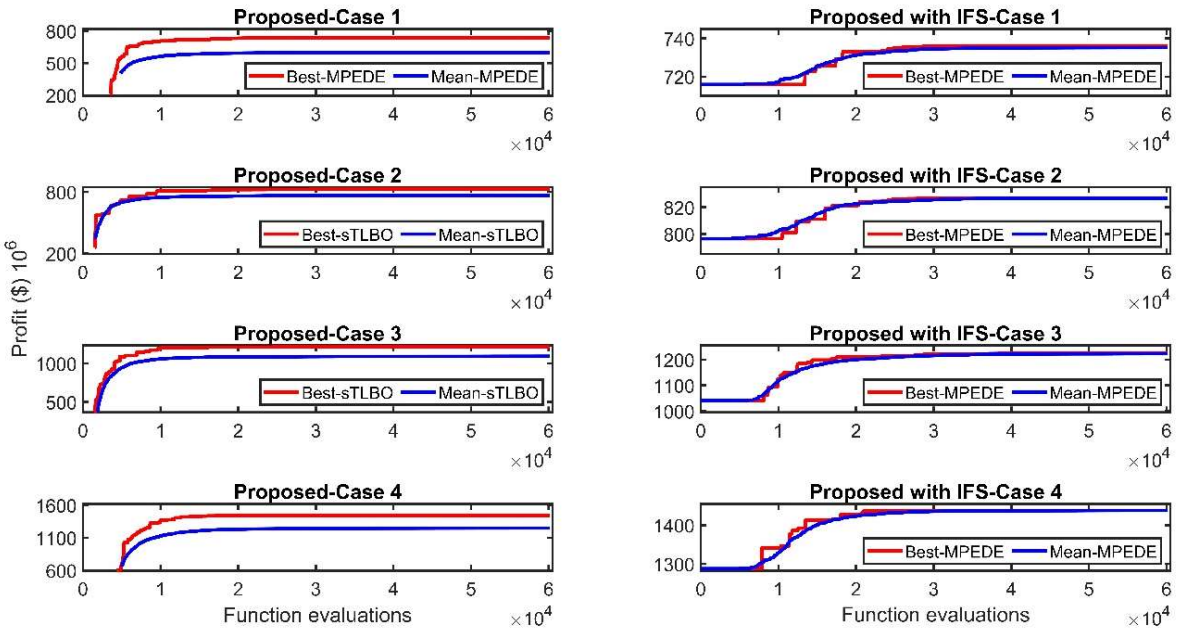


Fig. 4.8 Best and mean convergence of the best technique

4.4.3. Analysis of the best production plan

A comparison of the best fitness value identified and respective techniques in solving the case studies using the proposed strategy and the strategy in the literature with and without including the IFS is provided in Table 4.6. In all cases except Case 2, the best production plan with the highest profit is obtained with the proposed strategy for the MUCPP cases. The best profit reported by MPEDE in Case 1 and Case 4 using the proposed strategy with and without IFS has a magnitude difference of the order 10^{-7} and 10^{-9}, respectively, and is considered equal. The best production plan determined in Case 1 and Case 4 is having an improvement in the profit of approximately 3% compared to the best profit reported in the literature. In Case 2, the best profit reported using the proposed strategy is around 0.3% worse than the value reported in the literature. While in Case 3, with the help of the proposed strategy and IFS, MPEDE has determined a plan having nearly 5% better profit than the literature.

Table 4.6 Comparison of the best fitness value

		Literature		Proposed	
		Without IFS	With IFS	Without IFS	With IFS
Case 1	Profit	683.03	716.80	**736.21**	**736.21**
	Technique	sTLBO	MPEDE	**MPEDE**	**MPEDE**
Case 2	Profit	820.49	**829.00**	826.65	826.76
	Technique	sTLBO	**sTLBO**	sTLBO	MPEDE
Case 3	Profit	1024.56	1165.50	1212.63	**1224.55**
	Technique	sTLBO	sTLBO	sTLBO	**MPEDE**
Case 4	Profit	1292.25	1399.10	**1437.79**	**1437.79**
	Technique	sTLBO	sTLBO	**MPEDE**	**MPEDE**

The pie charts showing the contribution of each product to the total profit in all cases are provided in Fig. 4.9.

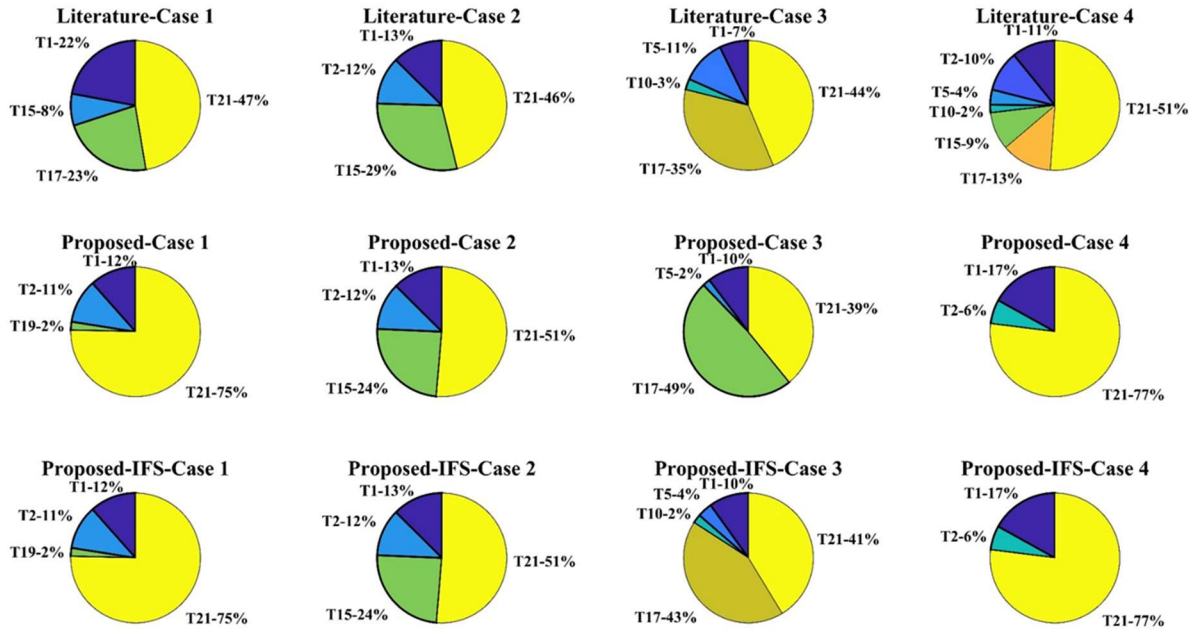


Fig. 4.9 Profit contribution of each manufactured product in the best solution.

The pie chart labels specify the product number and the percentage of its contribution to the profit. In solving MUCPP cases, the selected products and their profit contribution are the same for the proposed strategy with and without IFS, except in Case 3. The deviation is evident as the quantity of the fifth and twenty-first products is higher in the production plan obtained using the proposed strategy with IFS. In addition, the production plan with the IFS has produced the tenth product and reduced the production of the seventeenth product. The first and twenty-first products are manufactured based on the best production plan determined in all cases, indicating that those are the most beneficial products.

4.5. Conclusion

This chapter has analyzed the strategy employed in the literature for solving the multi-unit production planning problem to guide the petrochemical industry at its developmental stage. The shortcomings of the strategy utilized in the literature are identified, and a compact strategy is proposed to overcome them. The proposed strategy is able to reduce the complexity of the model by decreasing the number of variables up to 55 times and constraints by 25 times

compared to the literature. The solution structure utilized in the proposed strategy inherently satisfies the unique process constraints, which decreases the number of constraints to be handled. The choice of decision variables with respect to the products instead of the processes and number of units, as in the strategy utilized in the literature, is the primary reason behind the reduction in the problem dimension of the proposed strategy. The efficacy of the proposed strategy is analyzed by solving the four case studies with eight metaheuristic techniques.

The performance of all metaheuristic techniques has improved substantially in determining a better solution using the proposed strategy than the literature, indicating its suitability. The proposed strategy helped ABC in obtaining a feasible production plan for every run. At the same time, ABC could not determine a feasible solution in any case of MUCPP problem solved using the strategy available in the literature. The DNLPSO identified a production plan using the proposed strategy with a 96% increase in the profit compared to the value determined using the strategy followed in the literature. Additionally, the proposed strategy benefited the other techniques by aiding them to perform better than the strategy in the literature. Among the techniques used in this work, MPEDE is able to determine better production plans using the proposed strategy in most of the case studies than the strategy in the literature.

Chapter 5

Multi-objective optimization of multi-unit production planning with unique process constraints

In this chapter, the multi-unit production planning model studied in Chapter 4 is extended to the multi-objective model considering two conflicting objectives: maximization of profit and minimization of investment cost. The proposed optimization strategy is applied to analyze its efficiency in solving the multi-objective model. This study employed eight metaheuristic techniques to analyze the efficacy of the proposed multi-objective strategy of multi-unit production planning to develop the petrochemical industry. The chapter is structured as follows: Section 5.1 describes the multi-objective, multi-unit production planning model. Section 5.2 specifies the details of the case studies and other experimental settings. Section 5.3 analyzes the results provided by the metaheuristic techniques in solving the multi-objective case studies. Finally, Section 5.4 concludes this chapter with the major findings of the work.

5.1. Multi-objective production planning problem

The MUCPP model discussed in Chapter 4 is extended to solve the multi-objective problem considering maximization of profit and minimization of investment cost as the objectives subject to investment budget constraints, raw material constraints, unique process constraints, and domain hole constraints. The beneficial process for the product specified in a potential solution and the feasibility of the solution are determined in the same way as in Chapter 4.

In the proposed strategy for solving the multi-objective MUCPP problem, the process selected to produce any product is based on its highest profitability among other processes. Another possible way to select a particular process for manufacturing a product is based on the investment cost of implementing the process. In that scenario, selection has to be made based on the process with the least investment cost, as the objective is to minimize the investment cost required. The least value of investment cost is zero, which occurs when no product is

produced. Hence, selecting a beneficial process based on minimum investment cost might result in a solution pool with all zeros as the iterations of the metaheuristic technique proceed. Due to this reason, the current study has not considered selecting a beneficial process for a product as per the least investment cost.

The fitness value of a potential schedule corresponding to maximization of profit (f_1) can be determined using Eq. (5.1), where λ is the penalty factor. The proposed strategy ensures a feasible production plan in terms of domain and unique process constraints. Hence, when there is a violation of investment budget constraints, raw material constraints, or both, the fitness value of a potential plan is penalized.

$$\text{Max } f_1 = \sum_{i=1}^{I} Profit_i + \lambda\left(P^{Investment} + P^{Raw}\right) \tag{5.1}$$

The second conflicting objective is the minimization of the total investment cost, and the fitness value corresponding to it can be determined as Eq. (5.2).

$$\text{Min } f_2 = V^{Total} + \lambda\left(P^{Investment} + P^{Raw}\right) \tag{5.2}$$

5.2. Experimental settings

The same four case studies considered in Chapter 3 and Chapter 4 are used to analyze the efficacy of the proposed strategy in solving the multi-objective model of MUCPP. This study has considered eight multi-objective metaheuristic techniques namely Multi-objective Particle Swarm Optimization (MOPSO) (Coello et al., 2004), Fast and elitist multi-objective genetic algorithm (NSGA-II) (Deb et al., 2002), Nondominated Neighbor Immune Algorithm (NNIA) (Gong et al., 2008), Multi-objective Differential Evolution with spherical pruning (spMODE) (Reynoso-Meza et al., 2010), Multi-objective Cuckoo Search (MOCS) (Yang & Deb, 2013), Multi-Objective Slime Mould Algorithm (MOSMA) (Premkumar et al., 2021), Multi-objective Water Cycle Algorithm (MOWCA) (Sadollah et al., 2015), and Multi-objective Salp Swarm

Algorithm (MSSA) (Mirjalili et al., 2017) to solve the case studies. A brief description of the metaheuristic techniques is given in Appendix A

The population size for all the metaheuristic techniques is set to 100. The termination criterion set for all the algorithms is the fixed number of maximum function evaluations, and it is 60,100 for all the problems. On considering the stochastic nature of the metaheuristic techniques, each problem is solved 30 times using each metaheuristic technique. The simulations are performed in MATLAB R2020b environment installed in a machine with Intel Core i7 @ 4.20 GHz processor and 48 GB RAM.

This work employed the hypervolume ratio (Deb, 2001) to analyze the closeness and diversity of the determined Pareto solutions. A hypercube is constructed by considering the i^{th} solution in the obtained Pareto (Q) and nadir point (a vector of worst objective values) as the diagonal points. The hypervolume of Q is the summation of hypercube volume as Eq. (5.3). The hypervolume ratio of Q with respect to the global Pareto (P^*) is determined as Eq. (5.4).

$$HV = volume\left(\bigcup_{i=1}^{|Q|} v_i\right) \tag{5.3}$$

$$HVR = \frac{HV(Q)}{HV(P^*)} \tag{5.4}$$

This study also utilized the coverage metric (Deb, 2001) to identify the proportion of obtained Pareto solutions (Q) present in the best-known Pareto front (P*) as Eq. (5.5).

$$C_{2R} = \frac{\left|\left\{q \in Q \;\exists p \in P^* : p \preceq q\right\}\right|}{|Q|} \tag{5.5}$$

5.3. Results and discussion

This section analyses the hypervolume ratio obtained in each run using metaheuristic techniques with the help of the proposed strategy for solving four case studies. The non-dominated sorting method is employed to determine the best-known Pareto front from the set of all non-dominated solutions identified by the techniques. The set of non-dominated solutions, determined by considering all the runs of an algorithm, is compared with the best-known Pareto front of a particular case. The corner point corresponding to each objective function is identified from the best-known Pareto front. A comparison between the best-known corner points and the corner points reported by each algorithm is also analyzed.

5.3.1. Analysis based on performance metrics

The results depicting the quality analysis of non-dominated solutions determined by the metaheuristic techniques based on performance metrics such as hypervolume ratio and coverage metric are provided below.

Analysis of hypervolume ratio

The hypervolume ratio (*HVR*) is utilized to examine the convergence of the non-dominated solutions provided by each technique with the best-known Pareto front. The box plot depicting the variation of the hypervolume ratio obtained by the metaheuristic techniques in each run in solving the MUCPP cases is provided in Fig. 5.1. The red line in the interquartile region represents the median value of the hypervolume ratio. The outliers of the hypervolume ratio of each algorithm are indicated by the red-colored '+'.

The box plot corresponding to NSGA II is unavailable in Case 3 and Case 4, as it has not identified feasible solutions. MSSA shows the maximum deviation of the hypervolume ratio except in Case 1, and NNIA provides the minimum variation compared to all other algorithms. In all the cases, NNIA has provided the best median for *HVR* except Case 4, where the median *HVR* of MOSMA is better. Although the median of MOSMA is better than NNIA in Case 4,

the variation in the *HVR* is comparatively lesser in NNIA. Hence it can be inferred that the consistency in providing better Pareto fronts in all runs is better with NNIA than MOSMA. It can also be observed that the *HVR* of NNIA is always close to one in every case. In all cases except Case 2, the median *HVR* of MOWCA is worse than all other algorithms, whereas in Case 2, MSSA provided the worst *HVR*.

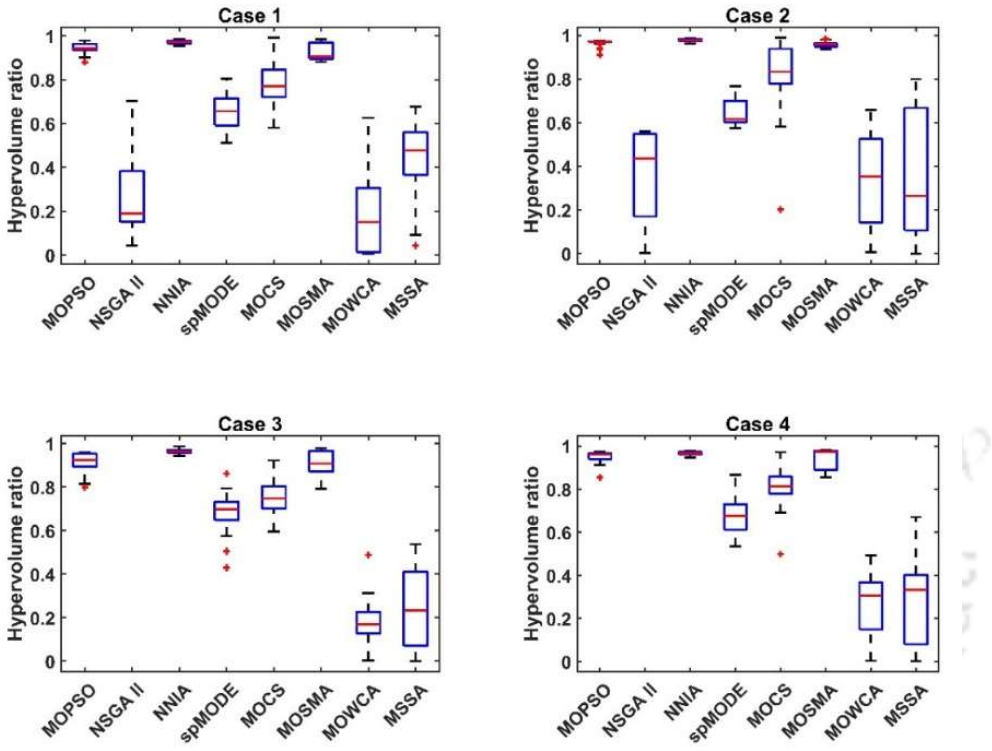


Fig. 5.1 Variation of the hypervolume ratio of each algorithm

Analysis of coverage metric

This study has used the set coverage metric (C_{2R}) to quantify the proportion of the Pareto front determined by an algorithm available in the best Pareto front. The best hypervolume ratio and the coverage metric of the Pareto front determined by each algorithm for solving the MUCPP cases are provided in Table 5.1. In all cases, the algorithm with the best hypervolume ratio has provided a better coverage metric than those algorithms providing an *HVR* closer to it. However, in Case 4, MOSMA has provided the best *HVR*, while NNIA, with a slightly lesser *HVR*, has provided a better coverage metric value. NSGA II on solving the MUCPP cases is

unsatisfactory as the hypervolume ratio and the coverage metric identified by this algorithm are the worst. The coverage metric of NSGA II is close to zero, indicating the minimal convergence of the solutions identified by it with the best-known Pareto front.

Table 5.1 Coverage metric (C2R) and hypervolume ratio (HVR) of algorithms

Algorithms	**Case1**		**Case2**		**Case3**		**Case4**	
	HVR	C_{2R}	*HVR*	C_{2R}	*HVR*	C_{2R}	*HVR*	C_{2R}
MOPSO	0.9888	0.6452	0.9879	0.5575	0.9787	0.5578	0.9880	0.5587
NSGA II	0.6989	0.0696	0.5324	0.0074	–	–	–	–
NNIA	**0.9981**	0.7845	**0.9978**	0.8690	**0.9977**	0.7408	0.9944	0.7160
spMODE	0.9757	0.6556	0.8807	0.7897	0.9120	0.3566	0.9566	0.6111
MOCS	0.9908	0.7137	0.9919	0.6503	0.9352	0.6166	0.9716	0.6396
MOSMA	0.9955	0.7440	0.9962	0.6268	0.9937	0.7279	**0.9952**	0.7020
MOWCA	0.6725	**0.9356**	0.6620	**0.8792**	0.4993	**0.9101**	0.5318	**0.8757**
MSSA	0.8297	0.5404	0.8719	0.4940	0.6413	0.4506	0.6847	0.4252

5.3.2. Analysis of the Pareto solutions

In this section, the corner solutions of the Pareto solutions determined by the metaheuristic techniques are analyzed. Also, the convergence of the non-dominated solution identified by each technique is compared with the best-known Pareto front. Additionally, the product portfolio of the Pareto solution with maximum profit is provided in this section.

Analysis of corner points

The corner point corresponding to the maximum profit and maximum total investment cost (TIC) determined by each algorithm for solving the case studies of MUCPP problems are provided in Table 5.2. The best-known profit provided in Table 5.2 is the best profit obtained by solving the MUCPP problem as a single objective optimization problem.

The corner points corresponding to NSGA II and FYYPO in Cases 3 and 4 are not provided as these algorithms are unable to identify feasible solutions. In all the cases, the algorithm that has determined the best corner point has reached 99% of the best-known profit. MOWCA has

identified the corner point with minimum profit (worse value) compared to other algorithms, implying that it has an unexplored region in the best-known Pareto front that is close to maximum profit and minimum total investment cost. Most algorithms except NSGA II and MOWCA dominate the corner point determined by MSSA in Case 1 of the MUCPP model.

Table 5.2 Corner points indicating maximum profit and maximum total investment cost (TIC)

Algorithm	**Case 1**		**Case 2**		**Case 3**		**Case 4**	
	Profit	TIC	Profit	TIC	Profit	TIC	Profit	TIC
MOPSO	716.14	996.77	803.31	999.54	1184.28	1998.98	1422.07	1977.29
NSGA II	533.28	986.25	538.66	824.10	–	–	–	–
NNIA	734.21	993.19	824.60	999.99	**1224.02**	**1996.72**	1437.34	1992.26
spMODE	727.55	963.65	823.64	998.92	1189.40	1993.17	1419.54	1993.86
MOCS	**734.44**	**999.48**	815.25	999.88	1109.32	1993.17	1393.84	1999.44
MOSMA	727.76	963.61	**825.16**	**999.81**	1202.67	1999.84	**1434.48**	**1989.12**
MOWCA	425.06	780.95	356.49	381.96	427.88	783.52	519.91	932.98
MSSA	546.21	999.59	644.22	995.81	843.11	1992.28	740.31	1962.16
Best profit	736.21	–	826.76	–	1224.55	–	1437.79	–

The corner point of minimum profit and minimum total investment cost of all metaheuristic techniques in each case of the MUCPP model is given in Table 5.3. The global optima corresponding to the minimum total investment cost and minimum profit has an objective function value equal to zero for both objectives.

Table 5.3 Corner points indicating minimum total investment cost (TIC) and minimum profit

Algorithms	**Case 1**		**Case 2**		**Case 3**		**Case 4**	
	Profit	TIC	Profit	TIC	Profit	TIC	Profit	TIC
NSGA II	0	0	0	0	–	–	–	–
MSSA	0	0	0	0	0.95	15.42	0	0
Other algorithms	0	0	0	0	0	0	0	0

The global optimum for minimum total investment cost is zero, implying no investment has been made to implement any processing units, resulting in zero profit. All algorithms other than NSGA II, and MSSA have identified the best corner point in all the cases. NSGA II

identified the best corner point in Case 1 and Case 2 but could not identify feasible solutions in the other two cases. MSSA has identified the best corner point in all cases except Case 3.

Convergence of non-dominated solutions

The convergence of the Pareto front obtained by algorithms, along with the best-known Pareto front is shown in Fig. 5.2.

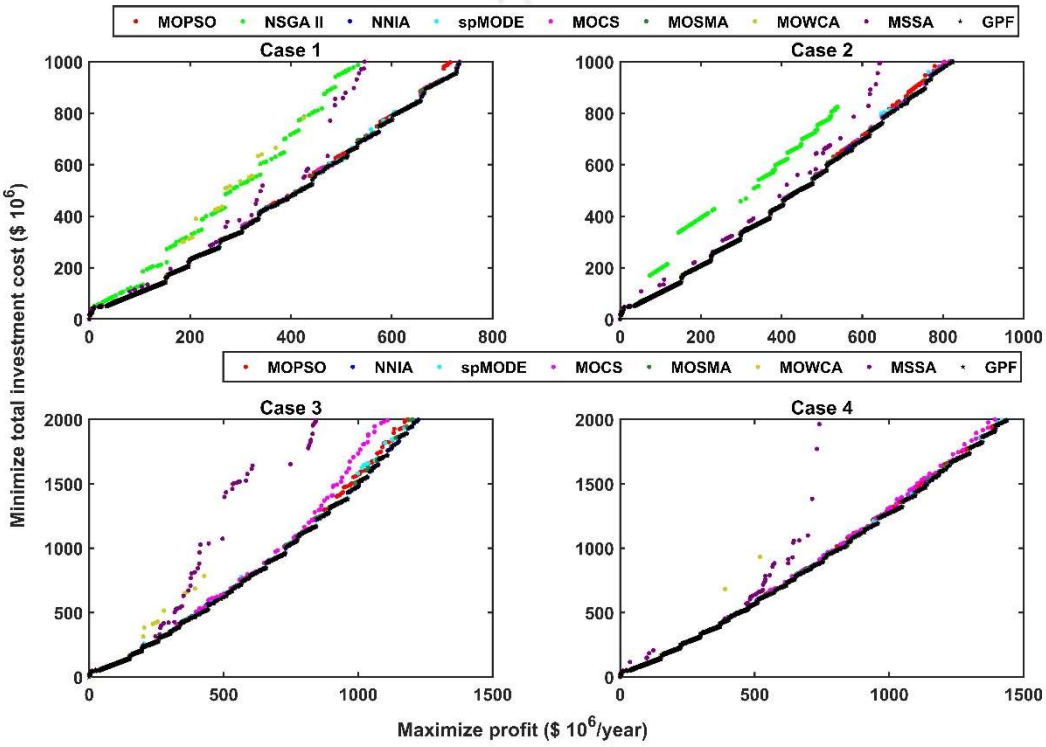


Fig. 5.2 Non-dominated solutions reported by algorithms

In every case, the convergence of all the algorithms in the region of minimum total investment cost and minimum profit is comparatively better than the region representing maximum profit and maximum total investment cost. The non-dominated solutions determined by NNIA are well converged with the best-known Pareto front, followed by MOSMA in all the cases. As NSGA II is unable to provide feasible production plans in Case 3 and Case 4, the solutions corresponding to NSGA II is not provided in Fig. 5.2 MOWCA is unable to explore the region of the Pareto front representing the maximum profit and maximum investment cost in all the

cases and a similar scenario is also observed with MSSA. The performance of NSGA II, MOWCA, and MSSA is not satisfactory, as most of the non-dominated solutions determined by these algorithms are dominated by the solutions of other algorithms.

Product portfolio of the solution with maximum profit

The profit distribution of the global Pareto solution corresponding to the corner point with maximum profit is provided in Fig. 5.3.

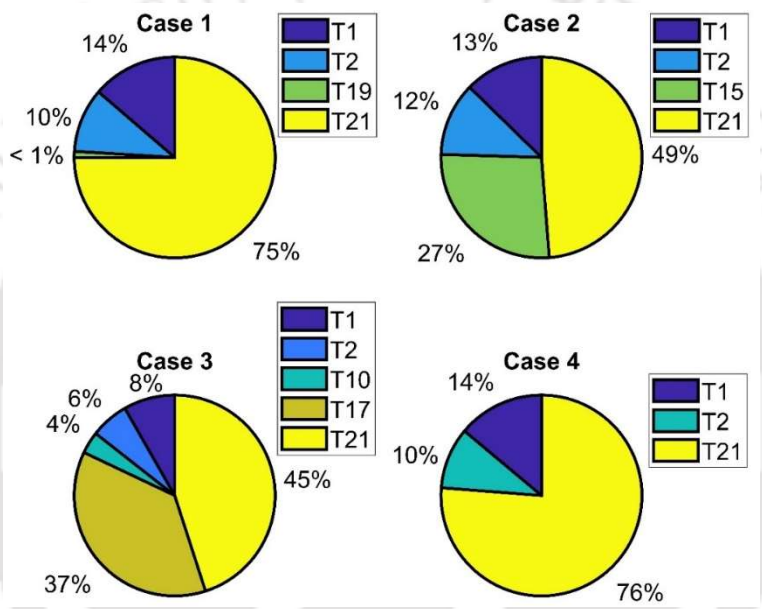


Fig. 5.3 Profit distribution of the corner solution with maximum profit

In all the cases, product T21 is observed to be more profitable as its contribution is high compared to other selected products. The other common products that are produced in all the cases are T1 and T2; however, their contribution to the total profit is lesser than product T21. The availability of the investment budget in Case 1 and Case 2 is the same, whereas the amount of raw materials is more in Case 2 than in Case 1. The number of products produced in Case 1 and Case 2 are equal. The profit contribution from product T19 is less than 1% in Case 1, while it is replaced with product T15 in Case 2, and the contribution to the total profit is the second-highest preceded by T21. These observations related to products T19 and T15 imply the

production of T15 is more beneficial; however, the lack of raw material availability in Case 1 selected T19 instead of T15. Case 4 has the maximum investment budget and raw material availability among all the cases. Hence the most profitable product T21 is produced maximum by employing multiple units. In Case 3, due to the deficit of raw materials, multiple products are produced instead of producing profitable products in large quantities.

The details, such as product choice, selected process, implemented units and their capacity, and the resource utilization of the corner point with maximum profit, are provided in Table 5.4.

Table 5.4. Analysis of the corner solution with maximum profit

	Product	Process	Total quantity	Number of units	Capacity of unit	R1 utilized	R2 utilized	TIC	Profit
Case 1	T1	P3	294.32	1	294.32	499.71	499.61	999.48	734.44
	T2	P4	230.87	1	230.87				
	T19	P41	58.95	2	29.47				
	T21	P48	1757.34	3	585.78				
Case 2	T1	P3	305.29	1	305.29	565.43	921.32	999.81	825.16
	T2	P4	288.83	1	288.83				
	T15	P31	589.28	2	294.64				
	T21	P48	1279.63	2	639.82				
Case 3	T1	P3	295.90	1	295.90	499.53	499.98	1996.72	1224.02
	T2	P4	229.12	1	229.12				
	T10	P22	95.85	1	95.85				
	T17	P36	2188.02	5	437.60				
	T21	P48	1758.62	3	586.21				
Case 4	T1	P3	616.22	3	205.41	999.97	999.88	1992.26	1437.34
	T2	P4	434.92	2	217.46				
	T21	P48	3516.98	7	502.43				

The products T1, T2, and T21 are produced in all the cases using processes P3, P4, and P48, respectively. This implies that among the multiple processes available to produce these products, the solution strategy employed in this study helped the metaheuristic techniques identify the most profitable process. It is observed that more than 99% of the investment budget is utilized in the corner solution with maximum profit. A similar scenario is also observed in

all cases except Case 3 in the utilization of raw materials. In Case 3, the raw materials R1 and R2 remained unused at approximately 43% and 8%, respectively.

5.4. Conclusion

This chapter solves a multi-objective production planning problem for developing the petrochemical industry that determines the trade-off between the investment budget and the profit using an efficient solution strategy well suited for metaheuristic techniques. The performance of the eight multi-objective metaheuristic techniques in solving the multi-objective production planning by utilizing the solution strategy is analyzed. All the techniques, except NSGA II, have determined feasible non-dominated solutions in all four case studies. On considering the hypervolume ratio and the coverage metric, the non-dominated solutions identified by the NNIA are comparatively better converged and diversified than the other algorithms. The corner point corresponding to the maximum profit determined in each case has utilized more than 99% of the investment budget. The profit value is also close to the optima of a single objective model. All algorithms except MSSA and NSGA-II determine the ideal scenario of zero profit corresponding to a zero investment cost in the second corner point representing minimum profit and minimum investment cost.

Chapter 6

Efficient Scheduling of Jobs on Dissimilar Parallel Machines using Heuristic Assisted Metaheuristic Techniques

This chapter proposes an optimization framework involving metaheuristic techniques and a new no-wait time heuristic mechanism to solve the scheduling of multiple jobs on dissimilar parallel machines. The rescheduling of jobs based on the proposed heuristic mechanism attempts to minimize constraint violations, resulting in the identification of feasible solutions in the earlier stages of metaheuristic techniques. The performance of the proposed framework is analyzed using multiple metaheuristic techniques by solving twelve instances of a scheduling problem. This study also studies the impact of tightening constraints on the optimization framework. This chapter has been structured as follows: Section 6.1 describes the scheduling problem used in this Chapter. Section 6.2 explains the heuristic-assisted optimization framework, which includes the mathematical formulation suitable for metaheuristic techniques and the proposed no-wait time heuristic mechanism. Section 6.3 provides the experimental settings related to the metaheuristic techniques, case studies, and other parametric values. Section 6.4 discusses the results demonstrating the benefits of the no-wait time heuristic mechanism. Finally, this chapter is concluded by providing the summary of this study in Section 6.5.

6.1. Problem statement

The scheduling problem considered in this chapter (Jain & Grossmann, 2001) constitutes a set of I independent orders/jobs and M machines. Each order has a release date (r_i) and a due date (d_i). The jobs can be processed using any of the available machines. As the machines are dissimilar, the processing time (p_{im}) and the processing cost (c_{im}) of a job on each machine are different and the respective data are already known. All the orders must be processed on any of the available dissimilar parallel machines. The various constraints are

Kommadath, R., D. Maharana, and P. Kotecha, *Efficient scheduling of jobs on dissimilar parallel machines using heuristic assisted metaheuristic techniques.* Chemical Engineering Research and Design, 2022. **188**: p. 916-934. DOI: https://doi.org/10.1016/j.cherd.2022.10.011

(i) Each machine can process multiple orders.

(ii) An order assigned to a machine should be entirely processed on the same machine.

(iii) Processing of an order on a machine should be completed once it has started.

(iv) Overlap constraints: All machines should process one order at a time.

(v) Release date constraints: Processing of orders must begin on or after their release date.

(vi) Due date constraints: Processing of orders should be completed on or before their due date.

The objective is to identify an optimal schedule with a minimum processing cost that satisfies the constraints mentioned above.

6.2. Proposed heuristic-assisted metaheuristic framework

This study employs the metaheuristic technique to determine the optimal solution for solving the scheduling problem mentioned above. As the metaheuristic techniques do not require the problem to be modeled in canonical form, the mathematical formulation that is suitable for metaheuristic techniques is provided below. A no-wait time heuristic mechanism proposed in this study to assist the metaheuristic techniques is also explained in this section.

6.2.1. Problem formulation suitable for metaheuristic techniques

The detailed mathematical model of the scheduling problem is provided in this section.

Decision variables: The machines assigned to process each order and the start time of an order on the allotted machine are the decision variables of this scheduling problem. Unlike the classical optimization models, this structure does not require binary variables to represent the allotted machine to process each order and to indicate the sequence for processing the order on a machine. The starting time of all the orders allotted to a specific machine provides their processing sequence. The structure of a schedule with I orders is as shown in Fig. 6.1. The first set of I variables indicates the machines assigned to process each order, whereas the second set of I variables indicates the starting time of the orders on its allotted machines.

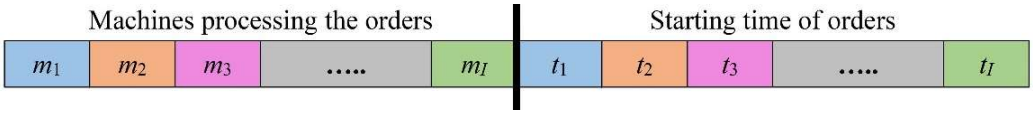


Fig. 6.1 Solution structure

The illustration of the solution structure with a numerical example is given below. As per the schedule given above, the first and fourth orders are assigned to machine 2, and their starting time is 3 and 9, respectively. Similarly, orders 2 and 3 are allotted to machine 1, which would start the processing at 10 and 4, respectively.

$$X = \left[\overbrace{2 \quad 1 \quad 1 \quad 2}^{machines} \quad \overbrace{3 \quad 10 \quad 4 \quad 9}^{starting\ time} \right]$$

Bounds of the decision variables: The lower and upper bounds of the first set of decision variables are one and the maximum number of machines available, respectively. The lower bound of the variables representing the starting time of the orders is their respective release date. The corresponding upper bound is the difference between the due date and the least processing time required to process the order among all machines, and such a choice for the upper bound tightens the bound of the starting time variables. Any value below the release date for the variables representing the starting time indicates a bound violation. Hence, the variable satisfying the lower bound constraint also satisfies the release date constraints.

Constraints: The solution structure inherently satisfies the following constraints: (i) specifying each machine can process multiple orders, and (ii) constraint representing the order assigned to a machine should be processed completely on the same machine. The rest of the constraints are handled using the penalty approach, as discussed below.

Due date constraints: The ending time of i^{th} order $\left(end_i\right)$ is the sum of its starting time $\left(t_i\right)$ and its processing time $\left(p_{im_i}\right)$ on the assigned machine $\left(m_i\right)$ as given in Eq. (6.1).

$$end_i = t_i + p_{im_i} \quad \forall i \in 1,2,...,I \tag{6.1}$$

To satisfy due date constraints, the end time of any order should not exceed its corresponding due date (d_i). The violation of due date constraints $\left(V_i^{due}\right)$ and the corresponding penalty $\left(P_i^{due}\right)$ incurred are determined as Eq. (6.2) and Eq. (6.3), respectively. The total number of due date constraints is equal to the number of orders.

$$V_i^{due} = end_i - d_i \quad \forall i \in 1,2,...,I \tag{6.2}$$

$$P_i^{due} = \begin{cases} 0 & if\ d_i \geq end_i \\ V_i^{due} & otherwise \end{cases} \quad \forall i \in 1,2,...,I \tag{6.3}$$

Overlap constraints: The processing of any two orders assigned to a machine should not overlap, which implies the starting time of the subsequent order $\left(t_{j+1}^m\right)$ should be greater than or equal to the end time of its previous order $\left(end_j^m\right)$. In addition, the impact of overlap constraints is only established for machines that process multiple orders. The magnitude of violation in the overlap constraint $\left(V_j^m\right)$ is calculated as Eq. (6.4), and penalty $\left(P_j^m\right)$ to be assigned is determined as Eq. (6.5). The total number of such constraints related to *M* machines is determined as $\sum_{m=1}^{M}\left(\left|I_m\right|-1\right)$ where $\left|I_m\right|$ indicates the cardinality of the set I_m.

$$V_j^m = end_j^m - t_{j+1}^m \quad \forall j \in I_m, m \in 1,2,...,M : t_j^m < t_{j+1}^m \tag{6.4}$$

$$P_j^m = \begin{cases} 0 & if\ end_j \leq t_{j+1} \\ V_j^m & otherwise \end{cases} \quad \forall j \in \mathrm{I}_m, m \in 1,2,...,M : t_j < t_{j+1} \tag{6.5}$$

Horizon constraints: These constraints restrict processing all orders allotted to a particular machine on or before the time horizon, which is the difference between the maximum due date and minimum release date (r_i). The time horizon (T_m) is determined as in Eq. (6.6), and total time required for processing all orders assigned to m^{th} machine is calculated as Eq. (6.7).

$$H = \max_{i=1,2,..,I}\left(d_i\right) - \min_{i=1,2,..,I}\left(r_i\right) \tag{6.6}$$

$$T_m = \sum_{j=1}^{|I_m|} p_{jm} \quad \forall m \in 1,2,...,M \quad (6.7)$$

The violation of horizon constraints $\left(V_m^H\right)$ is calculated using Eq. (6.8), and the corresponding penalty $\left(P_m^H\right)$ by using Eq. (6.9). The total number of horizon constraints is equal to *M*.

$$V_m^H = T_m - H \quad (6.8)$$

$$P_m^H = \begin{cases} 0 & if\ H \geq T_m \\ V_m^H & otherwise \end{cases} \quad \forall m \in 1,2,...,M \quad (6.9)$$

The total penalty $\left(P^{Total}\right)$ associated with the violation of different constraints is calculated using Eq. (6.10) (Deb, 2001). The number of constraints related to a problem with *I* orders and *M* machines is $(I + 2M)$.

$$P^{Total} = \sum_{i=1}^{I} P_i^{due} + \sum_{m=1}^{M} \sum_{j=1}^{|I_m|} P_j^m + \sum_{m=1}^{M} P_m^H \quad (6.10)$$

Fitness function: The fitness function (f) for determining a schedule with minimum cost while satisfying all the constraints is given in Eq. (6.11). The fitness function of an infeasible solution constitutes (i) the total cost associated with the schedule, (ii) the sum of all penalties corresponding to each constraint violation, and (iii) the sum of the maximum cost for processing an order among all machines. The third term in the cost calculation of an infeasible schedule ensures that it will not have a better fitness value than the worst feasible solution.

$$f = \begin{cases} \sum_{i=1}^{I} c_{im_i} & if\ P^{Total} = 0 \\ \sum_{i=1}^{I} c_{im_i} + P^{Total} + \sum_{i \in I} \max_{m \in M} \{c_{im}\} & otherwise \end{cases} \quad (6.11)$$

The overall optimization formulation of the scheduling model is provided in Eq. (6.12). It should be noted that the formulation provided in Eq. (6.12) should be used in conjunction with the proposed strategy discussed above.

$$
\begin{aligned}
&\textit{Minimize Total processing cost}\left(f\right)\\
&\textit{s.t.},\\
&\quad end_i \le d_i \quad \forall i \in 1,2,...,I\\
&\quad end_j^m \le t_{j+1}^m \quad \forall j \in I_m, m \in 1,2,...,M : t_j^m < t_{j+1}^m\\
&\quad T_m \le H \quad \forall m \in 1,2,...,M\\
&\quad 1 \le m_i \le M\\
&\quad r_i \le t_i \le \left(d_i - \min_{m=\{1,2,...,M\}}\left(P_{im}\right)\right)
\end{aligned}
\tag{6.12}
$$

6.2.2. No-wait time heuristic mechanism

Some metaheuristic techniques are unable to determine even a single feasible solution for problems with a large number of jobs and machines using the discussed formulation (Kotecha et al., 2010). This study has proposed a heuristic mechanism that attempts to reduce the violation of overlapping constraints and assist the metaheuristic techniques to attain bettersolutions. The proposed heuristic mechanism restructures a potential solution provided by the metaheuristic technique before evaluating its objective function, irrespective of its feasibility status. Subsequently, the repaired solution and its fitness value are returned to the metaheuristic technique. Based on the survival strategy utilized in the metaheuristic technique, the repaired solution is accepted into the population. The detailed pseudocode of the no-wait heuristic mechanism is provided in Fig. 6.2.

As per this proposed heuristic mechanism, if multiple orders are assigned to a machine, the order with the least starting time is processed at its release date. The subsequent orders are processed immediately after the completion of previous orders. The starting time of any orders other than the first order is determined as in Eq. (6.13).

$$
t_{j+1} = \max\left(r_{j+1}, end_i\right) \quad \forall j \in 2,3...,I_m \tag{6.13}
$$

For $m = 1$ to M

Identify all orders processed on m^{th} machine as I_m

If $|I_m| > 1$

Sort orders (I_m) with respect to their starting time as (I_s)

Replace the start time of the first order I_s^1 with its release date

For $k = 2$ *to* $|I_m|$

Replace the start time of k^{th} order ($t_{I_s^k}$) as ($end_{I_s^{k-1}}$)

Bound $t_{I_s^k}$ to the release date of I_s^k, if it violates the lower bound

End For

Identify the set of orders whose start time (t_{I_s}) exceeds the upper bound and represent that set as I_{ub}

If $|I_{ub}| \geq 1$

Replace the starting time of orders (I_{ub}) to its upper bound

Sort the orders (I_m) with respect to their updated starting time as I_s

Determine the end time (end_{I_s}) of all orders in I_s using Eq. (6.1)

For $i = 1$ to $|I_m| - 1$

Calculate violation and incurred penalty of i^{th} order corresponding to the overlap constraints using Eq. (6.4) and Eq. (6.5), respectively

End For

End If

Else

Replace the starting time of the order equal to its release date

End If

End For

Fig. 6.2 Pseudocode for the no-wait time heuristic mechanism

The steps followed while solving the scheduling problem using metaheuristic techniques and the heuristic mechanism are provided as a flowchart in Fig. 6.3. The proposed heuristic mechanism reduces the idle time of the machines, as the processing of subsequent jobs assigned to a machine is rescheduled to begin immediately after its previous order. If a modified solution becomes feasible and there exists idle time between the processing of two jobs on a machine, then it implies the release date of the subsequent job must be greater than the end time of the

preceding job. In view of this fact, the modified solution provides the best possible solution with minimum mean flow time for the jobs completed in each machine. The proposed heuristic mechanism modifies the starting time of the orders assigned to a machine and does not change the machine allotted to process each order.

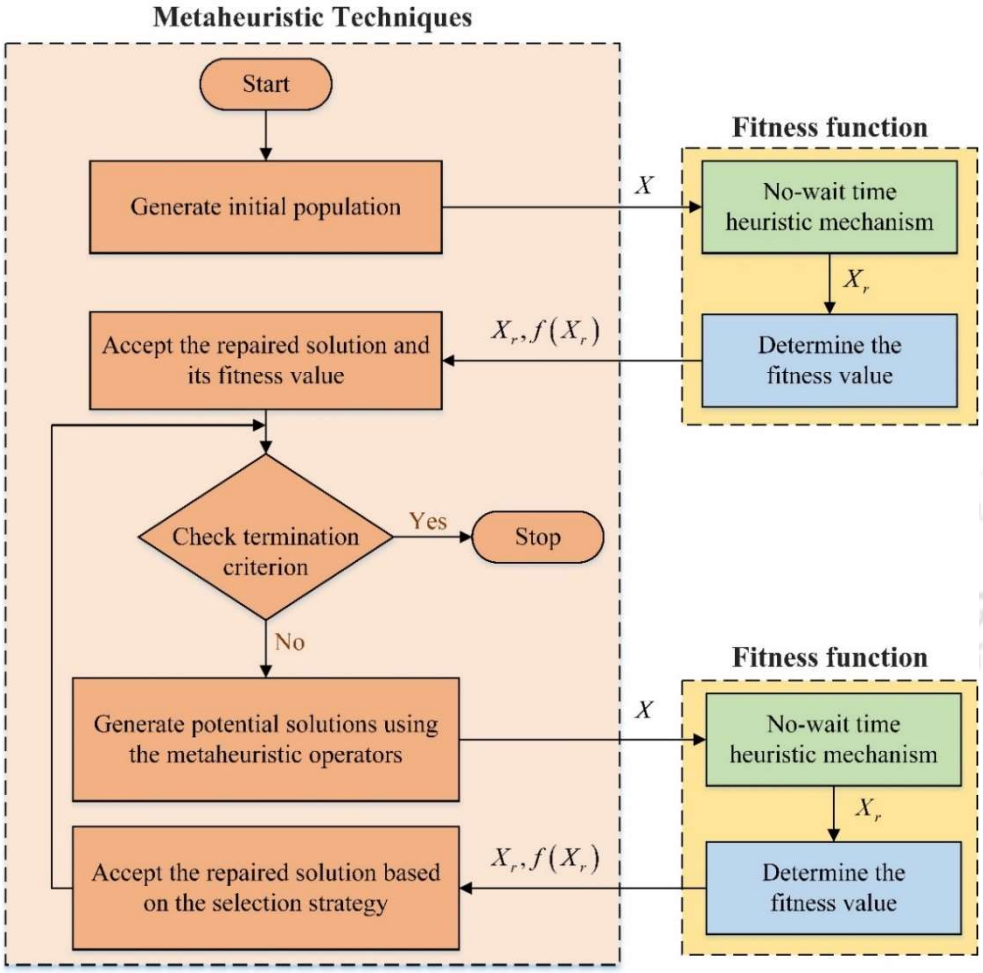


Fig. 6.3 Integration of metaheuristic technique, heuristic mechanism, and fitness function

In most of the potential schedules obtained from metaheuristic techniques, the starting time of the first order to be processed in a machine might not be its release date. As per the proposed heuristic mechanism, the starting time of the first order allotted to a machine is shifted to its release date, providing more time for processing subsequent orders. Similarly, the immediate processing of an order after completing its preceding order also minimizes the chance of

overlap constraint violation unless the end time of an order is greater than the upper bound of the starting time of its subsequent order. Thus, the proposed mechanism helps the metaheuristic technique in determining better solutions. Moreover, the proposed heuristic mechanism improves the quality of a feasible solution by reducing the idle time between subsequent jobs. Numerical examples of the scheduling problem with two machines and four jobs are provided in Appendix E to understand the proposed heuristic mechanism.

6.3. Experimental Settings

This study considered five different scheduling problem instances reported in Jain and Grossmann (2001), with varying levels of complexity. The problem with the lowest dimension constitutes three orders and two machines, and the problem with the highest dimension contains twenty orders and five machines. In addition, a high-dimensional problem with 100 jobs and 20 machines is also solved to test the performance of the proposed mechanism. The number of orders and machines for each problem is provided in Table 6.1.

Table 6.1 Details of the number of jobs and machines in each problem

Problem	P1	P2	P3	P4	P5	P6
Number of jobs	3	7	12	15	20	100
Number of machines	2	3	3	5	5	20

Each of the six problems (P1 – P6) has two different data sets that vary in the processing time required by the machines, whereas, for both data sets, the release date, the due date, and the processing cost of the orders are the same. The processing time required by all machines of data set 1 is longer than that of data set 2, and hence, a larger feasible region is available for data set 2 than data set 1. The resulting twelve problem instances are named according to the problem number and the dataset number. For example, a problem instance P4S1 indicates the fourth problem with the first dataset. All the required data corresponding to the twelve problems are provided in Appendix D.

This chapter considered fifteen metaheuristic techniques to study the efficiency of the proposed no-wait time heuristic mechanism and the impact of horizon constraints in determining a better schedule. The metaheuristic techniques considered in this study include Artificial Bee colony (ABC) (D. Karaboga, 2007), Adaptive Wind Driven Optimization (AWDO) (Bayraktar & Komurcu, 2016), Coyote Optimization Algorithm (COA) (Pierezan & Coelho, 2018), Chicken Swarm Optimization (CSO) (Meng et al., 2014), Dynamic Neighborhood Learning Particle Swarm Optimizer (DNLPSO) (Nasir et al., 2012), Grey Wolf Optimizer (GWO) (Mirjalili et al., 2014), Moth Flame Optimization (MFO) (Mirjalili, 2015), Multi-Population based Ensemble of mutation strategies Differential Evolution(MPEDE) (Wu et al., 2016), Multi-Verse Optimizer (MVO) (Mirjalili et al., 2016) , Salp Swarm Optimization (SSA) (Mirjalili et al., 2017), Sanitized Teaching Learning Based Optimization (STLBO), Simultaneous Heat Transfer Search (SHTS) (Maharana & Kotecha, 2016), Spotted Hyena Optimization (SHO) (Dhiman & Kumar, 2017), Symbiotic Organisms Search (SOS) (Cheng & Prayogo, 2014) and Yin-Yang Pair Optimization (YYPO) (Punnathanam & Kotecha, 2016). The salient features of these techniques are provided in Appendix A. This study used the publically available MATLAB codes of all the metaheuristic techniques.

The decision variables corresponding to the assigned machine and the starting time of each order are integer variables. However, the metaheuristic techniques considered in this study are fundamentally designed to solve optimization problems with continuous decision variables. Hence, the round-off scheme is utilized to incorporate the restriction of integer variables. The parameter settings of each metaheuristic technique used to solve the scheduling instances are provided in Table 6.2. The population size of the metaheuristic techniques is considered as ten times the number of orders. The only termination criterion considered for all the techniques is the maximum number of function evaluations, which is set to 20,000 times the number of

orders. In view of the stochastic nature of metaheuristic techniques, 30 independent runs of each technique are performed to solve problem instances.

Table 6.2 Details of algorithm-specific parameters

Algorithm	**Parameters**	**Value**
AWDO	Maximum allowed speed	0.3
COA	Probability of leaving the pack	$0.005(number\ of\ coyote)^2$
	Probability of a coyote birth	$1/dimension$
CSO	Percentage of roosters	0.15
	Percentage of hens	0.7
	Percentage of the mother hen	0.5
DNLPSO	Maximum and minimum inertia factor (w_0 and w_1)	0.9 and 0.4
	Acceleration coefficients ($c1$ and $c2$)	$1.49445r$
	Parameter for determination of velocity bound	2
	Refreshing gap	3
	Regrouping gap	5
MPEDE	Percentage of indicator population	0.2
	Generation gap	20
	Scaling factor	0.5
	Crossover probability	0.5
MVO	Minimum wormhole existence probability	1
	Maximum wormhole existence probability	0.2
YYPO	Bounds of the archive update interval	2 and 6
	Expansion/contraction factor	2
ABC	Limit	$popsize \times dimension$

This study has been implemented on a PC with an Intel i7 @4.2GHz processor and 32GB of RAM using MATLAB 2019a simulation environment. The average computational time required by each technique to solve all the scheduling instances presented in Jain and Grossmann (2001) in a single run ranges from approximately 2 to 28 seconds, depending on the complexity of the specific problem instance.

6.4. Results and discussion

This section firstly analyzes the benefits of the heuristic-assisted metaheuristic framework over the metaheuristic techniques alone in solving the scheduling instances. Secondly, the performance of each metaheuristic technique with the proposed framework is examined based on the statistical analysis of the obtained results. Lastly, the realizations determined by all metaheuristic techniques while solving each problem instance are also analyzed. The performance analysis on high dimensional problems is performed on instances with 100 jobs, and 20 machines are also studied. The results depicting the statistical analysis and the convergence analysis of the metaheuristic techniques are provided in Appendix E.

6.4.1. Benefits of the proposed framework

Before analyzing the efficacy of the proposed framework, the impact of horizon constraints on the performance of the metaheuristic techniques is analyzed. The benefits of the proposed framework are studied based on the best fitness value, the magnitude of the violation of constraints, and convergence analysis.

Impact of horizon constraints

A schedule that satisfies due date constraints and overlap constraints would not violate the horizon constraints. Even though the horizon constraints are redundant, they have been included in the literature (Jain & Grossmann, 2001) to serve as tightening constraints.

Table 6.3 provides the mean fitness determined by 15 techniques in all problems with (WH) and without (WoH) the horizon constraints. The values in which a technique obtained the same mean fitness value for solving WH and WoH problems are highlighted using bold typeface font. Out of ten problem instances, the mean fitness of SSA and SOS are identical in six instances, while only three are in CSO and DNLPSO. In P1S2 and P2S2, all techniques provided identical mean fitness.

Table 6.3 Mean fitness value of techniques with (WH) and without (WoH) horizon constraints

	ABC		AWDO		COA		CSO		DNLPSO	
	WoH	WH	WoH	WH	WoH	WH	WoH	WH	WoH	WH
P1S1	**26.0**	**26.0**	**26.0**	**26.0**	**26.0**	**26.0**	**26.0**	**26.0**	**27.1**	**27.1**
P1S2	**18.0**	**18.0**	**18.2**	**18.2**	**18.0**	**18.0**	**18.0**	**18.0**	**18.1**	**18.1**
P2S1	**60.0**	**60.0**	134.0	134.7	69.3	74.0	60.0	62.4	74.5	74.3
P2S2	**44.0**	**44.0**	**45.7**	**45.7**	**44.0**	**44.0**	**44.1**	**44.1**	**44.2**	**44.2**
P3S1	102.5	103.0	241.3	243.9	**102.5**	**102.5**	103.1	103.0	148.0	136.0
P3S2	**84.1**	**84.1**	**87.2**	**87.2**	83.2	83.0	83.4	83.5	84.2	84.1
P4S1	**118.1**	**118.1**	123.8	124.7	116.4	116.6	118.5	118.6	122.7	122.1
P4S2	104.1	104.2	104.6	104.7	**102.0**	**102.0**	102.3	102.4	106.4	106.5
P5S1	164.1	165.1	206.2	207.1	160.4	161.2	164.5	163.7	181.6	237.3
P5S2	143.5	143.7	**145.5**	**145.5**	140.3	140.2	**141.0**	**141.0**	147.2	147.4
	GWO		**MFO**		**MPEDE**		**MVO**		**SHO**	
	WoH	WH	WoH	WH	WoH	WH	WoH	WH	WoH	WH
P1S1	**26.0**	**26.0**	27.1	26.0	26.0	27.1	**26.0**	**26.0**	**26.0**	**26.0**
P1S2	**18.0**	**18.0**	**18.0**	**18.0**	**18.0**	**18.0**	**18.0**	**18.0**	**18.0**	**18.0**
P2S1	71.7	69.3	83.4	76.6	76.3	67.0	90.3	95.5	126.6	125.1
P2S2	**44.1**	**44.1**	**44.1**	**44.1**	**44.0**	**44.0**	**44.1**	**44.1**	**45.1**	**45.1**
P3S1	102.9	103.1	121.8	121.6	101.7	102.0	107.9	103.7	240.5	242.1
P3S2	83.7	83.8	**84.1**	**84.1**	**83.1**	**83.1**	84.0	83.4	87.3	87.1
P4S1	**118.3**	**118.3**	122.0	122.2	**115.8**	**115.8**	119.7	119.4	182.5	172.5
P4S2	103.2	103.1	**104.7**	**104.7**	**102.2**	**102.2**	**104.8**	**104.8**	**112.4**	**112.4**
P5S1	164.0	163.3	177.9	171.3	159.9	160.6	164.8	165.7	395.7	396.1
P5S2	**141.9**	**141.9**	144.2	143.8	140.3	140.4	144.0	143.3	**155.3**	**155.3**
	SHTS		**SOS**		**SSA**		**sTLBO**		**YYPO**	
	WoH	WH	WoH	WH	WoH	WH	WoH	WH	WoH	WH
P1S1	**26.0**	**26.0**	**26.0**	**26.0**	**26.0**	**26.0**	**26.0**	**26.0**	**26.0**	**26.0**
P1S2	**18.0**	**18.0**	**18.0**	**18.0**	**18.0**	**18.0**	**18.0**	**18.0**	**18.0**	**18.0**
P2S1	69.4	71.7	**60.0**	**60.0**	81.0	71.7	69.3	64.8	**60.0**	**60.0**
P2S2	**44.0**	**44.0**	**44.0**	**44.0**	**44.1**	**44.1**	**44.1**	**44.1**	**44.0**	**44.0**
P3S1	102.1	103.0	101.6	101.7	126.1	104.8	101.6	102.0	102.9	107.2
P3S2	83.4	83.5	83.1	83.0	**84.2**	**84.2**	83.3	83.4	83.5	83.6
P4S1	118.3	119.3	115.5	115.4	120.3	120.8	115.7	115.6	118.6	118.5
P4S2	103.0	103.1	**102.4**	**102.4**	104.3	104.5	**102.3**	**102.3**	**103.9**	**103.9**
P5S1	165.6	167.6	160.1	160.6	**175.3**	**175.3**	160.4	160.6	166.2	167.0
P5S2	141.7	141.8	**140.6**	**140.6**	**144.2**	**144.4**	140.7	140.5	143.5	143.6

The two-sided Wilcoxon signed-rank test is performed to identify if the mean fitness of the problems with and without horizon constraints are statistically different with a significance level of 5% ($\alpha = 0.05$). The p-values provided by each sample are provided in Table 6.4.

Table 6.4 p-value of Wilcoxon signed-rank test

Technique	*p*-value	Technique	*p*-value	Technique	*p*-value
ABC	0.06	GWO	0.48	SHTS	**0.02**
AWDO	**0.03**	MFO	0.05	SOS	1.00
COA	0.34	MPEDE	0.56	SSA	1.00
CSO	0.84	MVO	1.00	sTLBO	0.63
DNLPSO	0.81	SHO	0.81	YYPO	0.25

The p-value for every technique except for AWDO and SHTS is greater than the significance level of 0.05, which indicates that the test failed to reject the null hypothesis. In the case of AWDO and SHTS, the Wilcoxon test has rejected the null hypothesis. However, it is evident from Table 6.3 that the mean fitness of these two techniques without horizon constraints is better than the instances including these constraints. The performance analysis of the techniques has revealed that the exclusion of horizon constraints from the formulation of scheduling problems has no impact on determining the optima. Hence, in the rest of the analysis, the horizon constraints are excluded from instances, reducing their complexity by *M* number of constraints.

Benefits of no-wait time heuristic mechanism

The scheduling instances are solved with and without including the repair mechanism with the metaheuristic technique to analyze its effectiveness on the performance of metaheuristic techniques. The best fitness value determined by metaheuristic techniques with repairing (WR) and without repairing (WoR) the schedule using the proposed mechanism is provided in Fig. 6.4, and the corresponding worst, mean, median, and standard deviation of the fitness values are available in Appendix E.

The best fitness value determined by each technique with the heuristic mechanism is either better or equal to the solutions determined without the heuristic mechanism. In SHO, the

benefit of the heuristic mechanism is evident as it has determined a better schedule in all the problems with the heuristic mechanism. As the problem complexity increases, the techniques improve their performance using the heuristic mechanism. It is also evident that with the increase in problem complexity, none of the techniques identified the optima in any of the problems without the heuristic mechanism. In every problem except P5S1, the best value determined by COA, CSO, GWO, MPEDE, and sTLBO is the minimum among all the metaheuristic techniques. In all the problem instances, SOS has determined the best value compared to all the techniques in at least one run with the help of the heuristic mechanism.

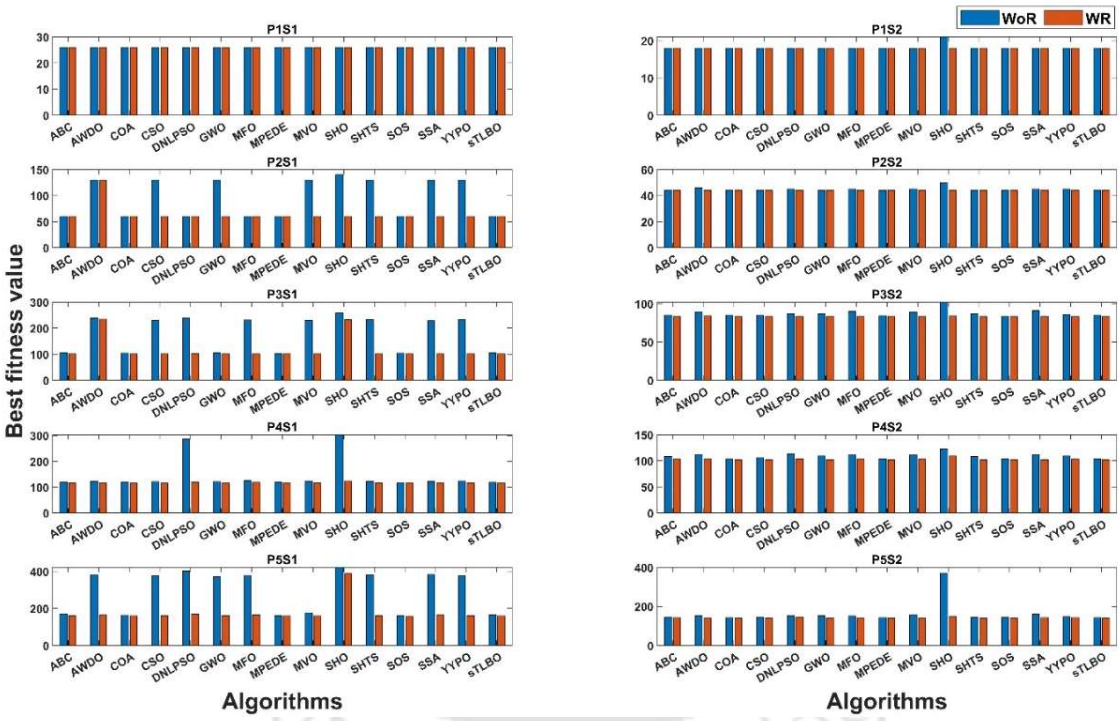


Fig. 6.4 Best fitness value of all techniques with and without the heuristic mechanism

The violation corresponding to the best schedule determined by each of the techniques is provided in Fig. 6.5. The best schedules for P1S1, P1S2, P2S2, P3S2 and P4S2 determined by all the techniques are feasible and hence are excluded from Fig. 6.5. In the absence of the heuristic mechanism, techniques such as AWDO, CSO, DNLPSO, GWO, MFO, SHO, SHTS SSA, and YYPO are unable to identify even one feasible solution among all runs in the most complicated problem (P5S1).

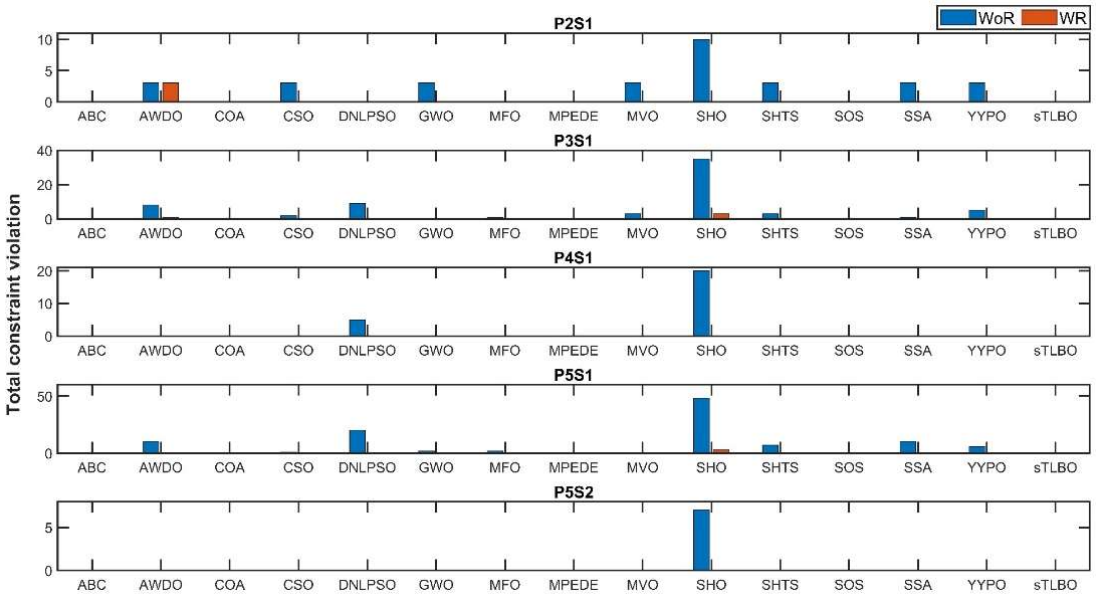


Fig. 6.5 Total constraint violation in the best schedule determined by techniques

The mean convergence of the most complex problem, P5S1, for all the techniques is provided in Fig. 6.6. The quick convergence and better final solution corresponding to each technique with the no-wait time heuristic depict the benefits of the proposed mechanism.

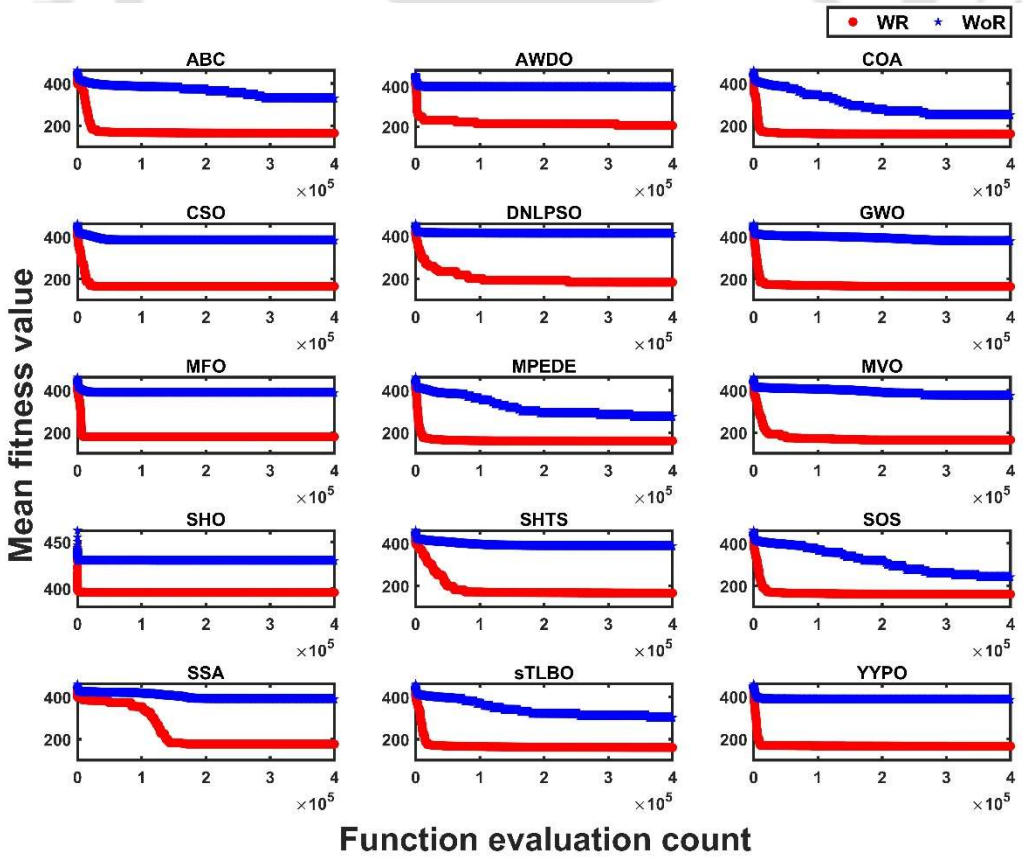


Fig. 6.6 Mean convergence of techniques with and without heuristic mechanism in P5S1

Every technique with the heuristic operator has converged to a feasible mean fitness value except SHO. In contrast, all the metaheuristic techniques without the heuristic mechanism could not obtain a feasible mean fitness, indicating the determination of infeasible solutions in at least one run. The majority of the techniques with the heuristic mechanism, except AWDO and DNLPSO, have reached the final mean value within half of the allowed function evaluations.

Feasibility analysis

The benefit of employing the heuristic mechanism with the metaheuristic technique based on the quick convergence of these techniques to a better solution is studied by analyzing the best solution obtained by SOS (best technique) while solving P5S1. The convergence profile of fitness value and the constraint violation with respect to the function evaluations are provided in Fig. 6.7.

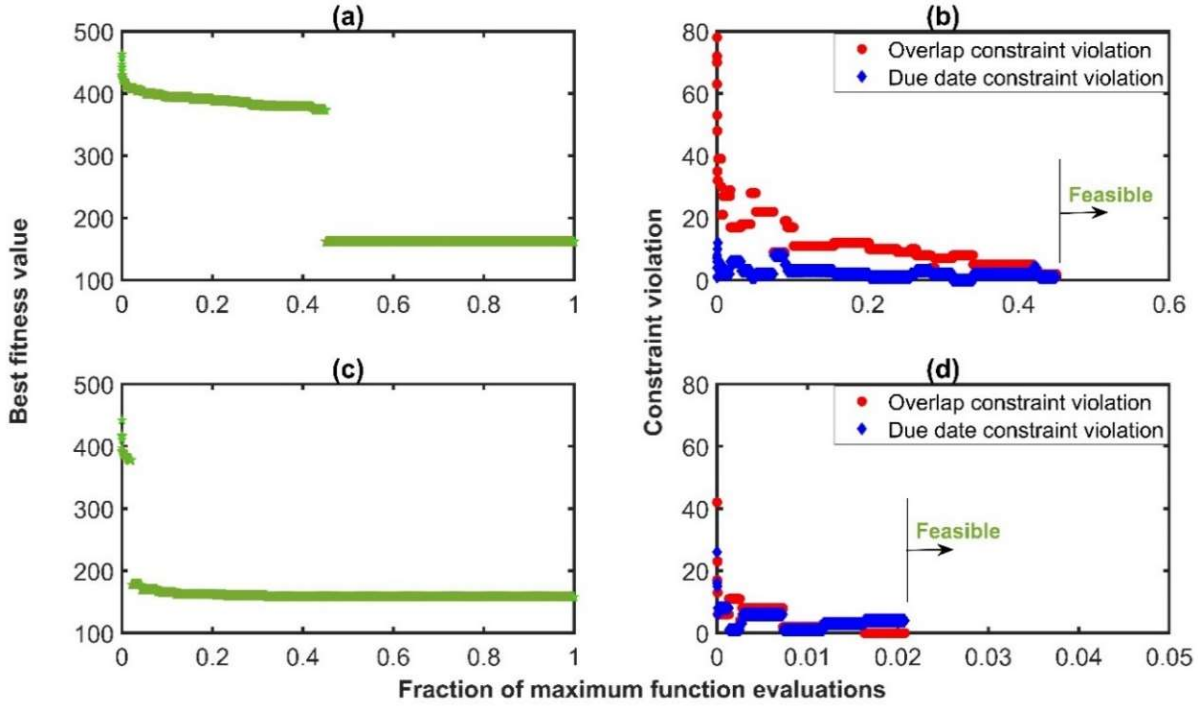


Fig. 6.7 Convergence of best fitness value and violation of constraints for P5S1 without heuristic mechanism ((a) and (b)) and with heuristic mechanism ((c) and (d))

It is evident from comparing Fig. 6.7 (a) and Fig. 6.7 (c) that the heuristic mechanism helped the metaheuristic technique converge to a better fitness value in its earlier iterations. It is also observed from Fig. 6.7 (b) and Fig. 6.7 (d) that with the help of the heuristic mechanism, SOS identified a feasible solution within 2% of the maximum function evaluation. At the same time, without the heuristic mechanism, SOS determined the feasible solution only after using 40% of the maximum function evaluation.

6.4.2. Statistical analysis of results provided by the proposed framework

The best (B), mean (M), and median (Md) statistics of the fitness value determined by the techniques in each problem are provided in Table 6.5. This study helps to compare the ability of metaheuristic techniques to determine better solutions.

The best value corresponding to each statistic across a technique is highlighted with a bold typeface. The major inferences from the statistical table are provided below:

- SOS has determined the best fitness in all the problems, followed by COA, GWO, MPEDE, and sTLBO, which determine the optima except in P5S1. For the problem instances P1S1-P2S2, all techniques have determined the optimal schedule in at least one run, except AWDO in P2S1.
- The mean fitness of MPEDE is better than that of other techniques, followed by SOS. The difference between mean and best fitness is minimum in SOS except in P5S1. A similar trend is observed in MPEDE and sTLBO except in P2S1 and P5S1.
- The median of the fitness determined by MPEDE is better than other techniques except in P2S1 and P4S2, followed by COA. The median value determined by sTLBO in eight problem instances is equal to the best fitness value. A similar observation is observed in seven problems with COA, MPEDE, and SOS.

Table 6.5 Statistical analysis of the metaheuristic techniques with the heuristic mechanism

		ABC	AWDO	COA	CSO	DNLPSO	GWO	MFO	MPEDE	MVO	SHO	SHTS	SOS	SSA	sTLBO	YYPO
P1S1	B	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**
	M	**26**	**26**	**26**	**26**	27.07	**26**	27.07	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**
	Md	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**
P1S2	B	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**
	M	**18**	18.2	**18**	**18**	**18.10**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**
	Md	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**
P2S1	B	**60**	130	**60**	**60**	**60**	**60**	**60**	**60**	**60**	**60**	**60**	**60**	**60**	**60**	**60**
	M	**60**	133.97	69.33	**60**	74.47	71.73	83.40	76.33	90.33	126.60	69.43	**60**	81	69.33	**60**
	Md	**60**	134	**60**	**60**	**60**	**60**	**60**	**60**	**60**	133.50	**60**	**60**	**60**	**60**	**60**
P2S2	B	**44**	**44**	**44**	**44**	**44**	**44**	**44**	**44**	**44**	**44**	**44**	**44**	**44**	**44**	**44**
	M	**44**	45.70	**44**	**44.1**	44.17	44.10	44.07	**44**	44.07	45.13	**44**	**44**	44.13	44.10	**44**
	Md	**44**	46	**44**	**44**	**44**	**44**	**44**	**44**	**44**	45	**44**	**44**	**44**	**44**	**44**
P3S1	B	**101**	234	**101**	**101**	101	**101**	**101**	**101**	**101**	233	**101**	**101**	**101**	**101**	**101**
	M	102.53	241.27	102.50	103.13	148.03	102.87	121.80	**101.73**	107.87	240.53	102.13	101.63	126.13	**101.63**	102.87
	Md	103	242	102	103	108	103	105	**101**	103	241	102	102	105.50	**101**	103
P3S2	B	**83**	84	**83**	**83**	**83**	**83**	**83**	**83**	**83**	84	**83**	**83**	**83**	**83**	**83**
	M	84.07	87.17	83.17	83.43	84.20	83.73	84.13	**83.10**	83.97	87.30	83.40	**83.07**	84.20	83.33	83.53
	Md	84	87	**83**	**83**	84	84	84	**83**	84	87.50	**83**	**83**	84	**83**	**84**
P4S1	B	116	117	**115**	**115**	120	**115**	118	**115**	116	124	116	**115**	116	**115**	116
	M	118.13	123.80	116.43	118.47	122.70	118.27	121.97	**115.80**	119.67	182.47	118.33	**115.53**	120.33	115.70	118.60
	Md	118	123.50	**116**	118	122.50	119	122	**116**	120	133.50	118	**115**	120.5	**115**	119
P4S2	B	103	103	**102**	**102**	104	**102**	103	**102**	103	109	**102**	**102**	**102**	**102**	103
	M	104.1	104.57	**102**	102.33	106.40	103.20	104.70	102.20	104.77	112.40	103.03	102.43	104.27	102.33	103.90
	Md	104	104	**102**	**102**	106	103	105	**102**	105	112.50	103	**102**	104	**102**	104
P5S1	B	161	164	159	160	169	161	165	159	159	388	161	**158**	164	159	161
	M	164.07	206.23	160.37	164.47	181.5667	163.97	177.87	**159.93**	164.77	395.70	165.63	160.13	175.27	160.43	166.23
	Md	164	171	**160**	164	174.50	164	171	**160**	165	396	165	**160**	168	160	166
P5S2	B	142	141	**140**	**140**	145	**140**	141	**140**	141	150	**140**	**140**	142	**140**	142
	M	143.53	145.53	**140.27**	141.03	147.23	141.90	144.17	**140.27**	144	155.30	141.67	140.63	144.17	140.70	143.53
	Md	144	145	**140**	141	146.5	142	144	**140**	143.50	155	141.50	141	144	141	144

6.4.3. Analysis of best schedule

This section analyzes the multiple realizations of the best schedule, and the Gantt chart depicts the best schedule, as given below.

Realizations of optima

Realizations are solutions with different values for the decision variables but have the same fitness value. This section analyzes the realizations of the optima obtained for solving the problem instances using metaheuristic techniques and incorporating the heuristic mechanism. The metaheuristic techniques with the proposed mechanism are able to determine multiple realizations of the optima in all problems except P1S1. The number of realizations obtained in each problem is provided in Table 6.6.

Table 6.6 Number of realizations obtained using heuristic-assisted metaheuristic technique.

Problem	**P1S1**	**P1S2**	**P2S1**	**P2S2**	**P3S1**	**P3S2**	**P4S1**	**P4S2**	**P5S1**	**P5S2**
No. of realizations	1	2	2	32	20	142	44	101	2	64

It is observed that the number of realizations corresponding to the problems with the first dataset is always lesser than the problem with the second dataset. This is primarily due to the large feasible space in problems with the second dataset. The heuristic mechanism helps identify a solution with the minimum idle time among the orders allotted to a machine. As per the proposed heuristic mechanism, idle time arises in a machine only when the release date of the order is higher than the ending time of its previous order. However, there might be many more realizations of the optima possible with a machine being idle between multiple orders.

This study obtained multiple realizations of the optima in all problems solved using the metaheuristic techniques with and without the heuristic mechanism. The number of realizations of each problem identified by a technique with and without using the heuristic mechanism is provided in Fig. 6.8, and a table representing the same is provided in Appendix E. SOS, with

the help of the heuristic mechanism, has determined the realizations of the optima in all the problems. Similarly, MPEDE and sTLBO, with the help of the heuristic mechanism, are able to determine realizations in all problems except in P5S1.

As per the proposed heuristic mechanism, an idle time arises in a machine only when the release date of the order is higher than the ending time of its previous order. However, there might be many more realizations of the optima possible with a machine being idle between multiple orders. Such a scenario can be observed in P1S1 and P1S2, where the number of realizations identified by all metaheuristic techniques without the heuristic mechanism is more than with the heuristic mechanism. In all other problems, the total number of realizations identified by all the metaheuristic techniques with the help of the heuristic mechanism is more than the techniques without the proposed mechanism.

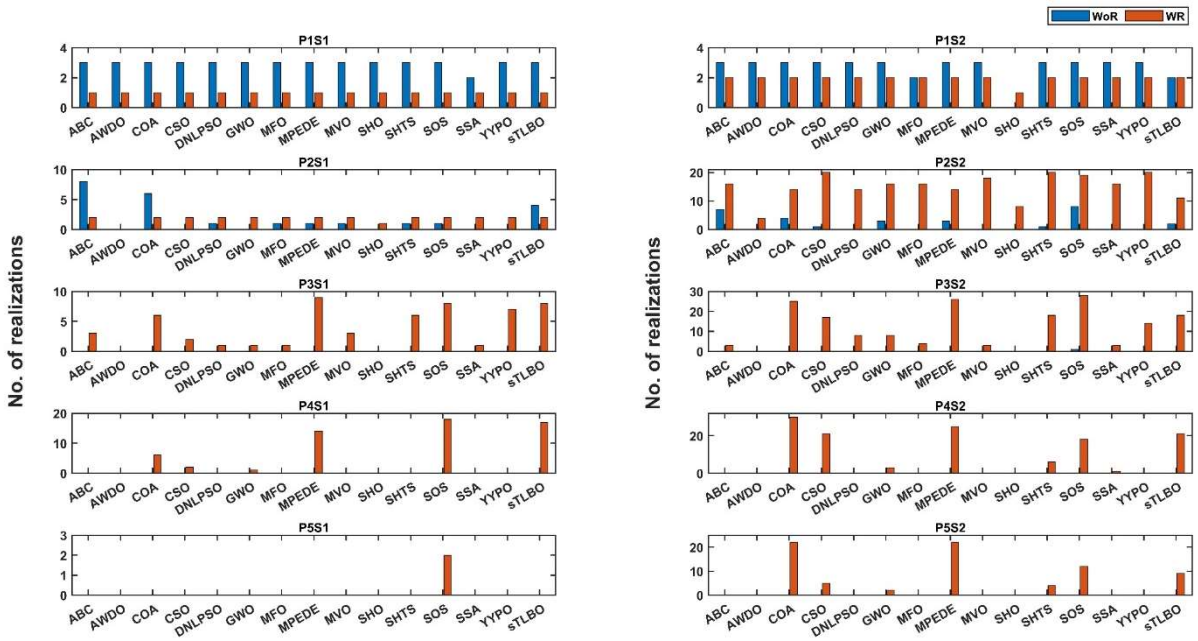


Fig. 6.8 Number of realizations obtained by metaheuristic techniques with and without the heuristic mechanism

The Gantt chart of the two realizations of P5S1 determined by SOS with the heuristic mechanism is provided in Fig. 6.9. In both solutions, all the available machines are utilized to process the orders, and the idle time is zero between two consecutive orders. It is observed that

the set of jobs processed on a machine differ among the realizations and did not incur any additional cost. For example, from Fig. 6.9, it can be observed that in the first schedule, M_1 processes J_7, J_5 and J_1, while in the second schedule, M_1 processes all orders except J_5, which is processed by M_3. In multiple realizations, even though a particular machine processes the same set of orders, the sequence of their processing can be different. Such a scenario can be observed in M_2 of both the schedules provided in Fig. 6.9.

Multiple realizations help to select a schedule that is appropriate to other requirements. The multiple realizations can be used to accommodate higher-level information on the problem. For example, the processing cost of both the schedules provided in Fig. 6.9 is 158, whereas the total time elapsed to complete all orders (makespan) of the first realization is 38, and for the second realization is 37. Therefore, the user interested in minimum processing cost and minimum makespan would prefer the second realization.

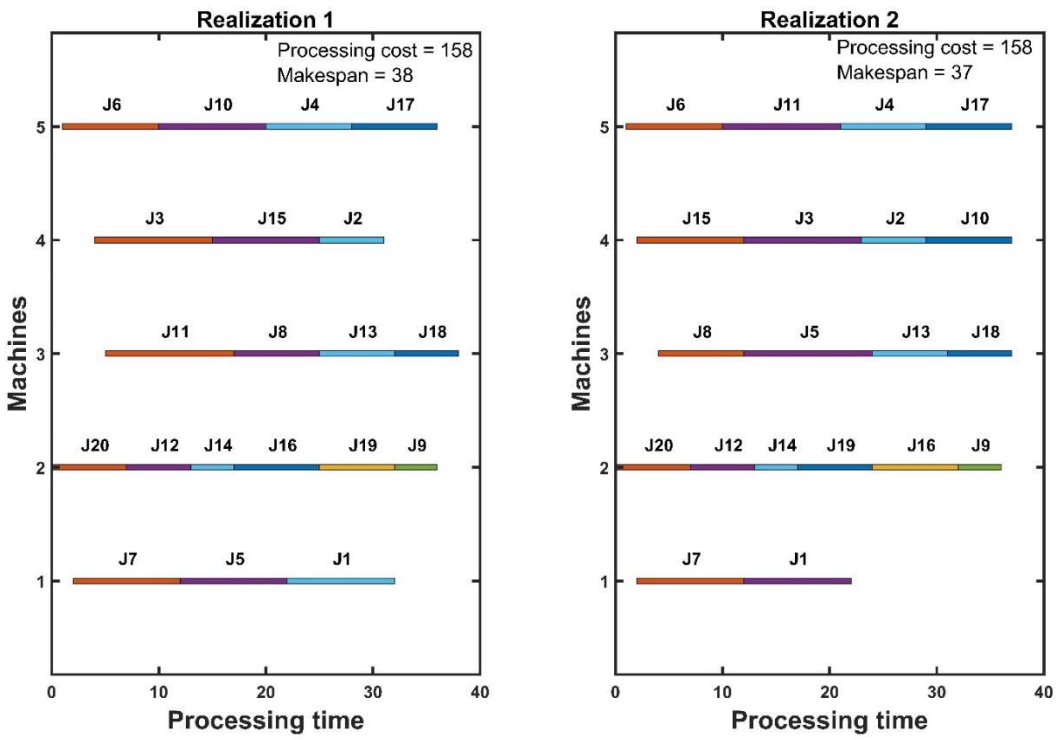


Fig. 6.9 Realizations of P5S1

The realizations determined by SOS with and without the heuristic mechanism in P3S2 is provided in Fig. 6.10. In both solutions, the same number of jobs are allotted to the second and third machines. The first machine is kept idle as it is the most costlier machine compared to the other two machines.

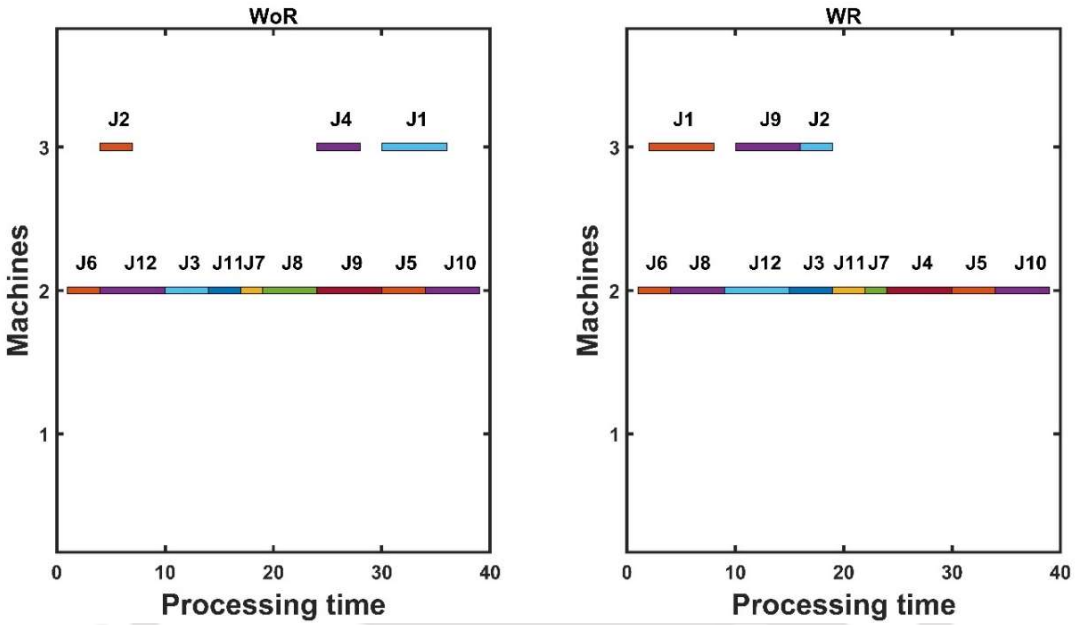


Fig. 6.10 Realizations of P3S2 determined by SOS with and without the heuristic mechanism

From Fig. 6.10, it is observed that switching the machines employed for processing jobs J_4 and J_9 has not affected the fitness value. The solution identified without the help of the heuristic mechanism possesses more idle time than the solution determined by employing the heuristic mechanism. In the solution determined using the heuristic mechanism, the release date of J_9 is higher than the end time of J_1, because of which an idle time exists between J_1 and J_9. The use of the proposed heuristic mechanism has helped reduce idle time among the orders on a machine and can be counted as a reason for identifying more realizations.

6.5. Conclusion

This chapter proposed a no-wait time heuristic mechanism to reduce the constraint violation of a potential schedule and assist metaheuristic techniques in determining better solutions for solving the scheduling problems. The rescheduling of processing the orders on a machine to

the earliest possible time is observed to be beneficial, as it provides more time for completing the succeeding orders assigned to a machine. The inclusion of the proposed heuristic mechanism does not require any modification in the metaheuristic techniques and can be easily integrated with other metaheuristic techniques. The no-wait time heuristic mechanism helped fifteen metaheuristic techniques improve their convergence and provide better schedules in all the problem instances in the literature compared to the techniques without employing the proposed mechanism. The schedules determined by SOS outperform the other techniques in all the problem instances of the literature, whereas YYPO and COA with the heuristic mechanism determined better schedules than other techniques for the high-dimensional problem instances. The benefits provided by the proposed mechanism in determining optimal schedules imply its suitability with metaheuristic techniques. A detailed solution analysis identified multiple realizations for each problem instance, and the proposed heuristic mechanism helped determine more realizations for the problems with a large number of machines and orders. Additionally, this study analyzed the effect of horizon constraints on the performance of metaheuristic techniques, which establishes that these constraints do not have a significant impact and that the removal of these constraints reduces computational complexity.

Chapter 7

Multi-objective model for scheduling jobs on dissimilar parallel machines solved using metaheuristic techniques

A schedule that achieves the fastest completion time or makespan holds immense significance as it directly indicates operational efficiency and effective resource utilization. Hence, obtaining an optimal schedule with minimum makespan and processing cost is crucial. This chapter extended the single objective scheduling problem discussed in the previous chapter to a multi-objective model by considering two conflicting objectives – minimization of processing cost and minimization of makespan. A new multi-objective variant of teaching learning-based optimization named Non-dominated Sorting phase-wise Teaching Learning Based Optimization (NSpTLBO) is proposed to solve the multi-objective scheduling problem. The performance of the optimization framework to solve the multi-objective model is analyzed in solving eight problem instances using multiple metaheuristic techniques. The chapter has been structured as follows: Section 7.1 provides the multi-objective scheduling problem, and Section 7.2 briefly describes the proposed multi-objective variant of PTLBO. The various parameter settings and details of the problem instances considered in this chapter are discussed in Section 7.3. In Section 7.4, a detailed depiction and analysis of the results obtained by the metaheuristic techniques are provided, and finally, Section 7.5 concludes the chapter by providing the significant findings of this study.

7.1. Multi-objective scheduling problem

The scheduling problem studied in Chapter 6 is extended to solve the multi-objective problem by considering the minimization of processing cost and minimization of makespan as the objectives subjected to the overlap constraints, release date constraints, and due date constraints. The no-wait time heuristic mechanism is used to modify a potential solution provided by the multi-objective metaheuristic techniques before evaluating its fitness.

The fitness value corresponding to the total processing cost (f_{TC}) is calculated in the same way as given in Eq. (6.11). The makespan corresponding to a potential solution is determined using Eq. (7.1).

$$Min.\ f_{MS} = \left(\max_{i\in\{1,2,..,I\}} \left(end_i\right) - \min_{i\in\{1,2,..,I\}} \left(t_i\right) \right) + P^{Total} + \lambda \max_{i\in\{1,2,..,I\}} \left(d_i\right) \tag{7.1}$$

In Eq. (7.1), end_i and t_i represent the end time and start time of an order processed in the allotted machine, respectively. Adding the maximum due date to the makespan of an infeasible schedule helps distinguish a feasible solution from an infeasible one. The parameter (λ) indicates whether a potential schedule is feasible, which is determined as Eq. (7.2).

$$\lambda = \begin{cases} 0 & if\ P^{Total} = 0 \\ 1 & otherwise \end{cases} \tag{7.2}$$

The constraint handling technique employed in this study is adapted from Deb's approach (Kalyanmoy Deb, 2001), which is based on selecting feasible solutions over infeasible ones. The fitness value of a solution consists of three terms (i) the objective function value (total cost or makespan), (ii) the total constraint violations, and (iii) a penalty term. The absence of the third penalty term might lead to selecting an infeasible solution with a lower fitness value over a feasible solution. The value of the parameter λ assures that the penalty term is added only to the objective value of an infeasible solution. Using a penalty function with a penalty coefficient is another way of selecting the feasible solution over the infeasible one. However, the selection of the penalty coefficient is not trivial and hence avoided in Eq. (6.11) and Eq. (7.1).

7.2. Non-dominated Sorting Phase Teaching Learning Based Optimization

The multi-objective optimization problem constitutes two or more conflicting objective functions that have to be maximized/minimized and subjected to equality, inequality constraints, and bound constraints on the decision variables. The mathematical representation of a multi-objective optimization model can be represented as

$$
\begin{aligned}
&Min \;\; f_m(x) \qquad m = 1,2,...,M \\
&subject\ to \\
&\qquad g_j(x) \geq 0 \qquad j = 1,2,...,J \\
&\qquad h_k(x) = 0 \qquad k = 1,2,...,K \\
&\qquad x_d^L \leq x_d \leq x_d^U \qquad d = 1,2,...,D
\end{aligned} \tag{7.3}
$$

In the above equation, M represents the total number of conflictive objective functions, J and K indicate the number of equality and inequality constraints, respectively. D is the number of decision variables. It is impossible to determine a single optimum solution with respect to all the objective functions involved in a multi-objective model due to their conflicting nature. Hence, in multi-objective optimization, a set of solutions called the Pareto optimal set is identified. Each solution in the set is not worse in at least one objective value compared to the other solutions present in the Pareto optimal set and is denoted as non-dominated solutions. The concept of domination among solutions is elucidated using an example involving two solutions where the comparison is based on all the objective values. Consider two solutions $x^{(1)}$ and $x^{(2)}$, with the objective function values of $f\left(x^{(1)}\right)$ and $f\left(x^{(2)}\right)$. The conditions that have to be satisfied for the solution $x^{(1)}$ to dominate the solution $x^{(2)}$ $\left(f\left(x^{(1)}\right) \preceq f\left(x^{(2)}\right)\right)$ for minimization of objectives are given as

(i) Solution $x^{(1)}$ is not worse than $x^{(2)}$ in all the objectives

i.e., $f_m\left(x^{(1)}\right) \leq f_m\left(x^{(2)}\right) \qquad for\ all\ m = 1,2,...,M$

(ii) Solution $x^{(1)}$ is better than $x^{(2)}$ in at least one objective

i.e., $f_m\left(x^{(1)}\right) < f_m\left(x^{(2)}\right) \qquad for\ at\ least\ one\ m = 1,2,...,M$

If the solutions $x^{(1)}$ and $x^{(2)}$ do not dominate each other, then those solutions are considered as non-dominating solutions. i.e., $f\left(x^{(1)}\right)$ is not worse than $f\left(x^{(2)}\right)$ in at least one objective

function and vice versa. The trade-off among all non-dominated solutions in the objective space is termed the Pareto front.

The metaheuristic techniques are independent of the shape or continuity of the Pareto front, like the mathematical programming techniques, and hence suitable solvers for determining the Pareto optimal set (Coello et al., 2004). Moreover, most metaheuristic techniques are population-based algorithms and can determine multiple solutions of the Pareto front in a single run. In this study, we have utilized Phase-wise Teaching Learning Based Optimization (PTLBO), a variant of Teaching Learning Based Optimization (TLBO), to solve the multi-objective scheduling problem by employing the non-dominated sorting algorithm and crowding distance method to obtain the Pareto solutions. A detailed description of the proposed multi-objective variant of PTLBO is given in this section. Phase Teaching Learning Based Optimization (PTLBO) is a variant of TLBO in which a phase-wise generation of the potential solutions and population update is performed. The whole population (N) performs the teacher phase and generates new solutions using Eq. (7.4).

$$X_i^{new} = X_i + r\left(X_i^{best} - T_f^i X_i^{mean}\right) \tag{7.4}$$

In the single objective PTLBO, the population in the teacher phase is updated using the $(\mu + \lambda)$ updating strategy by combining the N newly generated solutions with the population that has undergone the teacher phase. The best N solutions are selected and will perform the learner phase. In the learner phase of PTLBO, two potential solutions are generated using Eq.(7.5) and Eq. (7.6). Here S_i^1 and S_i^2 are the two potential solutions generated from the solution undergoing the learner phase (X_i), and the randomly selected partner solution (X_p) for the i^{th} solution.

$$S_i^1 = X_i + r\left(X_i - X_p\right) \tag{7.5}$$

$$S_i^2 = X_i + r\left(X_p - X_i\right) \tag{7.6}$$

The population for the next iteration is selected from the learner phase using the $(\mu+\lambda)$ updating strategy. The best *N* solutions are selected from the combined population of 2*N* new potential solutions and the current population members.

The multi-objective variant of the PTLBO algorithm, named Non-dominated Sorting phase Teaching Learning Based Optimization (NSpTLBO), generates new solutions using the variation operators of the PTLBO. However, the $(\mu+\lambda)$ updating strategy could not be applied directly to select the best *N* solutions, as in the multi-objective model, two or more conflicting objectives are involved, and a solution could not be best in all the objectives. Hence, NSpTLBO uses the non-dominated sorting and crowding distance measures to select solutions based on multiple objectives. The population is updated after generating new solutions from the whole population in both phases. The current population is combined with the newly generated solutions, and then each solution is assigned dominance and diversity indices, which help select the required number of best solutions. During each population update, only a predefined number of solutions are selected as the population for the next iteration. Hence, the possibility of discarding certain Pareto solutions cannot be ignored. In this regard, the NSpTLBO algorithm uses an archive to store the first front solutions determined in each iteration. The archive is updated iteratively at the end of each iteration by selecting the first front solutions among the previously archived solutions and the updated population of the learner phase. In the archive update stage, only the dominance measures are applied for the selection procedure, as the objective of the archive is to preserve all the non-dominated solutions determined by the algorithm. The procedure followed in NSpTLBO to determine the non-dominated solutions and their respective fitness is outlined in Fig. 7.1.

Input: Number of decision variables (D), lower bound (lb) and upper bound (ub) of decision variables, population size (N), number of objective functions (M), archive size (A_s), number of iterations (T)

Output: Pareto solutions (PS) and their objective function values (PF)

Generate the initial population (P) of size $(N \times D)$ within the bounds of decision variables.

Evaluate the objective function of the initial population (f).

Perform non-dominated sorting and crowding distance and update the archive population:

$[A, f_A] = Archive\ update\ (P, f, A_s)$

For $t = 1$ to T

> Perform the teacher phase: $[P, f] = Teacher\ phase\ (P, f, A, f_A, N)$
>
> Perform the teacher phase: $[P, f] = Learner\ phase\ (P, f, N)$
>
> Add the updated population with the archive population and assign the combined population and fitness as *combinedArcPop* and *combinedArcFit*, respectively
>
> Perform non-dominated sorting and crowding distance to update the archive population: $[A, f_A] = Archive\ update\ (combinedArcPop,\ combinedArcFit,\ A_s)$

End

Assign $PS = A$ and $PF = f_A$

Return *PS and PF*

Fig. 7.1. Pseudocode of NSpTLBO

The pseudocode of the teacher phase employed in the proposed NSpTLBO is provided in Fig. 7.2. For each solution, the best solution (teacher) required for generating a new solution is selected from the archive solutions. The variables of the newly generated solutions that have violated the bound constraints are handled by shifting them to the nearest violated bound, which is also specified in Fig. 7.2. The population is updated after evaluating all new solutions generated in the teacher phase. Non-dominated sorting method and crowding distance measure are used to obtain a converged and diversified population before proceeding to the learner phase.

Input: P, f, A, f_A, N

Output: Updated P, f

Randomly assign teacher $\left(X^{best}\right)$ for each solution in the population from the archive, A.

Determine the mean of the population $\left(X^{mean}\right)$.

Determine the teaching factor $T_f = round\left(1+r\right)$, where $r \in \left[0,1\right]$ for each of the solutions in the population, P.

Determine the new solution using the teacher phase as $P^{Tnew} = P + r\left(X^{best} - T_f X^{mean}\right)$.

Bound the newly generated solutions, $P^{Tnew} = \min\left(\max\left(P^{Tnew}, lb\right), ub\right)$

Evaluate all the new solutions generated in the teacher phase $\left(f\left(P^{Tnew}\right)\right)$.

Combine the newly generated solutions and current population, and assign as *combinedPop* and their fitness as *combinedFit*

Perform non-dominated sorting and crowding distance to update the population:

[P, f] = *Population update* (*combinedPop, combinedFit, N*)

Return P and f

Fig. 7.2. Pseudocode of the teacher phase

A solution in the learner phase is generated with the help of a partner solution, which is selected randomly from the population. Two new solutions are generated using the same partner solution in the learner phase. It is not necessary for each solution in the population to get a unique partner solution. The same bounding strategy of the teacher phase is employed in the learner phase to handle the bound violated variables. After evaluating the newly generated solutions, they are combined with the current population and employ the non-dominated sorting and crowding distance to update the population for the next iteration. The detailed pseudocode of the learner phase is given in Fig. 7.3.

Input: P, f, N

Output: Updated P, f

Randomly select a partner (X_p) for each solution in the population

Determine the new solution using the learner phase as $P_1^{Lnew} = P + r\left(P - X_p\right)$ $P_2^{Lnew} = P + r\left(X_p - P\right)$

Bound the newly generated solutions, $P^{Lnew} = \min\left(\max\left(P^{Lnew}, lb\right), ub\right)$

Evaluate all the new solutions generated in the teacher phase $\left(f\left(P^{Lnew}\right)\right)$.

Combine the newly generated solutions and current population, and assign as *combinedPop* and their fitness as *combinedFit*

Perform non-dominated sorting and crowding distance to update the population:

[P, f] = *Population update* (*combinedPop, combinedFit, N*)

Return P and f

Fig. 7.3. Pseudocode of the learner phase

The non-dominated sorting method and the crowding distance measure are used to update the population after each phase. It is not necessary that all the solutions in the updated population are present on the first front. However, the archive includes only the first front solutions representing the converged as well as diversified solutions identified till the current iteration. The population size restricts the number of solutions selected for the next iteration. Similarly, archive size restricts the set of first front solutions selected for the archive. Initially, the combined population is sorted based on their dominance into multiple fronts. The whole solutions in each front in their ascending order are considered for the updated population until a set of solutions less than the size of the last potential front. In that instance, the required number of solutions are selected by using the crowding distance measure. The pseudocode for the population is provided in Fig. 7.4.

Input: *combinedPop*, *combinedfit*, *N*

Output: Updated *P*, *f*

Sort the *combinedPop* based on the level of non-domination using the non-dominated sorting method and assign as *NSPop*.

L = rank of the last front solutions in the sorted *NSPop*

Initialize the number of solutions in the population for the next iteration, $n = 0$, and the next iteration population, $P = [\ ]$ and their fitness, $f = [\ ]$.

For $k = 1$ to L

  Q = number of solutions in the k^{th} front of *NSPop*

  If $N > n$

    Number of more solutions to be selected for the updated population, $S = N - n$

    If $S \geq Q$

      Add all the Q solutions in the k^{th} front of *NSPop* to P and their corresponding fitness to f.

      $n = n + Q$

    Else

      Determine the crowding distance of all k^{th} front solutions in *NSPop*.

      Sort the k^{th} front solutions of *NSPop* in the descending order of their crowding distance and assign as *NSCDPop*.

      Select the first S number of solutions from the *NSCDPop* and add to P and their corresponding fitness to f.

      $n = n + S$

    End If

  End If

End For

Return P and f

Fig. 7.4. Pseudocode of population updating procedure

In archive updating stage, the updated population at the end of an iteration and the previous archive are considered. Unlike the population update, which occurs twice in an iteration, the

archive is updated only once in an iteration. The detailed pseudocode describing the archive updating procedure is provided in Fig. 7.5, respectively.

Input: $combinedArcPop, combinedArcfit, A_s$

Output: Updated A, f_A

Sort the *combinedArcPop* based on the level of non-domination using the non-dominated sorting method and assign as *NSPop*.

Assign the updated archive A with all the solutions in the first front of *NSPop* and their corresponding fitness as f_A.

N_A = number of solutions in A

If $N_A > A_s$

  Determine the crowding distance of all the solutions in archive A.

  Sort the solutions in the descending order of crowding distance.

  Select the first A_s solutions from the crowding distance sorted solutions as A and the corresponding fitness as f_A.

End If

Return A, *and* f_A

Fig. 7.5. Pseudocode of archive updating procedure

7.3. Experimental settings

This study considered eight different scheduling instances with varying levels of complexity, where the instance with the lowest dimension constitutes three orders and two machines, and the instance with the highest dimension comprises twenty orders and five machines. The problem instances P1S1 and P2S1 of Chapter 6 are not included in this chapter, as these instances possess only one feasible solution.

The performance of the proposed NSpTLBO algorithm is compared with six multi-objective metaheuristic techniques, namely, Fast and elitist multi-objective Genetic algorithm (NSGA II) (Deb et al., 2002), Front based Yin Yang Pair Optimization (FYYPO) (Punnathanam &

Kotecha, 2017), Immune algorithm with Non-dominated Neighbor based selection (NNIA) (Gong et al., 2008), Multi-objective Slime Mould Algorithm (MOSMA) (Premkumar et al., 2021), Multi-objective Water Cycle Algorithm (MOWCA) (Sadollah et al., 2015), and Multi-objective Salp Swarm Optimization (MOSSA) (Mirjalili et al., 2017) by solving the eight combinatorial scheduling cases. A brief description of the multi-objective metaheuristic techniques considered in this chapter is provided in Appendix A.

The objective function and all the metaheuristic techniques are implemented in MATLAB R2020b. In view of the stochastic nature of metaheuristic algorithms, 30 independent runs were performed for each of the scheduling cases using a desktop PC with an Intel Core i7 processor and 16GB RAM. The population size for every algorithm is five times the number of decision variables in each problem. The termination criterion for solving the problems is the maximum allowed function evaluation equal to 10,000D, where D is the number of decision variables. The decision variables representing machines assigned to process jobs and their start time are integers, while metaheuristic techniques are generally designed for working with continuous decision variables. Hence, a round-off strategy is employed to handle the integer variables.

7.4. Results and discussion

In this section, the performance of each metaheuristic technique by incorporating the no wait time heuristic mechanism is compared with their respective stand-alone technique in terms of the determined corner points, the hypervolume ratio, and the coverage metric measures. The performance of NSpTLBO with and without including the heuristic mechanism is analyzed by convergence analysis of Pareto fronts obtained on solving all scheduling problems. The Gantt chart of the corner solution obtained on solving the most complex problem among the considered scheduling problems is analyzed to determine the benefits of the heuristic mechanism.

7.4.1. Analysis based on performance metrics

The hypervolume ratio and the coverage metric are utilized for the quality analysis of the non-dominated solutions identified by the techniques with and without the no-wait time heuristic mechanism. The hypervolume ratio and the coverage metric of an algorithm are determined based on the Pareto front identified from the non-dominated solutions reported in all the runs by an algorithm. The hypervolume ratio helps to identify the convergence and the diversity of the nondominated solutions identified by the algorithms to the global Pareto front. In this study, the global Pareto solutions for each case are determined by selecting the non-dominated solutions from the combined solution pool of all the runs performed by all the algorithms. The hypervolume ratio of the Pareto front determined by each technique to the global Pareto front is provided in Table 7.1.

Table 7.1. Hypervolume ratio of the Pareto front reported by metaheuristic techniques (MT) and the hybrid metaheuristic-heuristic techniques (MHT)

Problems	NSpTLBO		NSGA II		FYYPO	
	MT	MHT	MT	MHT	MT	MHT
P1S2	**1.0000**	0.9995	**1.0000**	0.9995	**1.0000**	0.9995
P2S2	**0.9853**	0.9487	**0.9747**	0.9487	0.9348	**0.9487**
P3S1	0.0960	**0.9997**	0.9507	**0.9810**	0.0896	**1.0000**
P3S2	0.8614	**0.9994**	0.8776	**0.9872**	0.8550	**0.9843**
P4S1	0.9395	**0.9997**	0.9274	**0.9995**	0.9600	**0.9864**
P4S2	0.8465	**0.9763**	0.8549	**0.9985**	0.8519	**0.9758**
P5S1	0.9428	**0.9891**	0.9077	**0.9932**	0.8939	**0.9679**
P5S2	0.7754	**1.0000**	0.7858	**0.9978**	0.7505	**0.9634**
Problems	**NNIA**		**MOSMA**		**MSSA**	
	MT	MHT	MT	MHT	MT	MHT
P1S2	**1.0000**	0.9995	0.9444	**0.9995**	**1.0000**	0.9995
P2S2	0.9271	**0.9487**	0.8643	**0.9438**	0.9458	**0.9487**
P3S1	0.0826	**0.9673**	0.0161	**0.9078**	0.9686	**0.9998**
P3S2	0.8567	**0.9878**	0.4780	**0.9223**	0.8126	**0.9965**
P4S1	0.9224	**0.9939**	0.0045	**0.9069**	0.9472	**0.9931**
P4S2	0.8198	**0.9847**	0.5064	**0.8470**	0.8011	**0.9378**
P5S1	0.0923	**0.9572**	0.0051	**0.0634**	0.0984	**0.9748**
P5S2	0.7632	**0.9830**	0.2468	**0.7375**	0.6391	**0.9177**

In P1S2, the hypervolume ratio of all the techniques with the repair mechanism is slightly lesser than the stand-alone solvers. On considering the hypervolume values obtained by NSpTLBOwith the heuristic mechanism, the Pareto solutions are observed to be well converged and diversified. The hypervolume ratio reported by all the hybrid algorithms except MSSA is significantly better than their stand-alone version, where the maximum benefit is observed in P3S1 by NSpTLBO, FYYPO, MOSMA, and NNIA and also in P5S1 by NNIA and MSSA.

A hypervolume ratio close to one would not be able to ensure that the solutions in the determined Pareto fronts are converged to the global Pareto front. A scenario might occur with solutions so close but not precisely converged to the global Pareto front, where the hypervolume ratio would be close to one. In order to quantify the number of solutions that are entirely converged to the global Pareto front, we have utilized the coverage metric (C_2R). The coverage metric value obtained by all the algorithms with and without the heuristic technique is given in Table 7.2. The coverage metric of NSpTLBO and FYYPO is more than zero in all the problems, implying these algorithms have identified multiple non-dominated solutions in the global Pareto front. The coverage metric values of all hybrid techniques are better than the metaheuristic algorithms in the majority of the scheduling problems. A higher value of coverage metric obtained for the Pareto front identified by an algorithm does not infer that the obtained Pareto front contains all the solutions in the global Pareto front. A combined analysis of the hypervolume and coverage metric helps to determine the efficiency of any technique to solve a multi-objective problem.

In the case of P1S1, the coverage metric and hypervolume ratio obtained by NSpTLBO is one, indicating the identification of all non-dominated solutions converged to the global Pareto front. Similar inferences can also be found in P3S1 by the hybrid FYYPO-heuristic technique.

In contrast, the hybrid MSSA with the no-wait time heuristic technique has provided a high value for the hypervolume ratio, while the coverage metric value is too small. Such an inference implies that the technique has identified diversified non-dominated solutions that are close to the global Pareto; however, most of those solutions are not precisely converged to the global Pareto. Similar observations are present in the results provided by MOSMA and MSSA in the majority of the scheduling cases.

Table 7.2. Coverage metric of the Pareto front reported by metaheuristic techniques (MT) and the hybrid techniques (MHT)

Problems	NSpTLBO		NSGA II		FYYPO	
	MT	MHT	MT	MHT	MT	MHT
P1S2	**1.0000**	0.6000	**1.0000**	0.6000	**1.0000**	0.6000
P2S2	**0.9789**	0.6250	**1.0000**	0.6250	0.1111	**0.6250**
P3S1	0.0000	**0.3333**	**0.0000**	**0.0000**	0.0000	**1.0000**
P3S2	0.0059	**0.9444**	0.5089	**0.9412**	0.0000	**0.8000**
P4S1	0.0000	**0.4286**	0.0000	**0.4286**	0.0000	**0.1667**
P4S2	0.0000	**0.9333**	0.0000	**0.8235**	0.0000	**0.5000**
P5S1	0.0000	**0.3333**	0.0000	**0.8000**	**0.0000**	**0.0000**
P5S2	0.0000	**1.0000**	0.0000	**0.7222**	0.0000	**0.1176**
Problems	**NNIA**		**MOSMA**		**MSSA**	
	MT	MHT	MT	MHT	MT	MH
P1S2	**1.0000**	0.6000	**1.0000**	0.5714	**1.0000**	0.6000
P2S2	0.3333	**0.6250**	0.0000	**0.9231**	0.1367	**0.6250**
P3S1	**0.0000**	**0.0000**	0.0000	**0.0714**	0.0000	**0.2500**
P3S2	0.0000	**1.0000**	0.0000	**0.9435**	0.0000	**0.7222**
P4S1	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
P4S2	0.0000	**1.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
P5S1	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
P5S2	0.0000	**0.9333**	**0.0000**	**0.0000**	**0.0000**	**0.0000**

7.4.2. Analysis of Pareto solutions

The performance evaluation of NSpTLBO and other metaheuristic techniques along with the effectiveness of the heuristic mechanism, are analyzed with the help of Pareto solutions. Initially, the corner solutions determined by each metaheuristic technique are studied to determine their efficiency in identifying converged corner solutions. The Pareto front

determined by NSpTLBO with and without using the heuristic mechanism is also studied in this section. Gantt charts of some Pareto solutions are also analyzed to visualize the corner solutions of the most converged Pareto front.

Analysis of corner solutions

The effect of the heuristic mechanism on all the metaheuristic techniques is studied by solving the eight scheduling problems and comparing the obtained results with their respective stand-alone algorithms. The corner solutions obtained by metaheuristic techniques (MT) and hybrid metaheuristic-heuristic techniques (MHT) for instances P2S2-P5S2 are provided in Table 7.3.

The two corner solutions representing (i) minimum makespan and maximum processing cost and (ii) maximum makespan and minimum processing cost, determined by each of the algorithms with and without the heuristic mechanism, are compared to analyze the convergence of the non-dominated solutions to the extreme region of the Pareto front. The better solution among the corner points obtained on solving the problem with and without the heuristic mechanism is highlighted using the bold-typeface font.

In P1S2, the fitness value corresponding to the corner solution representing the minimum makespan and maximum processing cost determined by all the algorithms with and without incorporating the no-wait time heuristic mechanism is 8 and 25, respectively. Similarly, the other corner solution representing the minimum processing cost of 18 and a maximum makespan of 18 is determined by all the hybrid and stand-alone techniques, except the stand-alone MOSMA. In the case of MOSMA without the heuristic mechanism, the corner solution has a minimum processing cost of 21 and a maximum makespan of 18, indicating it has not converged to the best corner solution. In all problem instances except P2S2, the hybrid technique corresponding to each algorithm has converged to better corner solutions. In P2S2, most stand-alone algorithms have better corner solutions corresponding to the minimum makespan and maximum processing cost.

Table 7.3. Corner points identified by metaheuristic techniques (MT) and the hybrid metaheuristic heuristic techniques (MTH)

	P2S1				**P3S1**				**P3S2**				**P4S1**			
Algorithms	MT		MTH		MT		MTH		MT		MTH		MT		MTH	
	MS	PC	MS	PC	MS	PC	MS	PC	MS	PC	MS	PC	MS	PC	MS	PC
NSpTLBO	**7**	**62**	9	58	**75**	**234**	34	108	13	106	**12**	**109**	27	131	**25**	**126**
	22	46	**26**	**44**	78	230	**36**	**101**	27	91	**38**	**83**	29	123	**36**	**116**
NSGA II	**8**	**56**	9	58	**35**	**111**	35	104	14	103	**13**	**103**	28	125	**25**	**131**
	26	**44**	**26**	**44**	36	107	**36**	**102**	33	89	**38**	**83**	32	123	**35**	**116**
FYYPO	**9**	**53**	9	58	**75**	**240**	78	232	14	107	**13**	**103**	26	133	**26**	**125**
	20	46	**26**	**44**	34	105	**36**	**101**	34	89	**38**	**83**	33	121	**36**	**116**
MOSMA	10	66	**9**	**64**	91	254	**37**	**110**	27	108	**13**	**112**	89	306	**28**	**133**
	24	48	**26**	**44**	91	254	**37**	**110**	32	96	**36**	**86**	95	304	**32**	**127**
MSSA	**8**	**56**	9	58	35	107	**34**	**105**	13	105	**12**	**111**	26	128	**25**	**135**
	21	47	**26**	**44**	37	104	**37**	**101**	31	93	**38**	**83**	33	124	**37**	**117**
NNIA	**9**	**53**	9	58	77	236	**36**	**109**	17	98	**13**	**102**	28	130	**25**	**129**
	17	47	**26**	**44**	80	233	**37**	**102**	32	87	**38**	**83**	29	124	**34**	**117**
	P4S2				**P5S1**				**P5S2**							
NSpTLBO	12	129	**12**	**120**	32	176	**31**	**168**	18	160	**12**	**167**				
	27	111	**36**	**103**	34	170	**37**	**160**	26	150	**36**	**141**				
NSGA II	13	122	**11**	**125**	33	177	**30**	**168**	18	169	**12**	**169**				
	25	110	**37**	**102**	33	177	**36**	**162**	31	148	**36**	**141**				
FYYPO	13	120	**12**	**122**	34	178	**31**	**171**	17	169	**13**	**167**				
	26	110	**37**	**102**	34	178	**35**	**166**	28	152	**36**	**141**				
MOSMA	22	127	**12**	**137**	105	428	**81**	**396**	32	175	**16**	**174**				
	26	121	**31**	**109**	110	423	**94**	**392**	32	175	**32**	**153**				
MSSA	11	134	**12**	**122**	72	391	**31**	**174**	16	169	**13**	**166**				
	23	114	**35**	**105**	74	382	**36**	**164**	30	160	**36**	**144**				
NNIA	15	121	**12**	**120**	75	385	**32**	**171**	18	164	**13**	**166**				
	26	110	**37**	**102**	81	382	**36**	**166**	30	150	**36**	**141**				

MS: Makespan; **PC:** Processing cost; **MT:** Metaheuristic technique without heuristic; **MHT:** Hybrid metaheuristic-heuristic technique

The corner solutions determined by NSpTLBO with the no-heuristic mechanism are more converged to global Pareto than the solutions identified by the algorithm alone, except in the corner solution representing minimum makespan and maximum processing cost in P1S2. The corner solutions determined by the hybrid NSpTLBO-heuristic technique in P3S2 and P5S2 are better converged than any other algorithms. The corner solutions corresponding to the maximum makespan and minimum processing cost determined by the hybrid NSpTLBO-heuristic mechanism in all problems except P4S1 and P5S1 have better or equal processing costs than other algorithms. Based on the corner solution analysis, all the hybrid techniques are able to obtain better-converged solutions than the stand-alone version. Similarly, out of the eight problem instances, the hybrid NSpTLBO and heuristic technique is able to determine either better or similar corner solutions compared to other algorithms in five problems for the solution with minimum makespan and maximum processing cost and in six problems for the other corner solution.

Analysis of Pareto front determined by NSpTLBO

The convergence analysis of Pareto fronts identified by NSpTLBO with and without the assistance of the heuristic mechanism is depicted in Fig. 7.6. The feasible, non-dominated solutions determined by the techniques are considered for the convergence analysis. The number of solutions in the global Pareto front of cases with longer processing time is lesser than those with shorter processing time. The Pareto front of hybrid NSpTLBO with heuristic technique dominates the solutions determined by NSpTLBO in all the cases except P1S2 and P2S2. In P1S2, the NSpTLBO alone is able to determine all the solutions in the global Pareto front. At the same time, the solutions identified by NSpTLBO in the region representing the minimum makespan and maximum processing cost of the Pareto front dominate the hybrid NSpTLBO technique. It is observed that NSpTLBO is unable to determine at least one feasible solution in any of its runs for P3S1. Most of the Pareto points determined by NSpTLBO with

the heuristic mechanism converged with the global Pareto solutions in all the problems with the S2 dataset. As problem complexity increases, the performance of NSpTLBO with the heuristic mechanism is better than NSpTLBO.

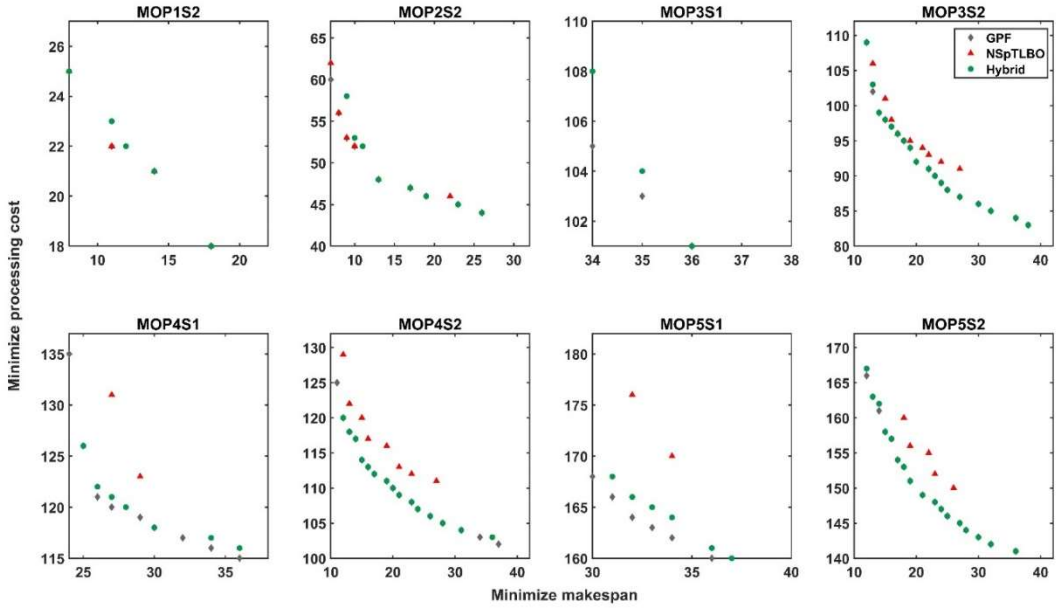


Fig. 7.6. Global Pareto front (GPF) and solutions determined by hybrid NSpTLBO heuristic technique (Hybrid) and NSpTLBO

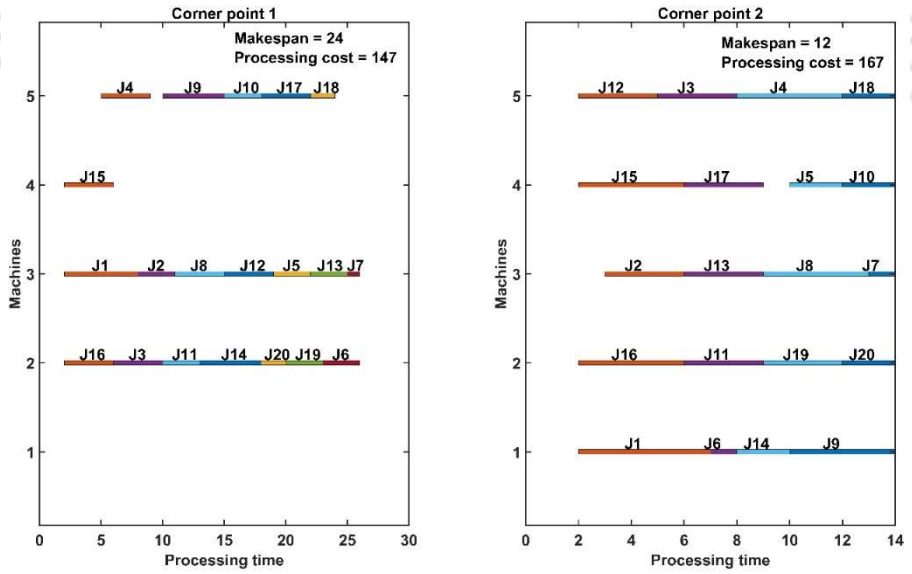


Fig. 7.7. Gantt chart representing the corner solution of MOP5S2

The Gantt chart representing the corner solutions of the global Pareto solutions for P5S2 is analyzed in Fig. 7.7 to study the benefits due to the no-wait time heuristic mechanism. Any job

is processed immediately after its preceding job in both the corner solutions because of the no wait time heuristic mechanism except J_9 in corner solution 1 and J_5 in corner solution 2. The release date of J_9 (= 10) and J_5 (= 10) is higher than the end time of their preceding order, resulting in an idle time before the processing of these jobs on the allotted machines.

7.5. Conclusion

This chapter proposed a multi-objective variant named Non-dominated Sorting phase-wise Teaching Learning Based Optimization for solving the multi-objective scheduling problem with minimization of the processing cost and the makespan as the conflicting objectives. The proposed multi-objective technique is hybridized using a no-wait heuristic mechanism to assist in determining better feasible schedules. The efficiency of the proposed NSpTLBO is analyzed by solving eight scheduling cases and compared with five other multi-objective metaheuristic techniques. Based on the corner solutions of the Pareto front determined by each algorithm and other matrices such as hypervolume and coverage metric, the proposed NSpTLBO has provided a competitive performance to techniques. This study also compared the performance of NSpTLBO and all other multi-objective metaheuristic techniques with and without incorporating the no-wait time heuristic mechanism, which revealed the inclusion of the heuristic mechanism to be beneficial in determining better converged non-dominated solutions.

This study has been extended to a practical application of scheduling the vegetable processing and packaging facility, where various vegetable consignments need to be packed using dissimilar parallel machines. The problem considers the real-life complexities, such as the arrival and discharge time of consignments and setup time for the pre-processing requirement between consecutive consignments. The objective is to identify optimal schedules with minimum total processing cost and minimum makespan by satisfying the associated constraints. More details related to this study are given in Appendix F.

Chapter 8

An efficient optimization strategy for vapour compression–absorption based cascaded refrigeration system using metaheuristic techniques

This chapter analyzes the efficacy of metaheuristic techniques in optimizing the combinatorial model of Compression–Absorption Cascade Refrigeration System (CACRS). The optimization procedure for optimizing the CACRS system using metaheuristic techniques presented in Nagraj et al. (2022) is studied, and identified its limitations. An efficient solution structure is proposed in this chapter to overcome this limitation. The efficacy of the proposed solution structure and the optimization procedure are analyzed using multiple metaheuristic techniques. The chapter is structured as follows: Section 8.1 describes the CACRS system. The detailed optimization procedure, with an efficient solution structure, employed for the thermodynamic optimization of CACRS is provided in Section 8.2. Section 8.3 specifies the experimental setting and analyzes the results using different metaheuristic techniques in Section 8.4. Finally, Section 8.5 concludes this chapter by highlighting the major findings of this study.

8.1. System description of CACRS

The schematic diagram of CACRS is shown in Fig. 8.1. It comprises a vapor compression refrigeration system (VCRS) in low temperature loop (LTL) and a vapor absorption refrigeration system (VARS) in high temperature loop (HTL) as multistage units for reducing the compression ratio and increasing the volumetric efficiency of the compressor. Refrigerants, such as R290, R123, R1234yf, R1234ze, R22, R134a, R152a, R236ea, R236fa R245ca, and R245fa can be used as working fluid in VCRS, while absorbent solution-refrigerant pairs of LiBr-H2O, LiCl-H2O and (CaCl2-LiBr-LiNO3)-H2O are used in VARS. The objective of thermoeconomic optimization is to identify optimal operating parameters with the

Kommadath, R., D. Maharana, P. Kotecha, and R. Anandalakshmi, *An efficient optimization strategy for vapor compression–absorption-based cascaded refrigeration system using various evolutionary techniques.*, International Journal of Green Energy, 2023 p. 1-16. DOI: https://doi.org/10.1080/15435075.2023.2244049

minimization of the total annual cost as the objective. The thermoeconomic model of the CACRS is detailed in Appendix G.

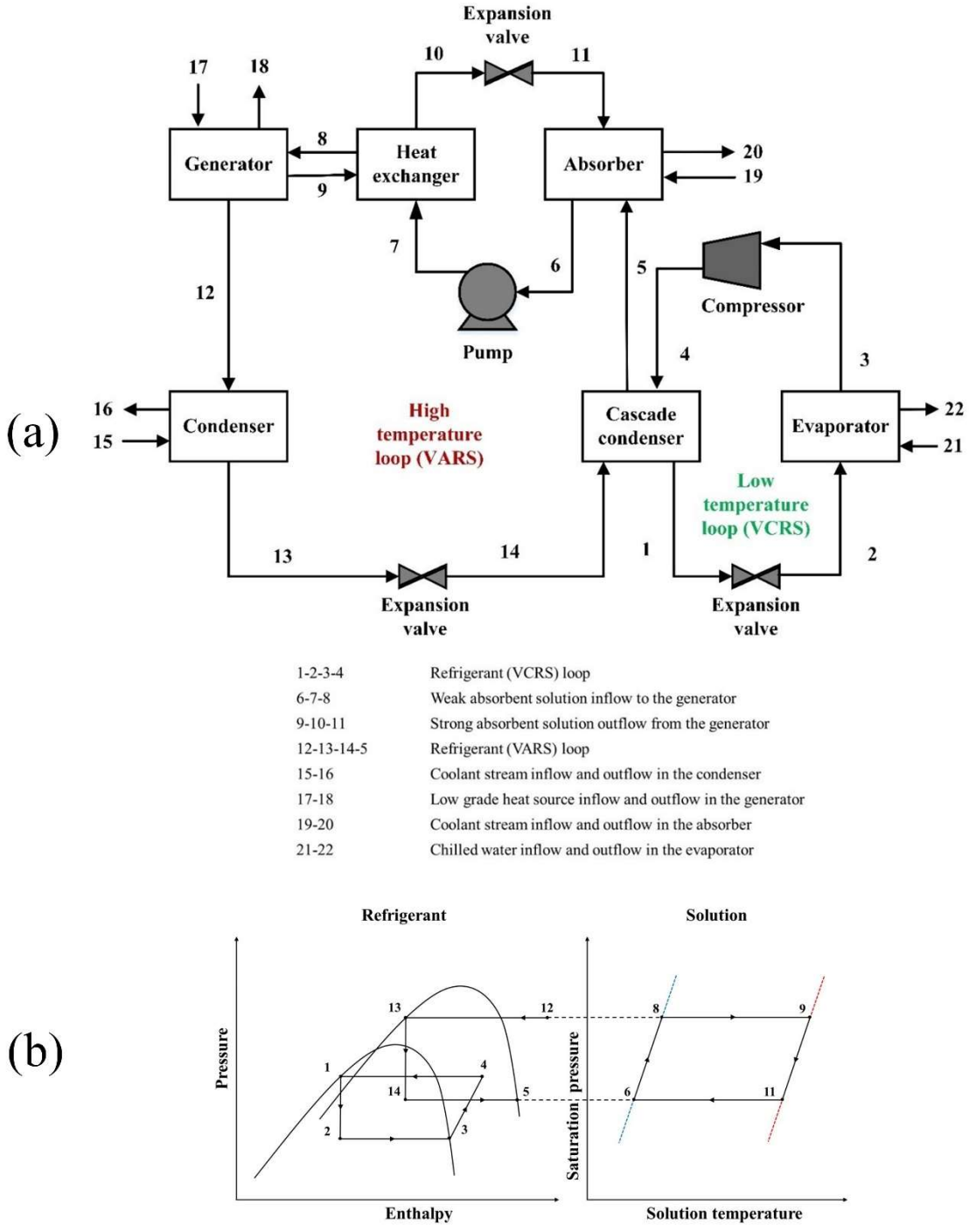


Fig. 8.1 (a) Compression-absorption cascaded refrigeration system (b) Pressure versus temperature diagram

8.2. Proposed optimization procedure

The mathematical representation of the objective function and associated constraints are provided in Eq. (8.1).

$$
\begin{aligned}
&\textit{Minimize}\ \ f\left(R,S,T_1,T_3,T_6,T_{12},T_{13},T_{overlap},\varepsilon\right)=\textit{total annual cost}\\
&s.t. \qquad X_9-X_6>0\\
&\qquad\quad T_{10}-T_C\geq 5\\
&\qquad\quad 1\leq R\leq 11,\ 1\leq S\leq 3,\ 15\leq T_1\leq 21,\ -10\leq T_3\leq 4,\ 36\leq T_6\leq 42,\\
&\qquad\quad 70\leq T_{12}\leq 87,\ 36\leq T_{13}\leq 48,\ 5\leq T_{overlap}\leq 10,\ 0.6\leq\varepsilon\leq 0.9\\
&\qquad\quad R,S\in Z^+\\
&\text{Refrigerants}\in\left\{\begin{matrix}\text{R290, R123, R1234yf, R1234ze, R22, R134a, R152a,}\\ \text{R236ea, R236fa, R245ca, R245fa}\end{matrix}\right\}\\
&\text{Salt}\in\left\{\text{LiBr , LiCl, } CaCl_2\text{-LiBr-}LiNO_3\right\}
\end{aligned}
\tag{8.1}
$$

The model consists of two integer and seven continuous decision variables that determine the optimal choice of absorbent solution-refrigerant combination and the corresponding design parameters for the model. The optimization model considers two more additional constraints apart from respective bound constraints on the decision variables as given in Eq. (8.1). The first constraint is attributed to the concentration of strong and weak absorbent solutions. It is always ensured that the concentration of the strong absorbent solution is higher than the concentration of the weak absorbent solution. The second constraint is attributed to the crystallization of salts from the absorbent solution inside the pipes. The second constraint of the optimization model ensures a minimum 5 ºC temperature difference between the crystallization temperature and solution temperature when the strong solution comes out of the heat exchanger.

The use of metaheuristic techniques requires significant care in developing an optimization model from a mathematical model, as the solutions must satisfy the physical limitations of the system. The CACRS model requires the selection of absorbent solution and refrigerant combination along with the variables such as the temperatures of cascade condenser (T_1),

evaporator (T_3), absorber (T_6), generator (T_{12}), condenser (T_{13}), the temperature difference in the cascade-condenser ($T_{overlap}$), and the effectiveness of the heat exchanger (ε) for determining the minimum total annual cost.

In order to determine the optimal solution, the optimization model needs to be solved for all possible combinations of absorbent solutions and refrigerants. The absence of the absorbent solution and refrigerant selection as decision variables requires the model to use the decision structure, as shown in Fig. 8.2, which is followed in Nagraj et al. (2022). In such a case, the optimization needs to be carried out for each absorbent solution-refrigerant combination and requires to be solved iteratively. This process significantly increases the computational load. For example, optimizing the CACRS model using a metaheuristic techniques by considering *m* refrigerents *n* absorbants, needs the model to be optimized *mnr* times before finding the best absorbent solution-refrigerant combination. Here *r* represents the number of runs considered for the metaheuristic technique.

T_1	T_3	T_6	T_{12}	T_{13}	$T_{overlap}$	ε

Fig. 8.2 Solution structure

The solution structure used in this study is designed to avoid repetitive optimization of the model and determine the best refrigerant (*R*) and absorbent solution (*S*) for the model, along with their optimal working parameters, as shown in Fig. 8.3.

R	S	T_1	T_3	T_6	T_{12}	T_{13}	$T_{overlap}$	ε

Fig. 8.3 Solution structure with absorbent solution-refrigerant pair and design parameters

Each solution from the metaheuristic algorithm is given to the optimization model that determines the required parameters to evaluate the total annual cost based on the selected absorbent solution-refrigerant pair. The optimization model uses the REFPROP to determine the appropriate state properties for the refrigerants, coolants, and hot streams. The model

checks for the constraint violation given in Eq. (8.1) and uses a penalty approach to avoid solutions with violations as shown in Eq. (8.2).

$$Objective(f)=\begin{cases}10^{7} & if\ (X_9-X_6\le 0)\vee(T_{10}-T_C\le 5)\\ TAC & otherwise\end{cases} \tag{8.2}$$

A flowchart for the optimization procedure is given in Fig. 8.4.

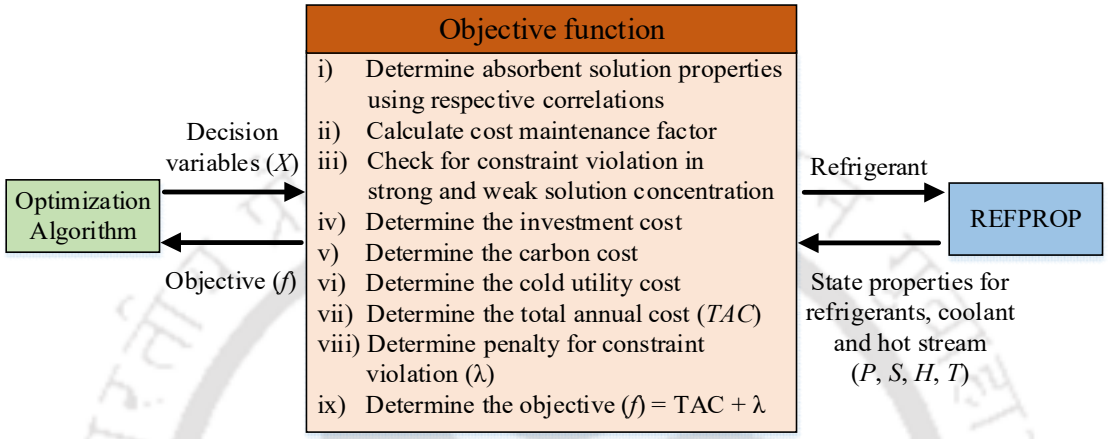


Fig. 8.4 Solution procedure

The optimization model returns the objective value corresponding to the decision variables received from the optimization algorithm. The objective returned for a feasible solution is the total annual cost as determined using the CACRS model; however, the infeasible solution returns a high penalty value to the optimization algorithm.

8.3. Experimental settings

The input data used in the modeling of CACRS is summarized in Table 8.1. This study applies ten metaheuristic techniques, namely Artificial Bee Colony (ABC) (Karaboga and Basturk 2007), Crow Search Algorithm (CSA) (Askarzadeh 2016), Dynamic Neighbourhood Learning-based Particle Swarm Optimizer (Nasir et al. 2012), Flower Pollination Algorithm (FPA) (Yang 2012), Genetic Algorithm (GA) (Goldberg 1989), Moth-flame Optimization (MFO) (Mirjalili 2015), Differential Evolution with Multi-population based Ensemble of Mutation Strategies (MPEDE) (Wu et al. 2016), Spotted Hyena Optimizer (SHO) (Dhiman and Kumar

2017), Symbiotic Organisms Search (SOS) (Cheng and Prayogo 2014) and Single-phase Multi-group Teaching-learning Algorithm (SPMGTLO) to optimize the CACRS model. The techniques are selected based on popularity, recently proposed, and variants.

Table 8.1 Input data used in the modeling of CACRS

	Data	**Value** (Jain, Sachdeva, and Kachhwaha 2015a; Jain, Sachdeva, and Kachhwaha 2015b)
Absorber	Inlet temperature of the coolant ($T_{19,}$ ˚C)	30
	Outlet temperature of the coolant ($T_{20,}$ ˚C)	35
Condenser	Inlet temperature of the coolant ($T_{15,}$ ˚C)	30
	Outlet temperature of the coolant ($T_{16,}$ ˚C)	35
Evaporator	Inlet temperature of the chilled water ($T_{21,}$ ˚C)	10
	Outlet temperature of the chilled water ($T_{22,}$ ˚C)	5
Generator	Inlet temperature of the hot stream ($T_{17,}$ ˚C)	100
	Outlet temperature of the hot stream ($T_{18,}$ ˚C)	90
Other data	Overall heat transfer coefficient (U, kW/m^2 K)	0.05 − 5
	Conversion factor for CO_2 emission (λ kg/ kWh)	0.968
	Maintenance factor (*MF*)	1.06
	CO_2 emission cost (C_{CO2}, US$/ton)	90
	Cold utility cost (CUC, US$/kWh)	10
	Input exergy cost (C_{exergy}, US$/kWh)	0.03785
	Electricity cost ($C_{electricity,}$ US$/kWh)	0.06
	Pump efficiency (η_P)	0.9
	Operating hours of the plant per year (OHP, hours)	5000
	Interest rate (IR, %)	15
	Period of repayment (PR, years)	10

On considering the stochastic nature of metaheuristic techniques, 25 independent runs are performed for each technique. The maximum allowed function evaluation of five thousand is considered as the termination criterion for all metaheuristic techniques. The population size is fixed at 30, and other parameters related to each algorithm are used as given in the open-source codes. Since most of the metaheuristic techniques are designed to handle continuous variables, the round-off method to treat the integer variable constraints is employed in the current study. A desktop computer with an Intel i7@3.4GHz processor and 16 GB RAM is used to perform the simulations using the MATLAB 2019a platform. The average computational time required by each technique to solve the CACRS model in a single run is approximately 84 seconds.

8.4. Results and discussion

The performance of the proposed solution procedure for determining the optimal absorbent solution-refrigerant-refrigerant combination is analyzed based on (i) the optimal operational parameters determined by different metaheuristic techniques, (ii) convergence and statistical analysis of the solutions and (iii) cost analysis of the optimal solutions.

8.4.1. Optimal operational parameters determined

The objective is to find optimal values of operational parameters, better absorbent solution, and refrigerant in VARS and VCRS, respectively, among all thirty-three absorbent solution-refrigerant combinations, which yield the minimum total annual cost of the CACRS. The optimum operating conditions and absorbent solution-refrigerant combinations in VARS and VCRS, respectively, are given in Table 8.2. The lowest value of total annual cost (13137.78 US$/year) is found for ($CaCl_2$-LiBr-$LiNO_3$)-$H_2O$-R152a, whereas the highest value for the total annual cost (14000.21 US$/year) is found for ($CaCl_2$-LiBr-$LiNO_3$)-$H_2O$-R123 in CACRS. It is also noted that ($CaCl_2$-LiBr-$LiNO_3$)-H_2O gives the lowest total annual cost regardless of any refrigerant used in the VCRS. The lowest and highest total annual cost (13137.78 US$/year and 14000.21 US$/year) is calculated for the optimal decision variables selected by MPEDE

and SHO, respectively. The lowest total annual cost is due to lesser condenser temperature (T_{13}) and higher temperature difference in the cascade-condenser ($T_{overlap}$) of CRS compared to the optimal operating parameters selected by other algorithms. MPEDE selects ($CaCl_2$-LiBr-$LiNO_3$)-H_2O-R152a as the best combination, which gives a lesser condenser temperature and lower minimum total annual cost compared to all other absorbent solution-refrigerant-refrigerant pairs.

Table 8.2. Optimal values of decision variables

Algorithm	R	S	T_1	T_3	T_6	T_{12}	T_{13}	$T_{overlap}$	ε	TAC (US$/year)
ABC	R152a	$CaCl_2$-LiBr-$LiNO_3$	15.00	4.00	36.00	82.72	40.51	8.09	0.60	13145.57
CSA	R152a		15.00	4.00	36.00	82.52	40.28	8.09	0.60	13139.67
DNLPSO	R152a		15.00	4.00	36.00	82.46	40.23	8.06	0.60	13138.06
FP	R290		15. 00	4.00	36.00	82.76	40.55	8.17	0.60	13191.16
GA	R152a		15.00	4.00	36.10	81.80	39.65	7.96	0.60	13165.80
MFO	R290		15.00	4.00	36.00	82.55	40.31	8.09	0.60	13164.68
MPEDE	**R152a**		**15.00**	**4.00**	**36.00**	**82.52**	**40.28**	**8.09**	**0.60**	**13137.78**
SHO	**R123**		**15.00**	**4.00**	**36.00**	**82.83**	**42.42**	**5.00**	**0.60**	**14000.21**
SOS	R152a		15.00	3.99	36.00	82.50	40.27	8.03	0.60	13139.28
SPMGTLO	R152a		15.00	4.00	36.00	82.50	40.27	8.10	0.60	13137.85

On the other hand, SHO optimizes higher condenser temperature and lower temperature difference in the cascade-condenser with ($CaCl_2$-LiBr-$LiNO_3$)-H_2O-R123 compared to all other absorbent solution-refrigerant-refrigerant combinations. This further increases the total annual cost calculated for the optimum operational parameters chosen by SHO. Among all the studied metaheuristic techniques, ABC, CSA, DNLPSO, GA, MPEDE, SOS, and SPMGTLO select ($CaCl_2$-LiBr-$LiNO_3$)-H_2O-R152a as the best absorbent solution-refrigerant-refrigerant combinations in CRS. It is noted that FP and MFO select ($CaCl_2$-LiBr-$LiNO_3$)-H_2O-R290, whereas SHO selects ($CaCl_2$-LiBr-$LiNO_3$)-H_2O-R123 as the best absorbent solution-refrigerant-refrigerant combinations in CACRS.

8.4.2. Convergence and statistical analysis

The convergence of the best and mean value of the total annual cost determined by all algorithms over function evaluation is given in Fig. 8.5. Most algorithms use at least 20% of the total function evaluations to reach their best objective function value except for SHO, which converges to its final value using 2% of maximum function evaluations. MPEDE converges to the best objective value compared to all other algorithms. Although the convergence of SHO to its final value is quicker, the final total annual cost determined by SHO is worse than all other algorithms.

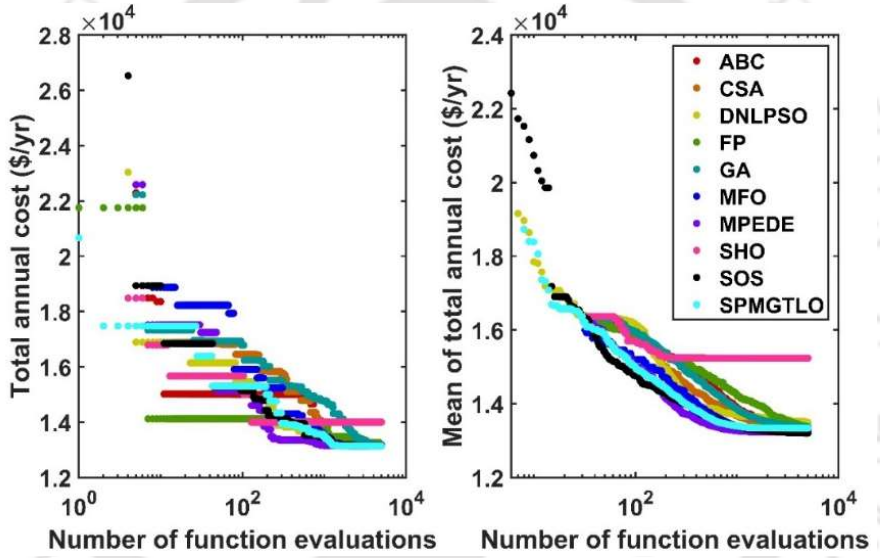


Fig. 8.5 Convergence of total annual cost against function evaluation

The mean convergence of SHO follows a similar trend of quick convergence to the final value as in its best convergence plot. The quick convergence of SHO to a worse value in its best and mean convergence analysis indicates its poor performance. The SOS algorithm slowly converges to a better mean value of total annual cost than all other algorithms, followed by MPEDE. It is observed from the best and mean convergence of GA and FP that both algorithms are not converged despite using the maximum function evaluations. The variance of objective function value in all the runs performed by each technique is shown in Fig. 8.6.

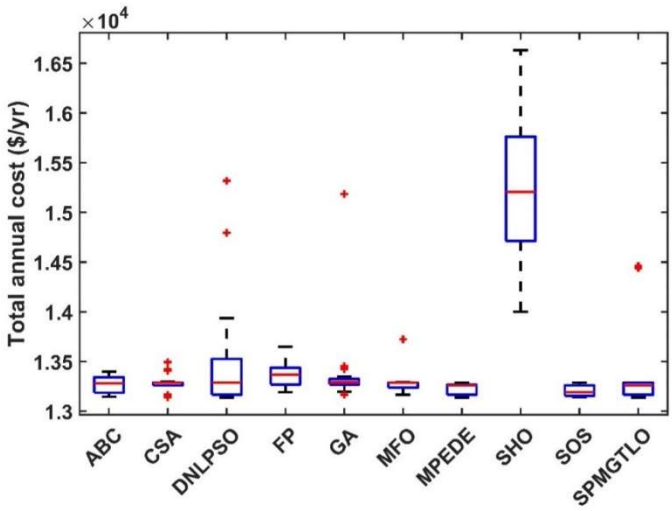


Fig. 8.6 Deviation of objective function value obtained in all runs by each algorithm

The highest deviation in the total annual cost is observed by SHO, whereas the least deviation is found in CSA. The annual cost determined in a few runs of DNLPSO, GA, and SPMGTLO reports the worst values, and it has a significant deviation from the median value of the runs, which causes these algorithms to determine a high mean value. In the case of SOS, the median value is close to the minimum annual cost, indicating its consistency in determining better annual cost in most runs. SOS determined the best median value among all the algorithms, followed by MPEDE. The median value of SPMGTLO is closer to that of MPEDE, and the deviation of annual cost determined by both algorithms is also similar. However, the box plot of MPEDE algorithms does not have any outliers, indicating it to be a more suitable algorithm than SPMGTLO to optimize the CACRS.

The bar chart representing the best, worst, mean, and median value of the total annual cost determined by all algorithms is provided in Fig.8.7. The values shown by SOS in all statistic measures except best statistics are better than MPEDE, which determined the best total annual cost value among all algorithms. All algorithms except SHO provide similar values in the best, mean, and median statistics. However, in the case of worst statistics, the algorithms such as DNLPSO, GA, and SPMGTLO provided high total annual cost value, which is also evident

from Fig. 8.6, where these algorithms show outliers far from the third quartile of their box plot. The performance of SHO in terms of all the statistics measures is not satisfactory.

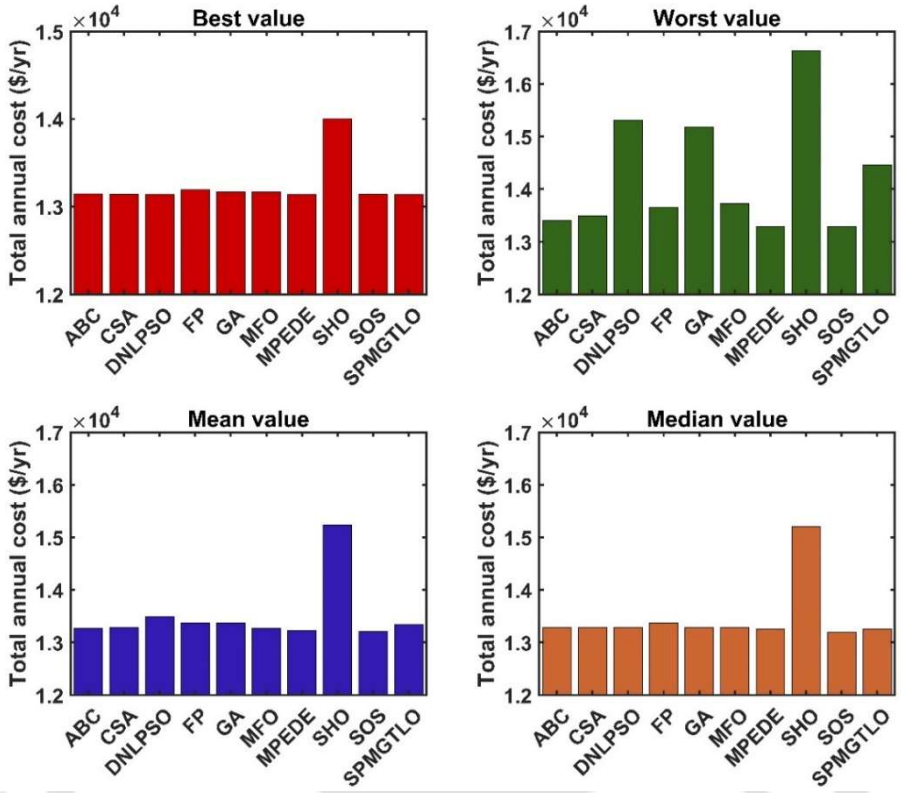


Fig.8.7 Statistical analysis of the objective value determined by algorithms

8.4.3. Cost analysis of optimal solution

The equipment cost, penalty cost, and cold utility cost are 70.9%, 15.7%, and 13.4%, for ($CaCl_2$-LiBr-$LiNO_3$)-H_2O-R152a and 72.9%, 14.7%, and 12.5%, for ($CaCl_2$-LiBr-$LiNO_3$)-H_2O-R123, respectively (Fig. 8.8). Cold utility cost is reduced due to lesser absorber heat load on the absorber and condenser for ($CaCl_2$-LiBr-$LiNO_3$)-H_2O-R152a when compared to ($CaCl_2$-LiBr-$LiNO_3$)-H_2O-R123 (Fig. 8.8). The penalty cost is lesser for ($CaCl_2$-LiBr-$LiNO_3$)-H_2O-R152a due to lesser work done by compressor and pump of CACRS, which required lesser electricity requirements. Investment cost comprises cost of exergy, electricity, and equipment. Exergy destruction is mainly due to the flow of hot and cold streams and compressor operation. The input exergy is related to exergy destruction, which is comparatively lesser for ($CaCl_2$-LiBr-$LiNO_3$)-H_2O-R152a and hence the cost of exergy is minimum for ($CaCl_2$-LiBr-$LiNO_3$)-

H_2O-R152a when compared to ($CaCl_2$-LiBr-$LiNO_3$)-H_2O-R123 (Fig. 8.8). It is noted that the cost of exergy is 16.9% of the total investment of 70.9% for ($CaCl_2$-LiBr-$LiNO_3$)-H_2O-R152a whereas it is 15.1% of the total investment of 72.9% for ($CaCl_2$-LiBr-$LiNO_3$)-H_2O-R123. Similarly, the cost of electricity is 13.8% of the total investment of 70.9% for ($CaCl_2$-LiBr-$LiNO_3$)-H_2O-R152a, whereas it is 12.6% of the total investment of 72.9% for ($CaCl_2$-LiBr-$LiNO_3$)-H_2O-R123.

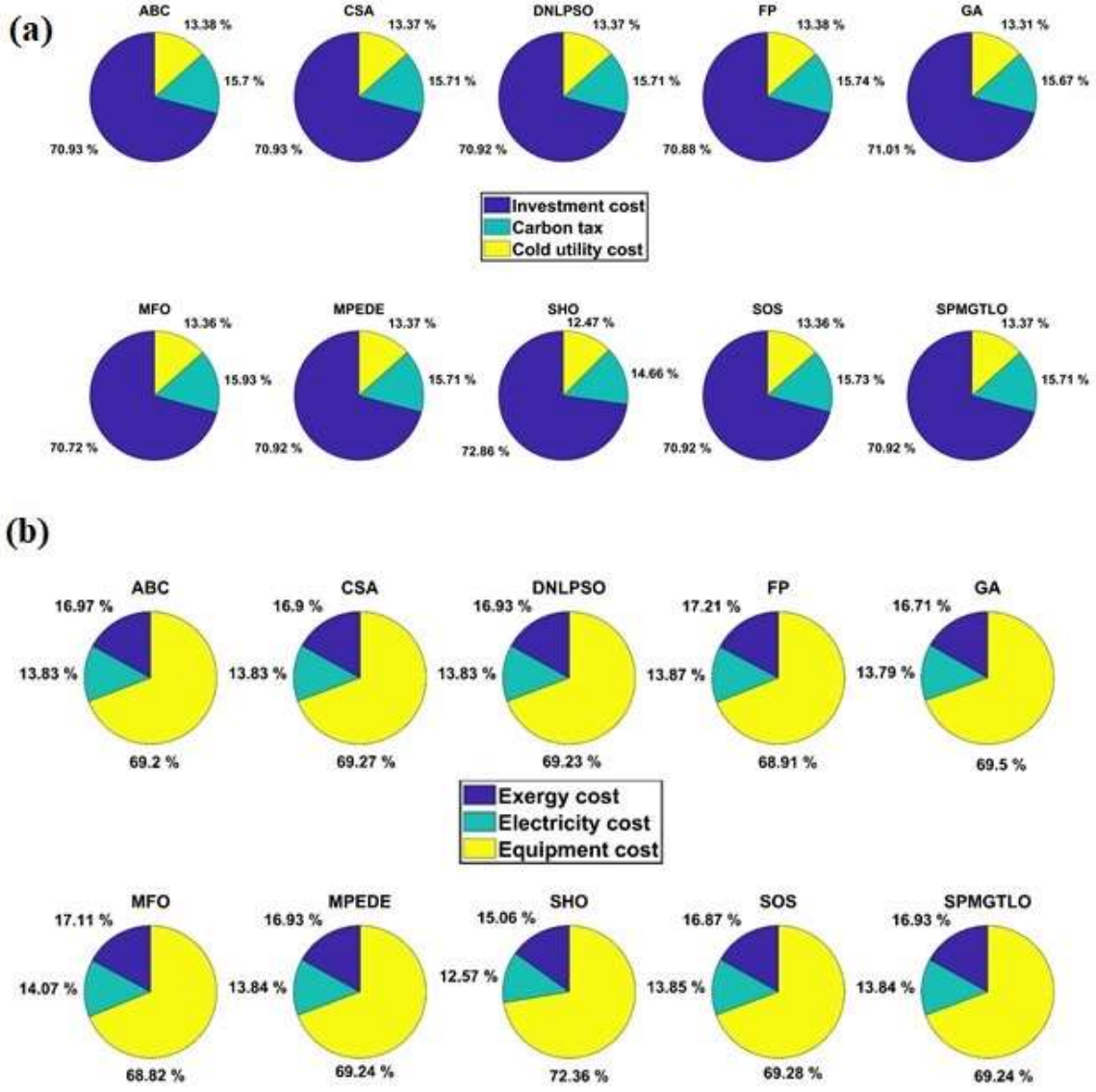


Fig. 8.8 (a) Cost break up and (b) investment cost breakup for the optimal solution

As mentioned earlier, ($CaCl_2$-LiBr-$LiNO_3$)-H_2O-R152a gives the lower minimum total annual cost among all combinations in CACRS (Fig. 8.8). It is noted that lesser condenser temperature and higher temperature difference in the cascade-condenser, which MPEDE selects, resulted in minimum investment cost for ($CaCl_2$-LiBr-$LiNO_3$)-H_2O-R152a than those obtained for ($CaCl_2$-LiBr-$LiNO_3$)-H_2O-R123, which is selected by SHO (Fig. 8.8). It is due to the lesser heat transfer area needed for ($CaCl_2$-LiBr-$LiNO_3$)-H_2O-R152a when compared to ($CaCl_2$-LiBr -$LiNO_3$)-H_2O-R123. The equipment cost is 69.2% of the total investment of 70.9% for ($CaCl_2$-LiBr-$LiNO_3$)-H_2O-R152a, whereas it is 72.4% of the total investment of 72.9% for ($CaCl_2$-LiBr-$LiNO_3$)-H_2O-R123. It may be concluded that these temperatures ($T_{13,}$ and T_{over}) affect the concentration and mass flow rate of absorbent solutions and lead to changes in the heat load, size of the components of CACRS, and total annual cost for each absorbent solution-refrigerant-refrigerant pairs (Fig. 8.8).

8.5. Conclusion

This study uses metaheuristic techniques to solve the thermoeconomic optimization of a compression-absorption cascaded refrigeration system (CACRS). Eleven refrigerants based on acceptable ODP and GWP values and three absorbent solution-refrigerant pairs are used as working pairs in the VCRS and VARS, respectively. The optimization of the combinatorial model is performed using a nonlinear objective function that minimizes the total annual cost of the system by selecting optimal operating decision variables related to the selection of absorbent solution and refrigerant combination in VARS, the refrigerant in VCRS, temperatures of cascade condenser, evaporator, absorber, generator and condenser, the temperature difference in the cascade-condenser and the effectiveness of the heat exchanger. The system optimization is carried out using ten metaheuristic techniques. MPEDE selected ($CaCl_2$- LiBr-$LiNO_3$)-H_2O-R152a as the best absorbent solution-refrigerant-refrigerant, which provided the lower minimum total annual cost for the optimal operating variables of CACRS,

whereas the use of ($CaCl_2$- LiBr-$LiNO_3$)-H_2O-R123 is observed to report the higher minimum total annual cost as selected by SHO. The use of ($CaCl_2$-LiBr-$LiNO_3$)-H_2O in VARS is chosen as the best optimal absorbent solution-refrigerant pair by all metaheuristic optimization algorithms. In contrast, seven out of the ten metaheuristic techniques selected R152a as the refrigerant in VCRS.

Chapter 9

Parallel computing strategies for sanitized teaching learning based optimization

A significant drawback of the metaheuristic techniques is that they require considerable computational effort due repeated evaluation of the objective function. Hence, to solve the computationally intensive problem, with metaheuristic techniques becomes prohibitively expensive, and the usage of parallel computing usually overcomes this. The inherent nature of Teaching Learning Based Optimization (TLBO) requires the sequential generation of new solution and its immediate evaluation to update the population, which restricts TLBO from utilizing parallel computing. This study addresses this lacuna by providing three alternate strategies that can substantially reduce computational time by harnessing the available resources. This chapter is structured as follows: Section 9.1 provides the details of Teaching Learning based optimization and discusses various issues in the implementation of Teaching learning based optimization. Section 9.2 provides the details of generic modifications employed and three modified strategies of Teaching Learning Based Optimization to harness the possibilities of parallel computing. Section 9.3 provides experimental settings employed to analyze the performance of the proposed strategies. Section 9.4 demonstrates the efficacy of these strategies on benchmark functions and their computational time benefits. Section 9.5 concludes this chapter by discussing the major findings of this study.

9.1. Teaching Learning Based Optimization

The major terminologies used in TLBO are: (i) population as the class, (ii) population size as the class size, (iii) potential solution as a learner, and (iv) value of the decision variables as marks obtained in a subject. The TLBO algorithm consists of two phases, viz., the teacher phase and the learner phase. In every iteration, each learner undergoes the teacher phase, followed by the learner phase. The solution with the best objective value in the whole class is

Kommadath, R., D. Maharana, C. Sivadurgaprasad, and P. Kotecha, *Parallel computing strategies for Sanitized Teaching Learning Based Optimization.* Journal of Computational Science, 2022. **63**: p. 101766. DOI: https://doi.org/10.1016/j.jocs.2022.101766

considered as the teacher. A new potential solution is generated in the teacher phase using Eq. (9.1), which is based on (i) variables of the class mean $\left(X_j^{avg}\right)$, (ii) the decision variables of the teacher $\left(X_j^{best}\right)$, (iii) teaching factor $\left(T_F\right)$, which is selected randomly either as one or two and (iv) the decision variables of the learner $\left(X_j^i\right)$ which is undergoing the teacher phase. If the objective value of this new potential solution is better than the objective function value of the solution that generated it, the new solution replaces the older one. The new solution is discarded if the new potential member has an objective value worse than the solution that created it. Thus, a class is updated after each learner undergoes the teacher phase.

$$T_j^{new} = X_j^i + r_j\left(X_j^{best} - T_F X_j^{avg}\right) \qquad \forall j = 1,..,N_d \tag{9.1}$$

After completing the teacher phase, the solution undergoes the learner phase. In the learner phase, the learner $\left(S^i\right)$ generates a new potential solution by learning from a random solution $\left(S^{rand}\right)$ in the class. If the objective function value of the random partner solution is superior to the solution undergoing the learner phase, then Eq. (9.2) is used to generate a new potential solution $\left(S^{new}\right)$. However, if the objective function value of the random partner is inferior to the solution performing the learner phase, then the new solution is generated using Eq. (9.3).

$$S_j^{new} = S_j^i + r_j\left(S_j^{rand} - S_j^i\right); \quad f\left(S^i\right) > f\left(S^{rand}\right) \qquad \forall j = 1,2,..N_d \tag{9.2}$$

$$S_j^{new} = S_j^i + r_j\left(S_j^i - S_j^{rand}\right); \quad f\left(S^i\right) \le f\left(S^{rand}\right) \qquad \forall j = 1,2,..N_d \tag{9.3}$$

As in the teacher phase, this new potential solution replaces the learner in the class who has performed the learner phase if it has a better objective function value; otherwise, it is rejected.

In the TLBO algorithm implemented by the inventors, a solution has to undergo the teacher phase followed by the learner phase before another member undergoes the teacher phase (Črepinšek et al., 2012; Rao & Patel, 2012). As the class can potentially be updated after a

solution completes its teacher-learner phase, the teacher and the mean of the class have to be updated. The teacher-learner phase is repeated for every solution in the class for a pre-specified number of iterations. The newly generated potential solution may exceed the domain of the decision variables, which are bounded using Eq. (9.4), before evaluating the objective function.

$$\begin{aligned} T_j^{new} &= \max\left(T_j^{new}, L_j\right) \qquad \forall j = 1,..,N_d \\ T_j^{new} &= \min\left(T_j^{new}, U_j\right) \qquad \forall j = 1,..,N_d \end{aligned} \tag{9.4}$$

Here, N_d represents the problem dimension, whereas L_j denotes the lower bound, and U_j is the upper bound of the variable j.

9.1.1 Issues with the implementation of TLBO

This section discusses three aspects of the TLBO algorithm, namely (i) identification of duplicate/ similar solutions, (ii) removal of duplicate solutions, and (iii) bounding of the out-of-bound variables. It should be noted that Rao and Patel (2012) have discussed none of these three steps in detail as part of the algorithm, probably because these aspects may be common to many of the metaheuristic techniques. However, these three steps are an integral part of the algorithm, as evident from the publicly available code given by inventors (Rao & Patel, 2012).

Identification of duplicate solutions

Duplicate solutions are those solutions that have identical values for all the decision variables. For an n-dimensional problem, solution A and solution B can be termed as duplicates only if $x_j^A = x_j^B \quad \forall j = 1,2,...,n$, where x_j^A, and x_j^B are the marks obtained by solution A and B in the j^{th} variable. It is evident that the objective function value will be equal for duplicate solutions. For instance, in a two-variable problem, learner A is characterized by (88, 95) and learner B by (95, 88). As their marks in two different subjects are different, both learners should not be treated as duplicates and must be maintained in the population without any modification. However, in the function "*remove_duplicate*" presented in the author code, the marks are sorted (using the MATLAB function "*sort*") before comparing them (using the MATLAB function

"*isequal*"). Thus the two solutions A and B would become A^{sorted} = (88 95) and B^{sorted} = (88 95) after sorting and thus making them duplicate solutions, whereas, in reality, these are not duplicate solutions. As mentioned earlier and in Črepinšek et al. (2012), this step of the algorithm has not been described in detail, and hence, the purpose of sorting the marks before identifying the duplicate solutions is not clear. To the best of our understanding, this sorting is not required and is non-intuitive while identifying duplicate solutions. It should also be noted that duplicate solutions are not similar to realizations or multiple optimal, which have a different set of decision variables but equal objective function value as described in (Deb & Saha, 2012) and (R. Kotecha et al., 2009).

Removal of duplicate solutions

As evident from the function "*remove_duplicate*" presented in the code provided by authors, the strategy employed to remove duplicate solutions is to start with the first solution and compare it with the rest of the solutions below it. On realizing a duplicate solution, the mark in one of the randomly selected subjects of the duplicate solution is modified (or mutated) using any random value within the bounds of the variable. This procedure is performed with all solutions by comparing each solution with its succeeding members only. The comparison between the current member with its preceding solutions is completely neglected. Such a strategy may not necessarily lead to the removal of all duplicate solutions in the class because the mutation itself may lead to duplication with an earlier member. Since the strategy is to compare with only successive members (i.e., members below), the duplicate solution would still exist at the end of the removal procedure. One could argue that the possibility of such an occurrence could be rare, but as with any algorithm, all possibilities need to be considered. This issue can be avoided by employing the following algorithm, which ensures that the modified solution is compared with all the unique solutions, thereby ensuring that it is unique.

It should be noted that the algorithm presented below is only suggestive, and there could be various efficient algorithms to replace duplicate solutions.

Step 1: From the given set of learners and their marks, determine the class size (*n*).

Step 2: Determine the set of unique learners (*unique_students*) and the number of unique learners (*m*). Also, separately store the set of learners that have been repeated (*non_uni_set*).

Step 3: Implement the following steps (*n-m*) times

Step 4: Select a member (*select_mem*) from the set (*non_uni_set*).

Step 5: For a randomly selected subject, modify any one of the marks of the selected solution to generate *mutated_student*. Ensure that the modified mark is within the range.

Step 6: Compare *mutated_student* with all the members in *unique_students*. If *mutated_student* is not identical to any of the learners in *unique_students*, add it to *unique_students* and remove it from *non_uni_set*, else repeat Step 5.

Step 7: Update values of *n* and *m* from size of *unique_students* and *non_uni_set*, respectively

Step 8: Go to Step 3, if (*n-m*) is not equal to zero

The detailed analysis of the TLBO code provided by the authors has revealed that the function "*remove_duplicate*" performs only partial removal of duplicate solutions. Moreover, it employs extra function evaluations to evaluate the solutions generated by removing the duplicate, thereby increasing computational load.

Bounding of the out-of-bound solutions

Marks of the learners that exceed the bound are modified using the following equations (in MATLAB functions in the author codes, *implementopti*, and *implementopti_new*).

$$\left.\begin{aligned} Mark(j) &= \max\left(Mark(j), lb(j)\right) \\ Mark(j) &= \min\left(Mark(j), ub(j)\right) \end{aligned}\right\} \quad \forall j \in 1,2,...,N_d \tag{9.5}$$

As per the above equations, if the mark in the j^{th} subject is less than the lower bound ($lb(j)$) of the j^{th} subject, then the mark is set to the lower bound. Similarly, if the mark is higher than the

upper bound, then the mark is set to the upper bound of the j^{th} subject ($ub(j)$). Such a strategy will always move the decision variable to its boundary and potentially risks the generation of duplicate solutions. For example, consider a three-variable problem with all the bounds lying between 0 and 5. Let three of the learners in the class be as shown below

$$\begin{bmatrix} S1 \\ S2 \\ S3 \end{bmatrix} = \begin{bmatrix} -0.8 & 5 & 4 \\ 0.9 & 3 & -7 \\ 0 & 5 & 4 \end{bmatrix}$$

It can be found that S1 and S2 have exceeded the bounds for the first and third subjects, respectively. Applying the strategy shown in Eq. (9.5) would yield the following class wherein S1 and S2 have been moved to the boundary and make S1 and S3 duplicate solutions.

$$\begin{bmatrix} S1 \\ S2 \\ S3 \end{bmatrix} = \begin{bmatrix} 0 & 5 & 4 \\ 0.9 & 3 & 0 \\ 0 & 5 & 4 \end{bmatrix}$$

Rather than these two equations, the following implementation can be used as they will reduce the possibility of a duplicate being generated and the solution being shifted to the boundary.

$$\begin{aligned} &if\ Mark(j) < lb(j)\ \vee\ Mark(j) > ub(j) \\ &\quad Mark(j) = \left(lb(j) + \left(ub(j) - lb(j)\right) * rand\right) \\ &end \end{aligned}$$

In the implementation shown above, if the mark of the j^{th} subject is less than the lower bound or greater than the upper bound of the j^{th} subject, then the mark in the j^{th} subject can be generated randomly within the bounds. In the example shown above, the learners S1 and S3 would not necessarily be equal if someone followed this bounding procedure.

It can also be noted that duplicate removal steps increase the computational burden while evaluating a computationally expensive objective function. Hence in this study, we have not considered the duplicate removal procedure in the TLBO. The TLBO algorithm that is not incorporated the identification and removal of duplicate solutions is termed as Sanitized

Teaching-Learning-based optimization (sTLBO). A detailed flowchart of sTLBO is shown in Fig. 9. 1 and detailed pseudocode is provided in Appendix H. In the rest of the chapter, to avoid any confusion, we will refer to the TLBO scheme in Fig. 9. 1 as sTLBO.

9.1.2 Limitations of Teaching Learning Based Optimization for parallelization

Let N_P denotes the class size, and N_G represents the number of generations. The number of times the objective function evaluation occurred in the entire algorithm is given by $N_P + 2N_PN_G$ in which N_P evaluations are required for the initial evaluations, whereas $2N_PN_G$ objective function evaluations are required in the actual algorithm. A careful analysis of the algorithm shows that the objective function evaluation can be parallelized only for the initial population. Subsequent to this, there is no scope for parallelizing the evaluation of multiple solutions because, as per the requirement of the algorithm, the class has to be updated immediately after every objective function evaluation.

For example, if the class size is 50, the number of generations is 100, and it requires one second to evaluate the objective function for a solution, then the total amount of computational time required by TLBO to evaluate the objective function would be 10050 seconds. The evaluation of the initial population requires 50 seconds, while the remaining 10000 seconds would be required to evaluate the objective function in iterative loop. Suppose there are 50 computational resources available that can simultaneously evaluate the objective functions. In that case, these can be used only to reduce the time required to evaluate the objective function of the initial population from 50 seconds to one second. The available parallel resources would not help to reduce the 10000 seconds required to evaluate the objective function. This limitation can potentially restrict the use of TLBO for computationally intensive problems despite its competitive performance compared to other techniques. The proposed strategies will overcome this drawback, making TLBO appealing to solve many complex real-life optimization problems.

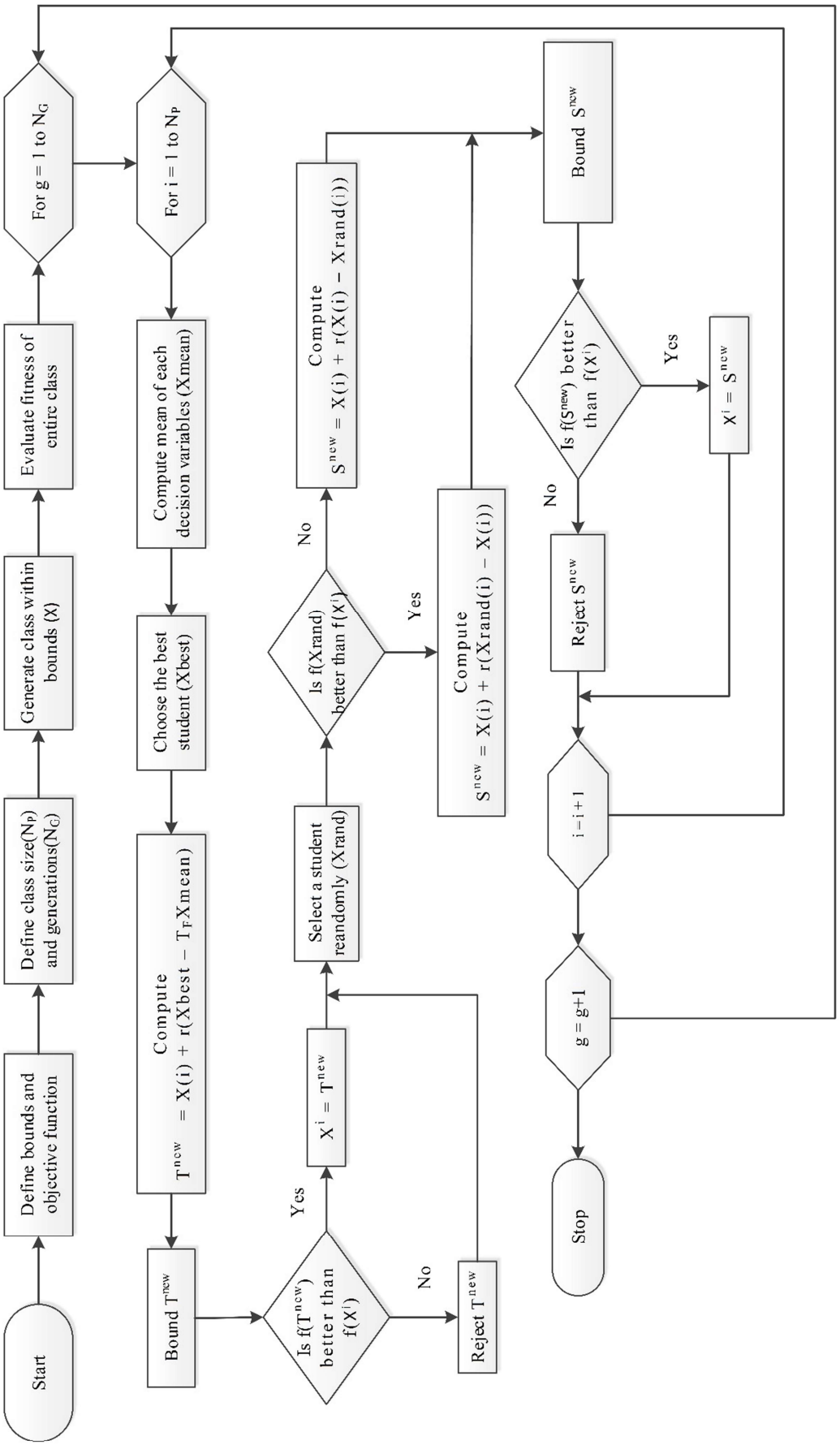


Fig. 9. 1 Schematic representation of the sTLBO

9.2. Modified Strategies of Teaching Learning Based Optimization

This section discusses some of the modifications performed to the sTLBO to acquire the benefits of parallel computing. Subsequently, three parallelized strategies capable of harnessing the benefits of parallel computing to reduce the computational efforts required by sTLBO are presented. It should be noted that all three strategies employ the generic modifications specified below to gain the promising features of parallel computing.

Updating the class with the solution generated in the teacher phase

In sTLBO, a new potential learner $\left(T^{new}\right)$ is generated every time a solution $\left(X^{i}\right)$ of the class performs the teacher phase using Eq. (9.1). In sTLBO, if the objective function value of T^{new} is better than the objective function value of X^{i}, then T^{new} replaces X^{i} else T^{new} is discarded. In the event that the solution T^{new} is inferior to X^{i}, the computational effort and the function evaluation utilized for T^{new} remain unused, even though there exist other solutions in the class that are inferior to T^{new}. Thereby, in all the proposed parallelized strategies discussed in this chapter, it is ensured that a newly generated potential solution is discarded if it is worse than the entire class, thus enabling efficient utilization of the computational effort.

Generation of potential solutions in the learner phase

A new solution $\left(S^{new}\right)$ is generated in the learner phase, using the current learner $\left(S^{i}\right)$ undergoing the learner phase and another random learner from the class $\left(S^{rand}\right)$. In sTLBO, for a minimization problem, based on the objective function of learner i $f\left(S^{i}\right)$ and the random solution $f\left(S^{rand}\right)$, the potential new solution is generated by either of the two equations, Eq. (9.2) and Eq. (9.3). However, to the best of our knowledge, there is no specific reason why the Eq. (9.2) is to be used if $f\left(S^{i}\right) > f\left(S^{rand}\right)$ and as to why the Eq. (9.3) is to be used if

$f\left(S^i\right) \leq f\left(S^{rand}\right)$. One can devise examples in which if $f\left(S^i\right) > f\left(S^{rand}\right)$, the use of Eq. (9.3) would provide a better solution and vice versa. Hence, in all the proposed strategies, we eliminate comparing the objective function value of learners S^i and S^{rand} and generate two potential solutions for a solution using the following two equations.

$$S1_j^{new} = S_j^i + r_j\left(S_j^{rand} - S_j^i\right); \qquad \forall j = 1, 2, ..N_d \tag{9.6}$$

$$S2_j^{new} = S_j^i + r_j\left(S_j^i - S_j^{rand}\right); \qquad \forall j = 1, 2, ..N_d \tag{9.7}$$

Updating the class with solutions generated in the learner phase

The updating procedure followed in the learner phase is the same as in the teacher phase. In the learner phase, the potential solutions ($S1^{new}$ and $S2^{new}$) generated as per Eq. (9.6) and Eq. (9.7) will replace the worst solutions in the class. This updating procedure contrasts with the TLBO and sTLBO, wherein S^{new} generated using either Eq. Eq. (9.2) and Eq. (9.3) is added to the class if and only if it is better than the current solution performing the learner phase.

Teaching Factor

The sTLBO uses a single teaching factor $\left(T_f\right)$ for all the decision variables. However, the teaching factor changes for each learner in the class in every iteration. Thus for the entire sTLBO algorithm, the teaching factor is to be generated for $N_P N_G$ times. In the proposed parallelized strategies, the teaching factor is selected randomly for all the decision variables and is not constant, which in some sense reflects the ability of learners to learn from the teacher at different levels for every subject. This can potentially lead to better solutions and hence require the generation of $N_d N_P N_G$ teaching factors.

Learner Partner

In sTLBO, a partner is selected randomly every time a learner in the class performs the learner phase. It may happen that the same learner can act as a partner for more than one learner. In

the proposed parallelized strategies, it is ensured that every member of the class becomes a learning partner of some other learner in the class exactly once.

9.2.1. Member Parallelized TLBO (MTLBO)

In every generation of the MTLBO algorithm, each member generates three potential solutions (one in the teacher phase using Eq. (9.1) and two in the learner phase using Eq. (9.6) and Eq. (9.7). These three newly generated members are combined with the current population, and the best N_P solutions are selected as the updated class. Subsequent to this class update, the next solution of the updated class carries out the teacher and learner phases. The pseudocode of MTLBO is given in Appendix H, and its schematic representation is shown in Fig. 9.2. It can be observed that in every iteration, three members can be generated in parallel, and the evaluation of their objective function can also be parallelized. This is not possible in sTLBO as it updates the class after a learner has completed the teacher phase, and subsequently, the same learner performs the learner phase. Unlike in sTLBO, which had two potential class updates for every member, there is a single update in this scheme after a member completes both phases. Thus, there are a total of $N_P N_G$ updates in this parallelized scheme compared to the $2N_P N_G$ updates in the sTLBO scheme.

Moreover, it should be noted that sTLBO does not require sorting the objective function value at any stage because the newly generated solution replaces that solution undergoing the teacher or learner phases if found beneficial. However, in MTLBO, the newly generated population members can replace any learner with a worse objective function value than them. Hence, the combined population of NP+3 has to be sorted to select the best NP members corresponding to each member, thereby incurring an additional computational load.

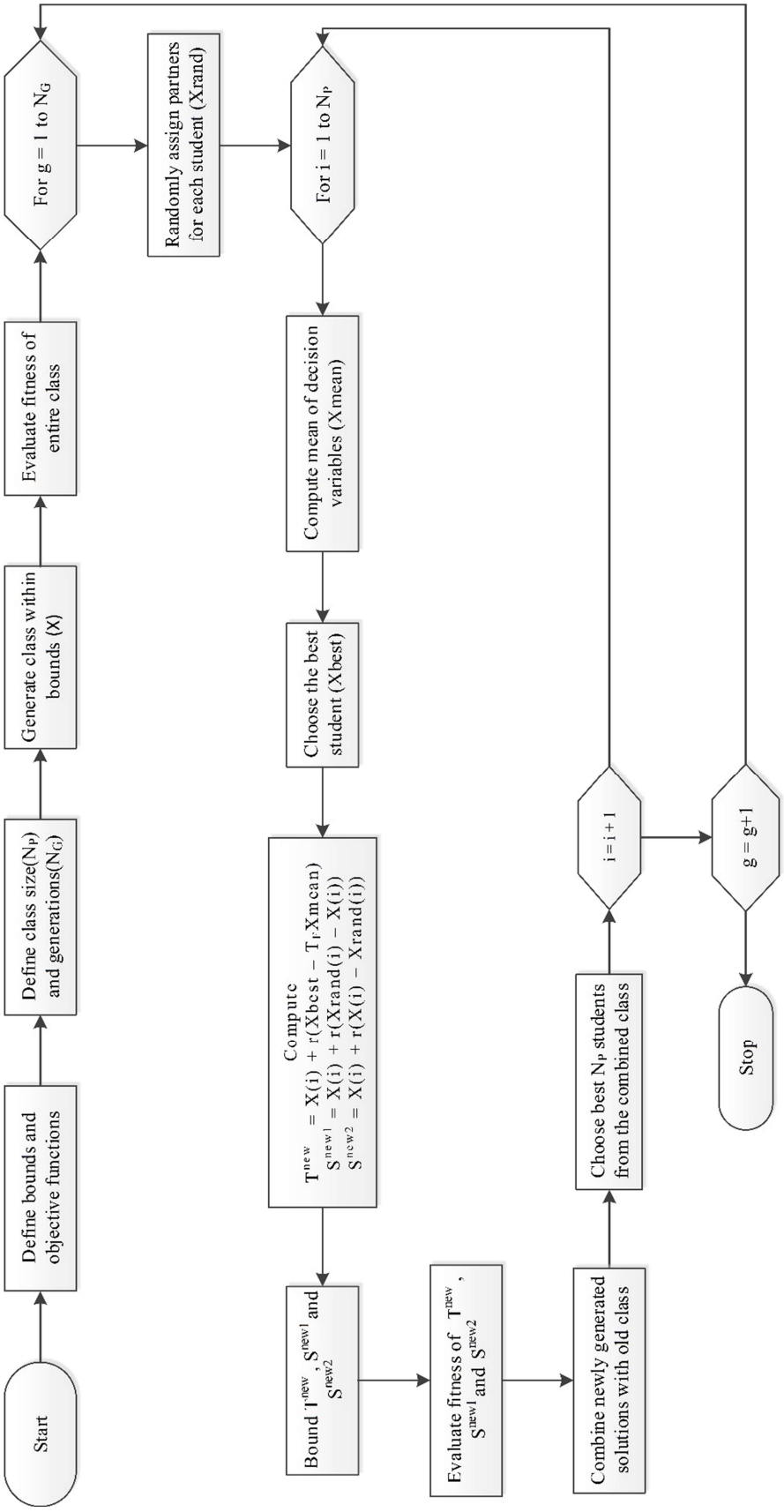


Fig. 9.2 Schematic representation of the proposed member parallelized algorithm

Without considering the duplicate removal, the number of objective evaluations required for this strategy is ($N_P + 3N_PN_G$), where N_P objective evaluations for the initial population and $3N_P$ evaluations (N_P in the teacher phase and $2N_P$ in the learner phase) in every generation. Unlike sTLBO, in which only the objective function evaluation of the initial members can be parallelized, in this proposed scheme, the objective function evaluation of initial members and the newly generated solutions from each member in every generation can be parallelized. If every function evaluation is assumed to incur a unit computational time and assumes N_R resources available for evaluating the objective function simultaneously, then the total time required to evaluate the objective function in the entire algorithm is determined as Eq. (9.8).

$$C_t^M = \left(\left\lceil \frac{N_P}{N_R} \right\rceil\right) + \left(\left\lceil \frac{3}{N_R} \right\rceil N_P N_G\right) \tag{9.8}$$

From the above equation, it can be observed that the member TLBO can benefit if there are more than one resource to evaluate the objective function simultaneously. Thus, if there are four resources, the class size is 50, the number of generations is 100, and the time required for objective function evaluation of a single member is 1 unit of time, then the total computational time taken for evaluating the objective function is 5013 units of time. The initial population could be evaluated simultaneously in 13 units of time, and the objective evaluation of the three potential solutions generated by each member can be parallelized in every iteration. Hence all the iterations of MTLBO can be completed in 5000 units of time. It is observed that a maximum of three solutions (except the initial population) can be evaluated simultaneously in the algorithm, and after the initialization phase, it cannot harness any benefits if the number of resources is more than three.

9.2.2. Class Parallelized TLBO (CTLBO)

In this scheme, the entire class performs the teacher and the learner phase simultaneously to generate $3N_P$ potential solutions using Eq. (9.1), Eq. (9.6) , and Eq. (9.7) in every iteration.

These $3N_P$ solutions are combined with the current N_P population members, and the best N_P solutions survive for the next iteration. A schematic representation of the entire algorithm is given in Fig. 9.3, and the detailed algorithm is provided in Appendix H. It can be noted that the generation and the evaluation of these $3N_P$ potential members can be parallelized. Unlike in the sTLBO, which has $2N_P$ updates per iteration, or the MTLBO, with N_P updates per iteration, this parallelized strategy has only a single update per iteration. However, CTLBO requires the sorting of the objective function value of $3N_P$ members, which is not necessary either in the sTLBO or in MTLBO. As this scheme simultaneously implements the teacher phase and the learner phase of the entire class, the mean of the class is computed only once in a particular iteration, which, in contrast to sTLBO, requires the determination of mean for N_P times in each iteration. The total number of function evaluations needed for CTLBO is also $(N_P + 3N_PN_G)$, where N_P evaluations are for the initial population, and $3N_P$ evaluations (N_P in the teacher phase and $2N_P$ in the learner phase) are required in every iteration. In this scheme, the evaluation of the initial population and the new potential solutions generated in every iteration can be parallelized. Assuming that every function evaluation incurs a unit computational time and N_R computational resources available to evaluate the objective function simultaneously, then the amount of time required to evaluate the objective function is given in Eq. (9.9).

$$C_t^C = \left(\left\lceil \frac{N_P}{N_R} \right\rceil\right) + \left(\left\lceil \frac{3N_P}{N_R} \right\rceil N_G\right) \tag{9.9}$$

For the previous example of $N_R = 4$; $N_P = 50$; $N_G = 100$, with the amount of time required for evaluating the objective function for a single solution being 1 unit of time, the amount of computational time required for evaluating the objective function is 3813 units of time. The initial population could be evaluated in 13-time units, and in every generation, the evaluation of the objective function of the 150 potential solutions can be parallelized, and those solutions are evaluated in 3800 units of time. This is 2.63 times less than what would be required in

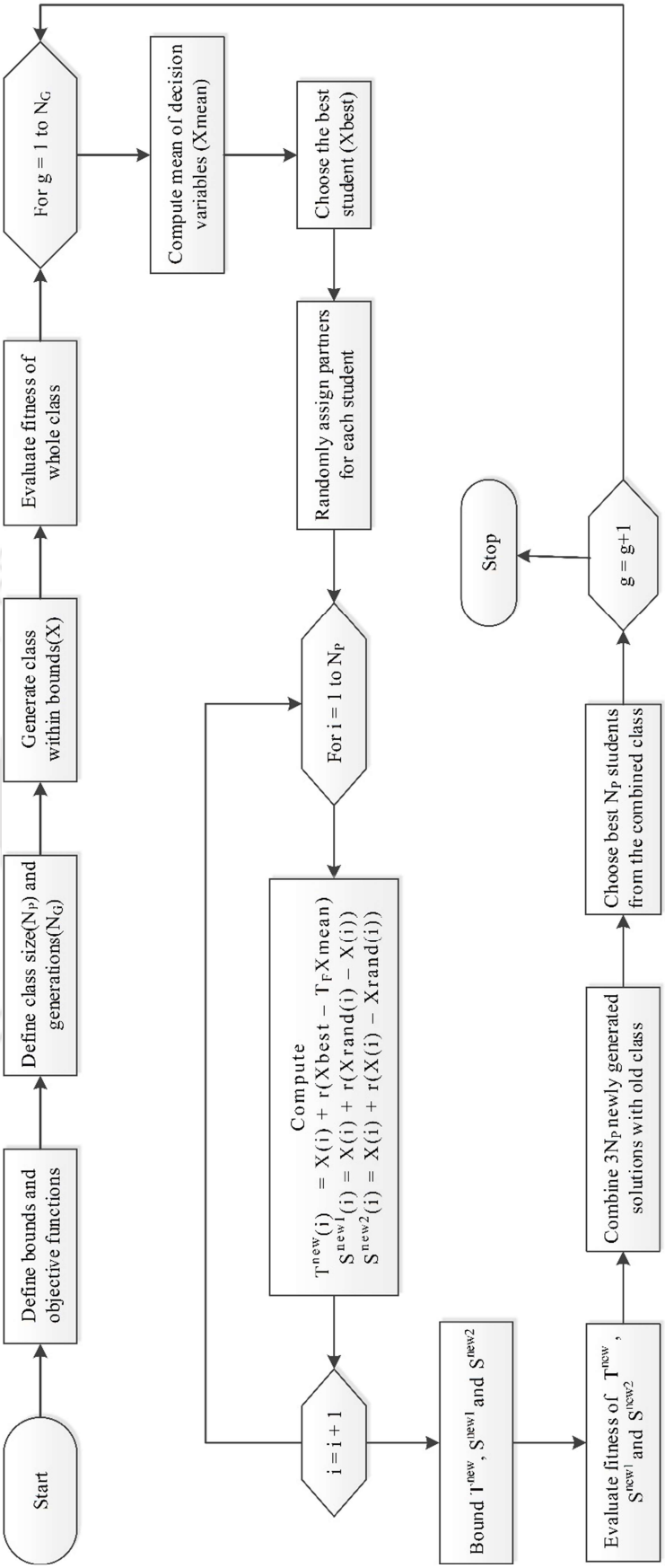


Fig. 9.3 Schematic representation of the proposed class parallelized algorithm

sTLBO despite exploring $N_P N_G$ additional solutions. It should be noted that as the number of computational resources increases, the advantage of this scheme would be even higher. Suppose the number of resources available to evaluate the objective function simultaneously is three, then CTLBO and MTLBO would require an identical amount of computational time for evaluating the objective function. If the number of available resources is more than 3, then CTLBO always requires lower computational time than MTLBO. However, it can be observed that for the instance in which the number of computational resources is greater than $3N_P$, then only $3N_P$ of them would be used for the evaluation of the objective function, and the remaining would not help in speeding up the evaluation of the objective functions in the algorithm.

9.2.3. Phase Parallelized Algorithm (PTLBO)

In every generation of this scheme, each learner does not undergo the teacher phase one after the other. In contrast, the entire class undergoes the teacher phase simultaneously to generate N_P potential solutions using the Eq. (9.1). Subsequently, these N_P potential solutions are sorted along with the current class, those who have performed the teacher phase, and the best N_P members (from the $2N_P$) are selected for the learner phase. In the learner phase, each of the N_P members can simultaneously generate two potential solutions using Eq. (9.6) and Eq. (9.7) thereby generating $2N_P$ potential solutions. These $2N_P$ solutions are combined with the N_P members that underwent the learner phase, and the best N_P solutions (from $3N_P$) are selected as the population for the subsequent iteration. In PTLBO, the teacher and the learner phase themselves are not simultaneous since there is an update of the class between the teacher and learner phases. The detailed stepwise algorithm is given in Appendix H, and the schematic representation is shown in Fig. 9.4. Unlike the other strategies, PTLBO updates the class twice in every generation (once after the teacher phase and the other after the learner phase). In the first update, the sorting involves $2N_P$ solutions, whereas, in the learner phase, the sorting of $3N_P$ solutions is performed. The sTLBO would update the mean of the class N_P times in every

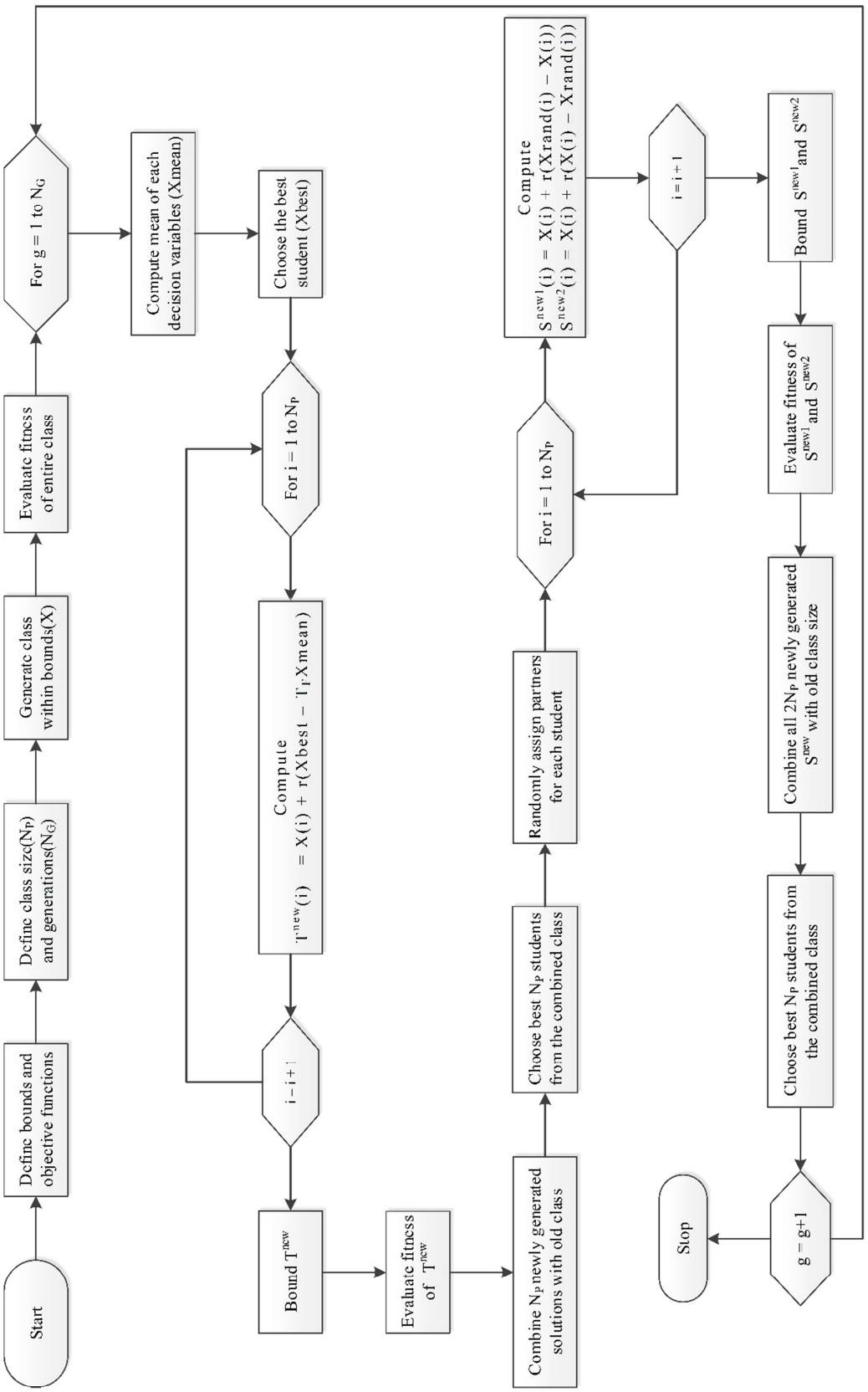


Fig. 9.4 Schematic representation of the proposed phase parallelized algorithm

iteration, whereas, in PTLBO, the mean and the best solution are to be determined only once per generation. A simultaneous evaluation of N_P solutions in the teacher phase and $2N_P$ solutions in the learner phase can be performed. It is also possible to harness the benefits of parallelization in other operations of this scheme, such as the generation of the N_P and $2N_P$ potential solutions in the teacher and learner phases, respectively. Like the other two parallelized schemes, the number of objective function evaluations required in the entire algorithm is ($N_P + 3N_PN_G$). The following equation can give the amount of time required for the entire algorithm to evaluate the objective function.

$$C_t^P = \left\lceil \frac{N_P}{N_R} \right\rceil + \left(\left\lceil \frac{N_P}{N_R} \right\rceil + \left\lceil \frac{2N_P}{N_R} \right\rceil \right) N_G \tag{9.10}$$

Consider $N_R = 4$; $N_P = 50$; $N_G = 100$, and the amount of time required for evaluating the objective function of a solution is 1 unit of time. Hence, the total computational time necessary for the objective function evaluation in the entire strategy is 3813 time units. The initial population could be evaluated in 13 seconds. In every iteration, the objective function evaluation of the 50 (= N_P) solutions in the teacher phase and the evaluation of 100 (= $2N_P$) solutions in the learner phase can be parallelized.

It is found that the computation time required to evaluate the objective function of potential solutions in CTLBO is 61.92% lesser than that of sTLBO. The benefits of CTLBO become even more pronounced with the increase in the evaluation time of a single solution. It can be observed that for this example, both the phase parallelized and class parallelized schemes require equal time. However, if $N_R = 6$; $N_P = 50$; $N_G = 100$, and the time required for evaluating the objective function is one second, PTLBO would need 2609 seconds, whereas CTLBO would require 2509 seconds for evaluating the objective function in the entire algorithm. A comparison of various aspects of sTLBO and its parallel variants is given in Appendix H.

Fig. 9.5 shows the amount of computational time required to evaluate the objective function in the sTLBO and its three parallelized strategies with respect to the availability of resources. The class size and the number of iterations are maintained constant at 100, and the computational time required for evaluating the objective function of a single member is 1 unit of time. It is observed that with the increase in the number of computational resources, the computational time required reduces for all four schemes. However, for sTLBO, the amount of computational time required is the highest among the four schemes followed by MTLBO.

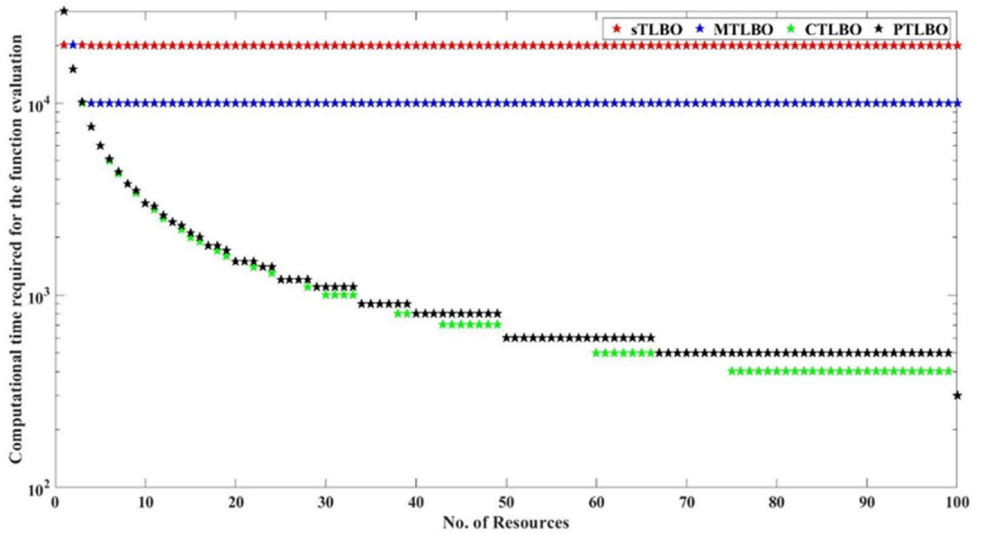


Fig. 9.5 Computational time required to evaluate the objective function with respect to N_R

It can also be observed that the computational time required by all three proposed strategies is greater than the sTLBO if $N_R = 1$ as the proposed strategies evaluate $3N_PN_G$ solutions against the $2N_PN_G$ evaluated by the sTLBO. It can also be observed that if $N_R > 3$, the amount of time required by the member parallelized scheme is lower than sTLBO and is higher than the other two schemes. In the case of CTLBO and the PTLBO, it can be seen that the computational time required for the former algorithm will never be greater than that required by the latter. On specific values of N_R, both these schemes (class and phase) require an identical amount of time for the evaluation of objective functions in the entire algorithm.

Fig. 9.6 shows the amount of computational time required to evaluate the objective function in the sTLBO and all three parallelized strategies with respect to the number of iterations. The class size and the number of resources are maintained constant at 100 and 6, respectively, with the computational time required for evaluating the objective function of a single member as 1 unit of time. It is trivial to note that the amount of computational time required increases with the increase in the number of generations. However, it can also be noticed that the amount of computational time required for sTLBO is very large compared to the other three parallelized schemes, although the other three techniques evaluate the objective function for $3N_PN_G$ whereas sTLBO evaluates it only for $2N_PN_G$. A similar relation is observed on analyzing the computational time required to evaluate the objective function with varying population size and the time required for a one-time evaluation of the objective function and the plots depicting the relationship is provided in Appendix H.

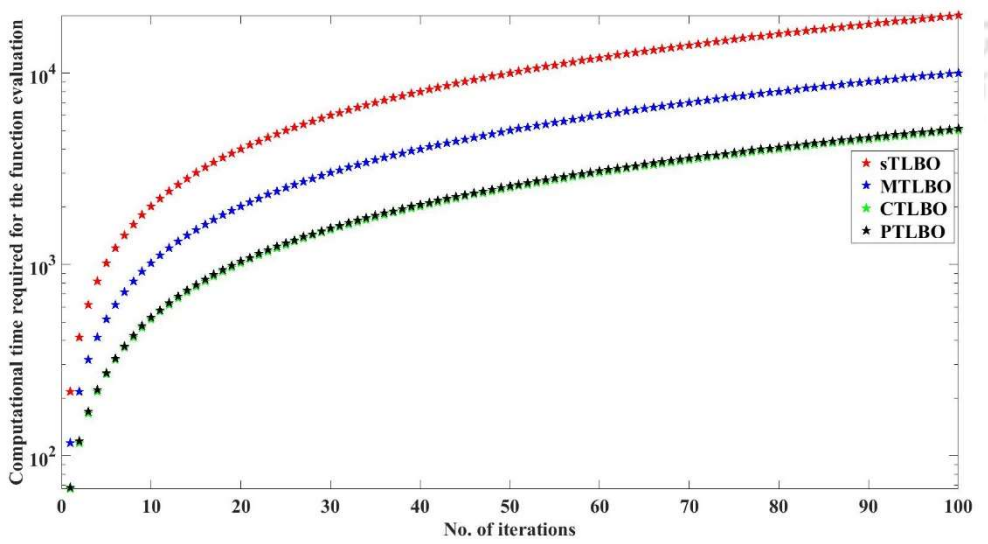


Fig. 9.6 Computational time required to evaluate the objective function with respect to N_G

It can be observed that the three parallelized strategies can lead to significant savings in computational time. The major observations from the results are summarized as

- The number of functional evaluations in the three parallelized strategies is greater than that required in sTLBO because two potential solutions are generated from a single solution in

the learner phase of parallelized strategies. It should be noted that if required, this can be modified as in the sTLBO, thereby not incurring any additional functional evaluations.

- If the number of resources is one, i.e., $N_R = 1$, the time required by the entire algorithm for evaluating the objective function at various stages will be identical for all the three parallelized strategies. However, this will be greater than sTLBO as it explores only $2N_PN_G$, whereas all the parallelized strategies explore $3N_PN_G$ learners.
- For all three parallelized strategies and sTLBO, the objective function evaluation of the initial population can be parallelized.
- In this study, we have discussed only the potential of parallel evaluating the objective function in all the schemes of sTLBO. However, other operations that can be potentially parallelized include (i) the generation of the initial population, (ii) the generation of random numbers required in the various stages, including the identification of random partners, teaching factors, and generation of potential members, (iii) the simultaneous generation of $3N_P$ learners in the class parallelized scheme, (iv) the simultaneous generation of two learners in the member parallelized scheme, (v) the simultaneous generation of N_P learners in the teacher phase as well as the generation of $2N_P$ learners in the learner phase of the phase parallelized scheme, (vi) determination of the population mean, (vii) determination of the teacher, (viii) sorting of the combined members in the three parallelized scheme, and (ix) bounding of the newly generated potential members. However, there might not be many benefits in parallelizing these operations for the moderately sized problems with high-powered processors. The primary reason for this is the overhead costs incurred in the parallelization may outweigh the reduction in time for performing these operations. Nevertheless, all these operations are potential candidates for parallelization.
- In the teacher and learner phases of all the parallelized schemes and sTLBO, the random number used for each decision variable is different while generating a new potential

solution. For example, consider an optimization problem with ten variables (N_d), and if the potential solution is generated using Eq. (9.1), then ten random numbers between 0 and 1 are required. Similarly, the teaching factor is unique for every subject, and ten random values of 1 or 2 will be required in Eq. (9.1). For the subsequent learner, the whole set of random numbers should be changed.

- It should be noted that Eq. (9.8) − (9.10) assumed that the time required for the evaluation of the objective function is 1 unit. If the time of evaluating the objective function for a single member requires N_T units of time, these equations can be modified into Eq.(9.11) − (9.13), respectively.

$$C_t^M = \left(\left\lceil \frac{N_P}{N_R} \right\rceil N_T\right) + \left(\left\lceil \frac{3}{N_R} \right\rceil N_T N_P N_G\right) \tag{9.11}$$

$$C_t^C = \left(\left\lceil \frac{N_P}{N_R} \right\rceil N_T\right) + \left(\left\lceil \frac{3N_P}{N_R} \right\rceil N_G N_T\right) \tag{9.12}$$

$$C_t^P = \left\lceil \frac{N_P}{N_R} \right\rceil N_T + \left(\left\lceil \frac{N_P}{N_R} \right\rceil + \left\lceil \frac{2N_P}{N_R} \right\rceil\right) N_T N_G \tag{9.13}$$

9.3. Experimental settings

This section demonstrates the efficacies of the four schemes on 14 benchmark functions that are widely used in literature (Ma et al., 2021; Mirjalili, 2015a; Mirjalili et al., 2014; Rao, Savsani, & Vakharia, 2012; Zhao et al., 2019). As the emphasis, we have chosen the same set of 14 benchmark functions that have been previously used by the inventors of TLBO (Rao, Savsani, & Vakharia, 2012) to compare the performance of parallelized techniques and sTLBO. Among the considered benchmark functions, thirteen problems are unconstrained optimization problems. One among the problems is a constrained optimization problem that has been converted into an unconstrained problem using a penalty approach. For the sake of brevity, the details of the benchmark problems are provided in Appendix J.

The class size for all four schemes is set to 100, and the number of generations for the three parallelized schemes is set to 100, whereas the number of iterations for the sTLBO is set to 150. This setting for the number of iterations ensures that the total number of function evaluations in all four schemes is identical (=30100) and compares all the algorithms fairly. Before establishing the benefits in computational time from these algorithms, it is necessary to determine the efficacy of these algorithms in determining the optimal solution. Due to the stochastic nature of the techniques, each benchmark problem is run 15 times with different random seeds, creating 210 (=15 x 14) unique instances. In addition to the 14 classical benchmark functions, the parallel variants and sTLBO performance are compared with other recently proposed algorithms on thirty real parameter single objective benchmark functions of CEC'14 test suite (J. J. Liang, 2013).

The recent algorithms considered for the performance analysis are the African vultures optimization algorithm (AVOA) proposed in 2021 (Abdollahzadeh et al., 2021), Mayfly algorithm (MA) proposed in 2020 (Zervoudakis & Tsafarakis, 2020), Moth-flame optimization (MFO) proposed in 2015 (Mirjalili, 2015b), Sine cosine algorithm (SCA) proposed in 2016 (Mirjalili, 2016), Spotted hyena optimizer (SHO) proposed in 2017 (Dhiman & Kumar, 2017) and Tree growth algorithm (TGA) proposed in 2018 (Cheraghalipour et al., 2018). The details of these techniques are briefly provided in Appendix A. This study uses default algorithm specific parameter values for all the algorithms, as provided by the respective inventors, and did not incorporate any parameter tuning. The population size for all algorithms is set to 100, and the termination criterion to 100,000 function evaluations to solve the CEC'14 test suite. The simulations are performed with MATLAB 2014b on an IBM System with Intel Xeon CPU E5-2630 @ 2.30 GHz machine.

9.4. Results and discussion

This section discusses the performance of three proposed parallel strategies of sTLBO on classical benchmark functions and the CEC'14 benchmark test suite. The benefits of parallel strategies over sTLBO are also analyzed based on the computational time by varying the resource availability and the complexity of the objective function.

9.4.1. Performance analysis on classical benchmark functions

The objective function value determined by the four schemes in each of the fifteen runs for all the fourteen problems is shown in Fig. 9.7. The absence of a bar corresponding to a particular scheme indicates that the specific scheme has determined zero as the optima. It can be observed from Fig. 9.7 that all the proposed schemes are able to determine an identical optimal solution of 3905.93 for the De Jong function and the optimal solution of 3 for the Goldstein and Price function. All four schemes have identified the optimal solution of zero for the Martin and Gaddy function, the B2 function, and the Booth function, and hence, the bar plots corresponding to these functions are not provided in Fig. 9.7. It is interesting to note that all four schemes can determine the global optima in all the 60 (15 Runs x 4 algorithms) instances for these three functions. For the Rosenbrock function, sTLBO outperforms the other three parallelized schemes, whereas MTLBO outperforms sTLBO in the sphere function and the Powell badly scaled function (except in run 11). These figures also help to estimate the deviation in the optimal solution determined by each algorithm. It can be observed that all four techniques have determined the identical optimal solution, indicating the three parallelized strategies do not compromise the optimality of the solution in their attempt to be computationally beneficial.

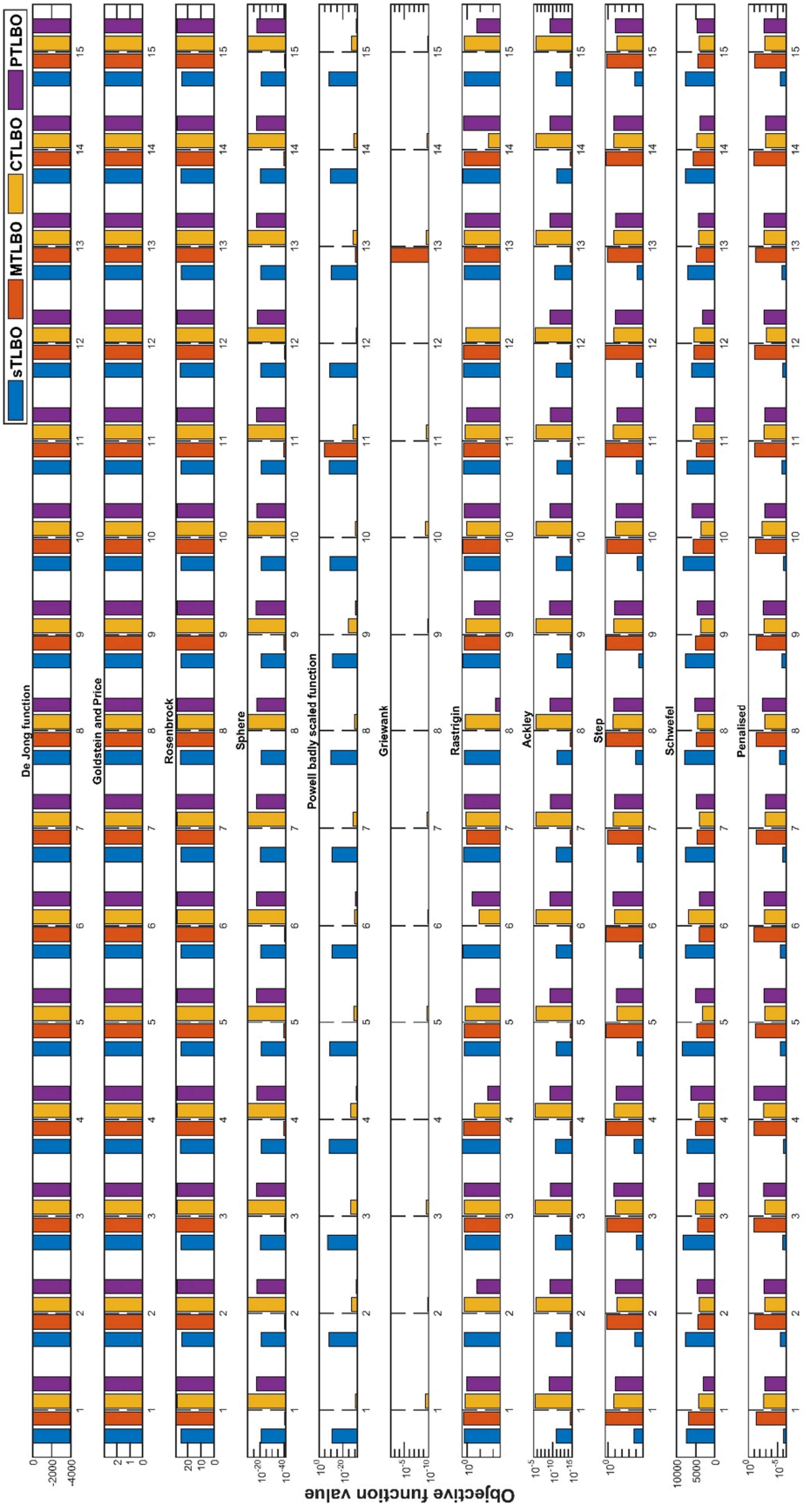


Fig. 9.7 Function values determined by all the four schemes

Ranking of the various schemes

For every problem, the scheme that determines the best fitness function value is given a better rank than the other schemes (within the specified number of function evaluations). Suppose two schemes have reported the same objective function value at the termination criterion. In that case, the tie is broken by identifying the number of function evaluations utilized by the respective schemes to determine the best solution. The scheme that determines the best solution in fewer function evaluations is assigned a better rank than the other schemes. If it happens to be a tie in this aspect, the scheme with a better average fitness value of the population at the end of all the functional evaluations is assigned a better rank. The schemes are provided with an identical rank if there is a tie in all these three aspects. The rank obtained in each run is summed up so as to determine the consolidated indication for the algorithm. Thus, a consolidated rank of 15 for a scheme on any benchmark problem would indicate that the scheme was able to determine the best optimal solution than other schemes in all 15 runs. Similarly, any scheme with a value of 60 has been ranked fourth in all the runs of that problem. Table 9.1 shows the ranking of the four schemes in all 14 benchmark problems.

It can be observed that sTLBO has been outranked by all the three parallelized schemes in eight problems (BF1, BF2, BF3, BF6, BF7, BF8, BF10, and BF13), whereas the sTLBO is better in three problems (BF4, BF12, and BF14). Moreover, MTLBO is able to outrank sTLBO in eleven of the fourteen problems (BF1, BF2, BF3, BF5, BF6, BF7, BF8, BF9, BF10, BF11, and BF13) while it is outranked by sTLBO in three benchmark problems. It should be noted that despite the high ranks of the proposed schemes for some problems, the optimal solution obtained by all three parallelized strategies is very close to the solution determined by sTLBO, as shown in Table 9.2. The difference in rank is because of the additional criteria in the ranking, such as the number of function evaluations in which the optimal solution is determined and the average objective value of the final population at the end of the entire algorithm.

Table 9.1 Ranking of algorithms

ID	Benchmark function	Consolidated Ranks			
		sTLBO	MTLBO	CTLBO	P TLBO
BF1	De Jong function	60	21	36	30
BF2	Goldstein and Price function	54	30	37	29
BF3	Martin and Gaddy function	60	15	45	30
BF4	Rosenbrock function	15	60	40	35
BF5	Sphere function	30	15	60	45
BF6	Powell badly scaled function	59	19	44	28
BF7	B2 function	60	15	45	30
BF8	Booth function	60	15	45	30
BF9	Griewank function	29	18	59	44
BF10	Rastrigin function	50	39	34	27
BF11	Ackley function	30	15	60	45
BF12	Step function	15	60	40	35
BF13	Schwefel function	60	33	25	32
BF14	Penalized function	15	59	35	41

The best, mean, and worst values of all the objective function value reported in every run for each benchmark problem is shown in Table 9.2. It can be observed from Table 9.2 that the mean, best, and worst values of all the algorithms are identical for five of the benchmark functions (BF1, BF2, BF3, BF7, and BF8). In the case of BF4, the mean objective function value of all runs determined by sTLBO is superior to the three parallelized schemes. However, in four benchmark functions (BF5, BF10, BF11, and BF13), the mean value is better for MTLBO than the sTLBO, whereas for BF6, the mean value determined by the CTLBO is better than the sTLBO. For BF9, all the statistical result of sTLBO is identical to the PTLBO. This analysis demonstrates that the proposed schemes are as effective as the sTLBO but have the added advantage of being computationally less expensive than sTLBO by harnessing the parallel computing benefits.

Table 9.2 Performance statistics of the four schemes with respect to convergence to the optimal solution

	sTLBO				MTLBO				CTLBO				PTLBO			
	Mean	Best	Worst	SD	Mean	Best	Worst	SD	Mean	Best	Worst	SD	Mean	Best	Worst	SD
BF1	-3905.9	-3905.9	-3905.9	9E-11	-3905.9	-3905.9	-3905.9	0	-3905.9	-3905.9	-3905.9	0	-3905.9	-3905.9	-3905.9	0
BF2	2.9999	2.9999	2.9999	1E-15	2.9999	2.9999	2.9999	7-16	2.9999	2.9999	2.9999	3E-16	2.9999	2.9999	2.9999	7E-16
BF3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
BF4	24.918	24.343	25.312	0.298	28.670	28.342	28.821	0.155	28.073	27.674	28.377	0.206	27.973	27.654	28.250	0.16
BF5	1E-21	5E-22	3.1E-21	8E-22	1E-40	6.2E-42	3.7E-40	1E-40	1.1E-10	4.1E-11	1.9E-10	4E-12	5E-18	1.6E-18	1.3E-17	3E-18
BF6	3.8E-9	6E-12	4.5E-8	1E-8	1.1E-6	0	1.6E-5	4E-6	9.7E-27	5E-32	1.3E-25	4E-26	4.1E-32	0	2.4E-31	8E-32
BF7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
BF8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
BF9	0	0	0	0	4.9E-4	0	0.0073	0.002	2.3E-10	1E-10	5.4E-10	1E-10	0	0	0	0
BF10	22.818	4.8419	89.469	21.37	12.145	0	42.896	11.80	5.347	6.4E-9	14.321	4.415	3.848	1.7E-13	18.909	5.96
BF11	1E-11	8E-12	3.3E-11	8E-12	2.6E-15	2.6E-15	2.6E-15	0	2.7E-6	1.9E-6	3.9E-6	6E-7	5.2E-10	4.0E-10	8.2E-10	1E-10
BF12	0.0001	9.3E-6	0.0002	6E-5	1.9611	1.119	2.7637	6E-5	0.1396	0.0596	0.2227	0.052	0.1157	0.0582	0.2311	0.049
BF13	7634.5	6053.5	8492.1	577.4	5025.4	4146.3	6858.1	665.7	4531.8	3200.4	6919.07	940.3	4627.04	3052.1	6320.0	880.0
BF14	1.9E-6	2.8E-7	4.8E-6	1E-6	0.0741	0.0406	0.1156	0.025	0.0017	0.0008	0.0041	8E-4	0.0087	0.0009	0.1054	0.027

9.4.2. Performance analysis on CEC'14 test suite

In order to demonstrate the efficacy of the proposed parallel variants, the statistical analysis of the error values on solving each function of CEC'14 benchmark test suite is compared with six recently proposed metaheuristic techniques. The detailed tables depicting the best, worst, mean, and median of the error values obtained in each function are given in Appendix K. The Wilcoxon signed-rank test (Gibbons & Chakraborti, 2011) is a commonly used hypothesis test to analyze whether the results provided by two algorithms are statistically different with a significance level (α) or not. If the p-value reported by a paired sample is less than the α-value (0.05), the null hypothesis that the two samples are derived from the same population is rejected. The R+ value is the sum of ranks provided to the positive differences of the samples, while R− indicates the sum of ranks obtained to the negative differences of the samples. Table 9.3 provides the results of the Wilcoxon signed-rank test performed between the proposed variants and the other six metaheuristic algorithms using the mean fitness values as the samples. The test results with a p-value less than α-value are highlighted using bold type-face font, and the corresponding sum of rank with a higher value is also highlighted.

Table 9.3 Results of Wilcoxon signed-rank test for the CEC-14 test suite

Algorithms	sTLBO			CTLBO			MTLBO			PTLBO		
	R+	R−	p-value	R+	R−	p-value	R+	R−	p-value	R+	R−	p-value
AVOA	116	**349**	**1.66E-02**	61	**404**	**4.20E-04**	276	189	3.71E-01	94	**371**	**4.39E-03**
MA	148	317	8.22E-02	123	**342**	**2.43E-02**	223	242	8.45E-01	125	**340**	**2.70E-02**
MFO	49	**416**	**1.60E-04**	15	**450**	**7.69E-06**	173	292	2.21E-01	23	**442**	**1.64E-05**
SCA	0	**465**	**1.73E-06**	0	**465**	**1.73E-06**	79	**386**	**1.59E-03**	0	**465**	**1.73E-06**
SHO	66	**399**	**6.16E-04**	65	**400**	**5.71E-04**	85	**380**	**2.41E-03**	65	**400**	**5.71E-04**
TGA	37	**428**	**5.79E-05**	36	**429**	**5.31E-05**	105	**360**	**8.73E-03**	57	**408**	**3.06E-04**

The R− value is the sum of ranks in which sTLBO (or its variants) outperforms other algorithms. The p-values obtained by CTLBO and PTLBO in each pair-wise comparison indicate that the results provided by these algorithms are significantly different, and the R− value of CTLBO and PTLBO demonstrate their significantly better performance with respect

to the other six algorithms. It can also be observed that sTLBO and three proposed parallel variants have significantly outperformed SCA, SHO, and TGA. The R+ value provided by SCA in comparison to sTLBO, CTLBO and PTLBO is zero indicating its inferior performance in all functions.

The performance of the proposed variants is compared with the winner algorithms (Brest et al., 2020; Mohamed et al., 2020; Sallam et al., 2020) of CEC 2020 competition on the real parameter bound-constrained benchmark test suite using the Wilcoxon signed-rank test. The analysis is performed between the proposed variants, five recently proposed metaheuristic algorithms, and the CEC 2020 winner algorithms and obtained result is provided in Appendix K. It is observed that the proposed variants have demonstrated competitive performance with all the algorithms other than CEC 2020 winner algorithms. In view of the "No free lunch theorem of optimization" (Wolpert & Macready, 1997), the efficacy of the proposed variants and other algorithms are further investigated on the single-level production planning problem (Chauhan & Kotecha, 2020) with maximization of profit as the objective function and the statistica analyzis is provided in Appendix K. It has been observed that the proposed variants have provided a superior performance in comparison to the majority of the other seven algorithms.

9.4.3. Analysis of computational time

This section demonstrate the benefits of three parallelized schemes in reducing the computational time compared to sTLBO. It can be observed that the benchmark functions do not require significant time for a single evaluation of the objective function. A standard technique that has been widely employed in literature (Arora et al., 2010; Stuart Kozola, 2009) is to artificially add a time delay in the objective function to demonstrate the benefits of the parallelized schemes. In this study, we employ this strategy used in literature by utilizing the

pause function in MATLAB. This section demonstrates the effect of an increase in the computational time to evaluate the objective function as well as the impact of increasing the number of simultaneous parallel resources available for evaluating the objective function.

The impact of available resources on the computational time for function evaluation is studied for all four schemes using the Rastrigin function. In this analysis, the population size is set to 100, whereas the number of generations for sTLBO and all parallelized strategies is considered as 150 and 100, respectively. The results of all four schemes are provided in Table 9.4. For all four schemes, simultaneous objective function evaluation of the initial population is not performed as it can potentially lead to unnecessary overheads in the algorithmic phase. Hence, there will be no effect of the number of resources on sTLBO, and only a single value of computational time has been reported in the table for the sTLBO. Similarly, there will be no effect on MTLBO for evaluating the objective function as $N_R > 3$; hence, the result is provided only for $N_R = 2$ and $N_R = 3$.

Table 9.4 Performance of the schemes with respect to the number of resources

		Computational time required (in seconds)				% Reduction over sTLBO		
	N_R	sTLBO	MTLBO	CTLBO	PTLBO	MTLBO	CTLBO	PTLBO
Pause of 0.5 secs	2	15,265.26 (with N_R=1)	10,952.24	7,643.81	7,654.73	28.25	49.92	49.85
	3		5,929.36	5,254.58	5,266.02	61.15	65.57	65.50
	4			4,033.10	3,991.21		73.57	73.85
	6			2,660.54	2,718.61		82.57	82.19
	8			2,050.92	2,012.37		86.56	86.81
	10			1,645.19	1,611.78		89.22	89.44
	12			1,391.49	1,406.7		90.88	90.78
Pause of 2 secs	2	60,573.66 (with N_R=1)	41,178.30	30,299.82	30,310.87	32.01	49.97	49.96
	3		21,069.33	20,808.22	20,818.91	65.21	65.64	65.63
	4			15,961.86	15,770.32		73.64	73.96
	6			10,514.22	10,723.97		82.64	82.29
	8			8,089.90	7,899.23		86.64	86.95
	10			6,474.53	6,288.97		89.31	89.61
	12			5,465.25	5,478.76		90.97	90.95

From Table 9.4, it can be observed that all the strategies require significantly lower computational time than sTLBO. Moreover, with increased resources, the computational time becomes significantly lower for the class and phase parallelized schemes. Among CTLBO and PTLBO, there is no significant time difference for the selected algorithm parameters and resources. It can be noticed that the proposed class parallelized and phase parallelized techniques can reduce the computational time required by up to 90%. For all three parallelized schemes, the percentage reduction in computational time increases with increased parallel resources, thereby significantly harnessing parallel computing benefits.

The rastrigin function is optimized to study the effect of the time required for the objective function evaluation on the four schemes with six simultaneous parallel resources to evaluate the objective function with the same algorithm settings considered in the previous analysis. Table 9.5 shows the effect of the variation of the artificial time delay in the objective function on the computational time required by the four schemes to determine the optimal solution.

Table 9.5 Performance of schemes with respect to the complexity of the objective function

Pause time (in seconds)	Computational time required (in seconds)				% Reduction over sTLBO		
	sTLBO	MTLBO	CTLBO	PTLBO	MTLBO	CTLBO	PTLBO
0.05	3,733.83	1,468.63	337.47	350.73	60.67	90.96	90.60
0.10	5,183.45	1,978.26	582.84	602.10	61.84	88.75	88.38
0.25	9,507.27	3,434.52	1,368.24	1,399.38	63.87	85.61	85.28
0.50	17,328.40	5,992.06	2,663.21	2,719.55	65.42	84.63	84.31
0.75	24,993.44	8,551.60	4,001.03	4,084.14	65.78	83.99	83.66
1.00	32,590.54	11,091.51	5,304.11	5,412.03	65.97	83.73	83.39

It can be observed that the amount of computational time required by the three strategies is significantly less than the sTLBO. Though the time required by the class parallelized TLBO is less than the phase parallelized TLBO, the time difference between these two algorithms is not substantial for the chosen set of parameters. The difference between the time required in the sTLBO for 0.5 seconds in Table 9.4 and Table 9.5 can be attributed to the fact that in the case of Table 9.5, we had initiated six workers, which leads to overhead without necessarily

benefiting in reducing the time required for evaluation of the objective function in the algorithmic phase. Moreover, it can be observed that the class and phase parallelized schemes are able to reduce the computational time up to 90 %, whereas the member parallelized scheme is able to reduce the computational time to around 66% and helps to harness the benefits of parallel computing.

9.5. Conclusion

In this chapter, three different parallelized schemes of sTLBO have been proposed that overcome the limitation of sTLBO for computationally intensive optimization problems. These schemes can harness the benefits of parallel computing in varying degrees to solve computationally intensive problems in comparatively lower computational time. The efficacy of the schemes in terms of their ability to determine the optimal solution has been demonstrated on classical benchmark functions and CEC'14 real parameter benchmark functions. It is observed that these schemes are able to determine the optimal solution in much less time and are potential candidates to be applied to computationally intensive problems. The effect of the number of simultaneous resources for evaluating the objective function and computational time required for a single evaluation of the objective function has also been studied.

Chapter 10

A note on multi-objective improved teaching–learning based optimization

This chapter critically analyzes the multi-objective variant of improved teaching learning based optimization (MO-ITLBO) (Patel & Savsani, 2014a; Patel & Savsani, 2016; Rao & Patel, 2014) and highlights the various issues in the steps employed in extending the Improved Teaching Leaning based Optimization to its multi-objective variant. The MO-ITLBO is proposed in three articles (Patel & Savsani, 2014a; Patel & Savsani, 2016; Rao & Patel, 2014) for solving different optimization problems. The steps provided for implementing MO-ITLBO are imprecise and lead to multiple variants of the same algorithm. In this regard, this chapter proposes two variants of MO-ITLBO, considering the possibilities to overcome the ambiguity in the implementation, and compared their performance with the results provided by MO-ITLBO on the same benchmark functions.

This chapter is structured as follows: Section 10.1 explains the MO-ITLBO and various issues in its implementation. Section 10.2 describes two variants of MO-ITLBO. Section 10.3 provides the experimental settings for analyzing the performance of the two variants. Section 10.4 analyzes the results obtained by variants and discusses their performance. Section 10.5 concludes this chapter by consolidating major findings of this work.

10.1. Multi-objective Improved Teaching Learning Based Optimization and its issues

The MO-ITLBO proposed in the literature comprises six basic steps, namely, (i) initialization, (ii) selection of teachers, (iii) assigning learners to teachers, (iv) teacher phase, (v) learner phase, and (vi) external archive. In the initialization phase, the details of the optimization problems (in terms of design variables and their bounds, the objective functions) along with the necessary algorithmic parameters (such as population size, the number of teachers, details of the external archive, and the termination criteria) are specified. In the second step, i.e.,

Chinta, S., R. **Kommadath,** and P. Kotecha, *A note on multi-objective improved teaching–learning based optimization algorithm (MO-ITLBO).* Information Sciences, 2016. **373**: p. 337-350. DOI: https://doi.org/10.1016/j.ins.2016.08.061

selection of the teacher, the teachers are selected from the population based on the value of the objective function. The best solution in the class is considered the chief teacher, and based on its objective function value, the teachers of each group are determined. Though such a strategy might be simple in the context of single-objective optimization, the presence of non-dominated solutions in the context of multi-objective optimization needs to be carefully addressed. In the third step of the algorithm, the solutions in the population (students) are assigned to various teachers depending on their objective function value. In the teacher phase, every student in a group learns from the teacher and a randomly selected solution in the group through a tutorial. In this step, a new solution is generated that replaces the old solution if it has a better objective function value. In the learner phase, every student of the population undergoes learning by interacting with another randomly selected solution of the same group and also through self-motivated learning. Similar to the teacher phase, if the new potential solution is better than the old one, it replaces the old one in the current population. The final step in every iteration is updating the archive with the best solutions based on the ε-dominance method. The sub-sections provided below explain different issues involved in various steps of MO-ITLBO. It should be noted that these issues could be understood only if a reader has studied MO-ITLBO, provided by Patel and Savsani (2016), Rao and Patel (2014), or Patel and Savsani (2014a).

10.1.1. Selection of teachers

This step includes determining the chief teacher and teacher for each group. However, it is difficult to implement both steps due to a lack of clarity in the explanation of MO-ITLBO.

Determination of chief teacher: As shown below in bold letters (Patel & Savsani, 2014a; Patel & Savsani, 2016; Rao & Patel, 2014), the ITLBO algorithm considers the best student of the class as the best or the chief teacher. For a single objective optimization problem (SOOP), it is straightforward to determine the solution with the best objective function value (and in instances of multiple solutions having an identical best value, randomly select one among

them). However, for a multi-objective optimization problem (MOOP), there may not be a single best solution as the entire set of non-dominated solutions constituting the Pareto front are the best solutions.

"Select the best solution (i.e. the solution obtained the first rank). This solution acts as the chief teacher of the class, i.e., $T_1 = f(X_b)$ "

The article in the literature does not mention any strategy to determine the best solution, which is crucial for the working of the algorithm. The following are some possible strategies that could be intuitively implemented to address this issue.

a) Randomly select one of the Pareto points as the best student (chief teacher). This may result in improving the mean of the class towards a particular region of the non-dominated front depending on the location of the selected Pareto point.

b) Randomly select one of the objectives and consider the best solution in the class for that objective as the best student (chief teacher). This may result in improving the mean of the class towards a corner of the non-dominated front depending upon the objective function that is selected. If multiple students have an identical objective function value for the chosen objective, the value of the other objective function can be used to break the tie. In the case of identical sets of objective function values for all the objectives, a random solution among these can be chosen.

c) One of the anonymous reviewers of this work has suggested another strategy in which the first member is considered the best solution, which can be compared with the subsequent solutions of the population. If any of the subsequent solutions dominate it, the dominating solution can be considered as the current best solution. This procedure is continued in which the current best solution is compared with the other remaining solutions. In this procedure, either the first non-dominated population member becomes the chief teacher or a completely dominating solution (if it exists) becomes the chief teacher.

Determination of group teachers: Assuming there is a strategy to determine the chief teacher (T_1) for MOOP, the algorithm requires the identification of members who can act as teachers for other groups. For solving SOOP, the strategy suggested (Patel & Savsani, 2014a; Patel & Savsani, 2016; Rao & Patel, 2014) is as follows

"Select the other teachers based on the best solution (i.e. $f(X_b)$)

$$T_s = f(X_b) \pm r_i f(X_b) \qquad s=2,3,..,N$$

(If the equality is not met, select the T_s closer to the value calculated above)"

Even for SOOP, it is difficult to obtain a value of T_s that is exactly identical to the objective value of one of the solutions. Hence, the solution with the objective function value close to T_s is considered as the teacher for a group. However, the article does not specify the procedure for determining group teachers in the context of MOOPs. Also, the article in the literature does not explicitly provide any strategy to determine "closeness" in the context of multiple objectives. If this is based on a norm, no information has been provided about the norm that was chosen to produce the results stated in the article. In the absence of any "closeness" measure, the group teachers can be directly selected using any of the following options.

a) Select to have as many groups as the number of Pareto points and consider each Pareto point to be a teacher for one group. Potential issues with such an approach could include a large number of groups but with very few students in each group. It might happen that there are only a few Pareto points in earlier stages, and other solutions of the group may subsequently improve to become a non-dominated solution. However, with the progress of generations, all solutions may become non-dominated solutions, with each solution becoming an individual group, and such a scenario might not lead to an improvement in determining the global Pareto front.

b) Randomly divide the population into a specified number of groups and consider one of the non-dominated solutions in the group to act as the teacher of that group. Such a strategy

may lead to a situation wherein one of the groups may potentially have a significant number of non-dominating points (from the Pareto front). In contrast, groups with less or even zero non-dominating points may exist. Suppose a group has a large number of non-dominating solutions. In that case, the solutions may spread along the Pareto front, whereas in the presence of fewer non-dominating solutions in a group, the other dominated solutions of the group might improve to become non-dominating by the learning process. However, if there are no non-dominated solutions in a group, the contribution of this group to the global Pareto front could be minimal.

10.1.2. Assigning learners to a teacher

In this stage, each solution in the population is assigned to one of the pre-specified groups whose teachers were determined in the previous step. In SOOP, given the objective function values of the teacher, the students whose objective function value lies between two teachers are assigned to one group (as shown below (Patel & Savsani, 2014a; Patel & Savsani, 2016; Rao & Patel, 2014)).

" For $k = 1$ *to Population*

If $T_1 \le f\left(X^k\right) < T_2$

Assign the learner $f\left(X^k\right)$ *to teacher 1* $\left(i.e\ T_1\right)$

Else If $T_2 \le f\left(X^k\right) < T_3$

Assign the learner $f\left(X^k\right)$ *to teacher 2* $\left(i.e\ T_2\right)$

.....

Else If $T_{N-1} \le f\left(X^k\right) < T_N$

Assign the learner $f\left(X^k\right)$ *to teacher N-1* $\left(i.e\ T_{N-1}\right)$

Else

Assign the learner $f\left(X^k\right)$ *to teacher N* $\left(i.e\ T_N\right)$

End If

End For"

In SOOP, the learners can be assigned to appropriate groups by sorting the objective function values. However, in the case of MOOP, the sorting of the objective function values is not possible due to the presence of multiple conflicting objective functions. For example, a student may be assigned to a group based on one of the objectives, and the same student may be assigned to another group based on the other objective. Thus, the strategy for assigning learners to various groups is unclear. Despite the critical nature of this step, the article does not provide any details about the strategy used for assigning students to various groups in the context of MOOP. A potential strategy for assigning learners to different groups can be randomly dividing the Pareto points in a given generation into the specified groups and similarly randomly distributing the dominated members into each of the various groups.

10.1.3. Teacher phase

The teacher phase involves the generation of potential solutions through the modification of each solution of every group based on (i) the decision variables of the teacher of the group, (ii) the mean of the decision variables of the solutions in the group, (iii) decision variables of a random solution in the group and (iv) decision variables of the current solution.

Modifying student: For SOOP, it has been proposed (in ITLBO (Patel & Savsani, 2014a; Patel & Savsani, 2016; Rao & Patel, 2014)) that the modified solution should be determined using one of the following equations.

$$\text{"}For\ j\ =\ 1\ to\ No.\ of\ Design\ variables$$

$$X_j^k = \left(X_j^k + Difference\_Mean_{s,j}\right) + r_i\left(X_j^h - X_j^k\right) \quad if\ f\left(X^h\right) < f\left(X^k\right), h \neq k$$

$$X_j^k = \left(X_j^k + Difference\_Mean_{s,j}\right) + r_i\left(X_j^k - X_j^h\right) \quad if\ f\left(X^k\right) < f\left(X^h\right), h \neq k$$

$$End\ For\text{"}$$

In the context of MOOP, it is possible that students *h* and *k* are non-dominating. In that scenario, which of the above mathematical expressions would be applicable is unclear. Some possible options to handle such a scenario include

a) Do not modify the solution k if h and k are non-dominating. This may cause the algorithm to use a larger number of function evaluations to converge to the final Pareto front, as some function evaluations are not being utilized.

b) Select another random solution of the group h' such that h' and k are not non-dominating. Such a strategy can potentially lead to faster convergence as one of the interacting solutions is always a dominating solution and can improve the other solution. If there is no such solution, then that solution in the group may not be modified. It should be noted that such a non-dominating solution would always be available if the 1B.1 strategy is selected. However, this could lead to convergence to a suboptimal solution if all the members of a particular generation lie on the local Pareto front. Moreover, it may be computationally expensive to determine a random solution in the group that is dominating with the current solution, as this operation would be required for each solution of every group for every iteration. Alternatively, one could choose to generate two potential students (using both the above equations) and subsequently employ some strategy to select the best among them.

Updating students: The article (Patel & Savsani, 2014; Patel & Savsani, 2016; Rao & Patel, 2014) proposes the following strategy to handle the newly generated solution.

" If the result has improved
Keep the improved result
Else
Keep the previous result
End If "

In such a strategy, if the modified solution (X^{mod}) dominates (X^k), then (X^k) is to be replaced by (X^{mod}). But if (X^{mod}) is dominated by (X^k) then (X^{mod}) is discarded. However, the article does not discuss how MO-ITLBO handles a scenario in which (X^k) and (X^{mod}) are non-dominating. One of the anonymous reviewers has pointed out that in a multi-objective perspective, "*if the result has improved*" can be interpreted as a solution that is "*totally dominating*", i.e., a solution

better in all objectives. Hence, it is necessary to clarify the concept "*if the result has improved*" corresponds to a non-dominated or totally dominating solution.

Adaptive teaching factor: In SOOP, there is a single objective function value, and therefore, it is straightforward to determine the adaptive teaching factor (Patel & Savsani, 2014a; Patel & Savsani, 2016; Rao & Patel, 2014) using the following equation.

$$"\left(T_F\right)_{s,i} = \begin{cases} \left(\dfrac{f\left(X^k\right)}{T_s}\right)_i & if\ T_s \neq 0 \\ 1 & if\ T_s = 0" \end{cases}$$

The articles (Patel & Savsani, 2014a; Patel & Savsani, 2016; Rao & Patel, 2014) do not mention which objective function value should be used to determine the teaching factor.

Updating teacher: It can be noted that the decision variables corresponding to the teacher of a particular group can potentially be modified in the teacher phase (as the teacher also undergoes the teacher phase in TLBO), and it might also happen that one among the new solutions of the group may become the best in the group. The article does not clearly describe whether this modified solution, which might be superior to all others in the group at the end of the teacher phase, becomes the new teacher for the learner phase of the group or whether the current teacher continues to be the teacher for the learner phase.

10.1.4. Learner phase

In SOOP, the following equations in ITLBO (Patel & Savsani, 2014a; Patel & Savsani, 2016; Rao & Patel, 2014) help determine a modified student.

$$"For\ j\ =\ 1\ to\ No.\ of\ Design\ variables$$

$$X'^{p}_{j,i} = \left(X'^{p}_{j,i} + r_i\left(X'^{p}_{j,i} - X'^{q}_{j,i}\right)\right) + r_i\left(X^{s}_{j,i} - E_F X'^{p}_{j,i}\right) \quad if\ f\left(X'^{p}\right) < f\left(X'^{q}\right)$$

$$X'^{p}_{j,i} = \left(X'^{p}_{j,i} + r_i\left(X'^{q}_{j,i} - X'^{p}_{j,i}\right)\right) + r_i\left(X^{s}_{j,i} - E_F X'^{p}_{j,i}\right) \quad if\ f\left(X'^{q}\right) < f\left(X'^{p}\right)$$

$$\left(p \neq q\ and\ p,q,s \in k,\ X^{s}_{j}\ is\ the\ grade\ of\ the\ teacher\ associated\ with\ group\ 's'\ in\ 'j'\ subject\right)$$

$$End\ For"$$

However, in MOOP, it is possible that the students p and q are non-dominating, and thus, neither of the above equations would be applicable (as discussed in the teacher phase). However, the article (Patel & Savsani, 2014a; Patel & Savsani, 2016; Rao & Patel, 2014) does not provide any strategy to handle such circumstances. In this scenario, the following multiple options are possible.

i. If X'^p and X'^q are non-dominating, do not generate a modified student.

ii. Select another random member of the group q' such that q' and p are not non-dominating.

In case there is no such solution, then the solution p may not be modified.

Additionally, the following strategy (Patel & Savsani, 2014a; Patel & Savsani, 2016; Rao & Patel, 2014) is used to handle the newly modified solution.

" If the result has improved
Keep the improved result
Else
Keep the previous result
End If "

However, this strategy also suffers from the same drawbacks that have been explained for updating students in the teacher phase.

10.1.5. Adaptive teaching factor and exploration factor

In the basic TLBO (R. V. Rao et al., 2012; R. V. Rao et al., 2012; Rao et al., 2011) and elitist TLBO (Rao & Patel, 2012), the teaching factor was randomly fixed to either 1 or 2 using the following relation.

$$" T_F = round\left(1 + r_i\right) "$$

However, it has been suggested in the literature (Patel & Savsani, 2014; Patel & Savsani, 2016; Rao & Patel, 2014) that T_F can be determined using the following relation to cater to the scenario wherein the learner can learn in varying levels.

$$''\left(T_F\right)_{s,i} = \begin{cases} \left(\dfrac{f\left(X^k\right)}{T_s}\right)_i & if\ T_s \neq 0 \\ 1 & if\ T_s = 0'' \end{cases}$$

It would have been more informative if the articles (Patel & Savsani, 2014a; Patel & Savsani, 2016; Rao & Patel, 2014) had provided an insight into the benefit of such a modification in the basic or elitist TLBO. In ITLBO, multiple modifications have been incorporated, but the benefit of each modification is not implicitly clear. Moreover, given the above modification in handling the teaching factor, the articles do not provide any reason for fixing the value of the exploration factor $\left(''E_F = round\left(1+r_i\right)''\right)$, which is still restricted to possess a value of either one or two, used in the following equations.

$$''For\ j\ = 1\ to\ No.\ of\ Design\ variables$$

$$X'^{p}_{j,i} = \left(X'^{p}_{j,i} + r_i\left(X'^{p}_{j,i} - X'^{q}_{j,i}\right)\right) + r_i\left(X^{s}_{j,i} - E_F X'^{p}_{j,i}\right) \quad if\ f\left(X'^{p}\right) < f\left(X'^{q}\right)$$

$$X'^{p}_{j,i} = \left(X'^{p}_{j,i} + r_i\left(X'^{q}_{j,i} - X'^{p}_{j,i}\right)\right) + r_i\left(X^{s}_{j,i} - E_F X'^{p}_{j,i}\right) \quad if\ f\left(X'^{q}\right) < f\left(X'^{p}\right)$$

$$\left(p \neq q\ and\ p,q,s \in k,\ X^s_j\ is\ the\ grade\ of\ the\ teacher\ associated\ with\ group\ 's'\ in\ 'j'\ subject;\ \right)$$

$$End\ For''$$

It would have been useful if the article provided theoretical/logical arguments or even computational results for modifying the strategy such that T_F can have any value greater than one whereas E_F is still chosen to be either 1 or 2.

10.1.6. Miscellaneous issues

The other issues observed in MO-ITLBO explained in articles (Patel & Savsani, 2014a; Patel & Savsani, 2016; Rao & Patel, 2014) are detailed below.

External archive: The extension of ITLBO to MOOP is based on maintaining an external archive using the ε-domination approach of Deb et al. (2005). The work of Deb et al. (2005) explicitly specified the value of ε for the benchmark problems that were used to determine the

efficacy of their algorithm. However, the article does not mention the value of ε, which plays a crucial role in determining the global Pareto front.

Removal of duplicate solutions: In the basic TLBO, the step showing the removal of duplicate solutions was not explicitly specified (Rao et al., 2011). However, this step was identified in a code review and has become a point of contention (Črepinšek et al., 2012; Waghmare, 2013). In view of this, it will be worthwhile to know if the authors employed a partial or complete duplicate removal step in MO-ITLBO. As the duplicate removal step requires the evaluation of the objective function, which in turn governs the termination of the algorithm, it is necessary to explicitly know the use of the duplicate removal step in MO-ITLBO.

Updating population: The work of Črepinšek et al. (2012) has shown that there were discrepancies in the reporting and the actual implementation of TLBO (Rao & Patel, 2012; Rao et al., 2011). For example, the original TLBO article (Rao et al., 2011) showed that all the solutions in the class complete the teacher phase and subsequently undergo the learner phase. Some researchers still use this approach (Dede & Ayvaz, 2015; Zou et al., 2013). However, in the actual implementation of code (Rao & Patel, 2012), each solution in the class undergoes the teacher and the learner phases, followed by updating the mean and the best solution of the class. This issue also exists in MO-ITLBO, and it is not explicitly specified if a solution in the group undergoes the teacher phase followed by the learner phase or if the entire group completes the teacher phase before undergoing the learner phase.

Computational time: In the absence of duplicate removal, the number of objective function evaluations in TLBO and ITLBO is identical, i.e., $N_c\left(1+2N_G\right)$ where N_G and N_c are the iteration and class size, respectively. However, for the large dimension problems (because of the additional operations involved in determining potential new students in the teacher and learner phases), the computational time required in ITLBO could be higher than TLBO.

Number of variables: The CEC 2009 problems (Zhang et al., 2008) are generic and can be solved for any number of variables. However, CEC 2009 competition requires the participants to demonstrate the performance of their algorithm with 30 variables. The article (Patel & Savsani, 2016; Rao & Patel, 2014) does not seem to specify the number of variables used in the benchmark problems.

Random seed: The random number generator can have a considerable impact on the performance of an algorithm, and hence, it is usually required to specify the random number generator and the seed (Črepinšek et al., 2012; Liang et al., 2013; Suganthan et al., 2005). For example, interested readers can execute the publicly available NSGA-II algorithm (https://bit.ly/41diBfz) to observe two completely different Pareto fronts for the same problem with different random values for the seed. Thus, it would have been beneficial if the article had specified the seeds (and their selection for multiple runs) and the algorithm used to generate the random numbers.

Runs and plots: The article mentions that each of the 20 problems was run for 30 runs, and the mean Inverse Generational Distance (IGD) and Standard Deviation (SD) have been reported in Table 1 – Table 3 of Patel and Savsani (2016). However, it is not clear as to which of these 30 runs have been plotted in Figure 2 – Figure 5 in Patel and Savsani (2016) and in Figure 3 – Figure 4 in Rao and Patel (2014).

Code availability: When all the steps implemented in an algorithm to generate the results reported in an article are not described in detail, it could often lead to difficulties in reproducing the results. In view of this, researchers proposing novel algorithms are usually encouraged to make the codes publicly available (Barnes, 2010; Deb et al., 2005; Mernik et al., 2015). Also, this enables members of the scientific community to use the algorithm for various applications. The code of ITLBO and MO-ITLBO are not publicly available making it difficult to understand some concepts or even adapt it for other applications.

Notations and schematic diagrams: One of the anonymous reviewers has indicated the fact that the symbol for the objective function in Figure 1 (Patel & Savsani, 2016) (and Figure 2 in Rao and Patel (2014) and Figure 2 in Patel and Savsani (2014a)) should be in "non-boldface fonts" so as to denote multi-objective optimization appropriately. In its current form, it seems that Figure 1 (Patel & Savsani, 2016) (and Figure 2 in (Rao & Patel, 2014) and Figure 2 in (Patel & Savsani, 2014a)) has been wrongly provided in the article instead of a figure corresponding to multi-objective optimization.

This work has raised some critical issues that need to be thoroughly addressed to harness the benefits of MO-ITLBO. Without any clarifications on these issues, it does not seem possible for independent researchers to test or use MO-ITLBO.

10.2. Implementation of the proposed variant I and variant II

In view of the issues mentioned in the previous section, it is impossible to implement the MO-ITLBO proposed in the literature (Patel & Savsani, 2016; Rao & Patel, 2014). However, to show that these issues are critical to the performance of the algorithm, we have implemented two variants and compared their performance to that reported in the literature (Patel & Savsani, 2016; Rao & Patel, 2014). It should be noted that this section has been included despite the explicit knowledge that the results reported in this section are inferior to those reported in the literature. Nevertheless, this exercise has been reported to emphasize the lacunae of the information provided in the MO-ITLBO and its impact. The following preliminary discussion will be used in both variants.

Generation of new solutions: There are two equations in the teacher phase and two equations in the learner phase which are employed in MO-ITLBO (as well as single objective ITLBO) to generate new solutions (Patel & Savsani, 2014a; Patel & Savsani, 2016; Rao & Patel, 2014). The selection of the equation is based on the value of the objective function of the two solutions

involved in the equation. However, the following generic equation can concisely express all four equations, which can be appropriately modified to realize the four equations.

$$Xnew(i) = X_g(i) + \left(X_{best}(i) - F(i)Y(i)\right)r_i + a_g r_i'\left(X_p(i) - X_g(i)\right) \quad \forall i = 1,2,..,V \tag{10.1}$$

In this equation, $Xnew(i)$, $X_g(i)$ and $X_p(i)$ indicate the value of the i^{th} decision variable of the new potential solution, the current member undergoing the teacher phase or the learner phase, and the random partner of the current member of the group, respectively. The terms r_i and r_i' indicate a vector of random numbers generated in between 0 and 1. It should be noted that for every use of the equation, the random numbers are to be repeatedly generated. The values of F, Y and a_g can be appropriately modified to obtain all the four equations used in MO-ITLBO and will be specified in the description of the algorithm.

Bounding of new solutions: The equations used in ITLBO (Rao & Patel, 2013) as well as MO-ITLBO (Patel & Savsani, 2014a, 2014b; Patel & Savsani, 2016; Rao & Patel, 2014) do not ensure that the newly generated solution is within the bounds of the decision variables. However, the bounding strategy employed either in the single objective ITLBO (Rao & Patel, 2013) or MO-ITLBO (Patel & Savsani, 2014a; Patel & Savsani, 2016; Rao & Patel, 2014) has not explicitly specified. In both variants, the following bounding strategy has been employed.

$$\begin{aligned} Xnew(i) &= \max\left(Xnew(i), L(i)\right) \qquad \forall i = 1,2,..,V \\ Xnew(i) &= \min\left(Xnew(i), U(i)\right) \qquad \forall i = 1,2,..,V \end{aligned} \tag{10.2}$$

This bounding strategy ensures that a variable is set to its lower value if it violates the lower bound and is set to its upper value if it violates the upper bound. The notation V denotes the number of decision variables with upper and lower bounds given by U and L, respectively. Both variants are proposed for the termination criteria of a specified number of iterations, and they can be suitably modified so as to incorporate the termination criteria of a specified number

of functional evaluations. All steps of both variants are described in detail to avoid the scope for ambiguity and enable independent implementation and verification of the reported results. For the sake of simplicity, the proposed variants assume that $S = N/G$ is an integer, where N and G correspond to the class size and group size, respectively. If the number of students per group (S) is not an integer, then $S = \lfloor N/G \rfloor$ and the remaining students can be randomly assigned to any of the G groups (with at most one of the remaining students assigned per group). Unlike the MO-ITLBO proposed in the literature (Patel & Savsani, 2014a; Patel & Savsani, 2016; Rao & Patel, 2014), which does not explicitly utilize the solutions of the archive in the generation of the new solution, both variants use at least some of the members of the archive in every generation as teachers of the individual groups.

Variant I: Solutions generated using the dominated student

In this variant, the students are randomly assigned to a specified number of groups. A random non-dominated solution determined using ε-dominance method is assigned to each group to act as the teacher of the group. This is similar to the ε-dominance based GA (Deb et al., 2005), which uses one of the solutions from the archive as a partner in the crossover operation. This also ensures that the teacher of the group is always a non-dominated solution, and the members of the group are expected to benefit from the teacher. For each member of the group, a partner is randomly selected from the group. A new solution is generated using the appropriate equation if either of the solutions dominates the other. If both the solutions are non-dominated, the equation to generate a new solution is randomly selected with equal probability. This newly generated solution is compared to each member of the group and replaces the first member, which is dominated by the new solution; otherwise, the new solution is discarded. This process is repeated for each member of all the groups to complete the teacher phase. In view of the discrepancy in determining the teaching factor for MOOP, the teaching factor is selected as in the basic TLBO and is thus restricted to either 1 or 2. Similar to the teacher

phase, the student phase uses the appropriate equation if one of the members dominates the other. If both are non-dominated solutions with respect to each other, the equation is selected randomly but with equal probability. Thus, this strategy ensures that the procedure stated in MO-ITLBO is used for generating the solutions in instances wherein there is no ambiguity. A detailed pseudocode of the algorithm is presented below.

Step A1: Define the size of the class (N), the number of groups (G) and the parameter ε to be used in ε-dominance method. Determine the number of students in a single group $S = N/G$.

Step A2: Generate N population members uniformly within the bounds of the decision variables and evaluate the value of the various objective functions.

Step A3: Determine the non-dominated solutions based on ε-dominance from the initial population and store them in an archive.

Step A4: Perform Step A5 to Step A30 till the required number of generations is completed.

Step A5: Randomly assign all students such that there are S students in each of the G group.

Step A6: Perform Step A7 to Step A16 for each of G groups.

Step A7: Randomly select a solution (X_{best}) from the archive as the teacher for this group.

Step A8: Perform Step A9 to Step A15 for each member of the group.

Step A9: Determine the teaching factor using $T_{F,i} = round\left(1 + r_i\right) \quad \forall i = 1,2,..,V$ where r_i is a random number between 0 and 1.

Step A10: Randomly select a partner from the group.

Step A11: Determine the mean of the group $\left(\bar{X}\right)$. Generate a potential student ($Xnew$) using Eq. (10.1) with $Y = \bar{X}$ and $F = T_F$. If the selected partner dominates the current member of the group, $a_g = 1$. If the current member of the group dominates the selected partner, $a_g = -1$. If the current solution and the selected partner are both non-dominating to one another, generate a random number between 0 and 1. If the random number is less than 0.5, $a_g = 1$ else $a_g = -1$.

Step A12: Bound *Xnew* using Eq. (10.2).

Step A13: Determine the objective value of the newly generated solution and set $d = 1$.

Step A14: Compare the newly generated member with the member d of the group. If the newly generated solution dominates the member d of the group, replace the member d of the group with the newly generated solution and go to Step A8, else continue to Step A15.

Step A15: If $d < s$, $d = d + 1$. Go to Step A14. If $d = s$, go to Step A8 if all the members have not completed the teacher phase.

Step A16: Go to Step A6 so that the next group undergoes the teacher phase. If all the groups have completed the teacher phase, then go to Step A17.

Step A17: Perform Step A18 to Step A27 for each of the G groups.

Step A18: Randomly select a solution (X_{best}) from the archive as the teacher for this group.

Step A19: Perform Step A20 to Step A26 for each member of the group.

Step A20: Determine the exploration factor using $E_F = round(1 + r)$ where r is a random number between 0 and 1.

Step A21: Randomly select a partner from the group.

Step A22: If the selected partner dominates the current member of the group, $a_g = 1$. If the current member of the group dominates the selected partner, $a_g = -1$. If the current solution and the selected partner are both non-dominating to one another, generate a random number between 0 and 1. If the random number is less than 0.5, $a_g = 1$ else $a_g = -1$. Generate a potential student (*Xnew*)using Eq. (10.1) with $Y = X_g$ and $F = E_F$.

Step A23: Bound Xnew using Eq. (10.2).

Step A24: Determine the objective value of the newly generated solution and set $d = 1$.

Step A25: Compare the newly generated member with the member d of the group. If the newly generated solution dominates the member d of the group, replace the member d of the group with the newly generated solution and go to Step A19, else continue to Step A26.

Step A26: If $d < s$, $d = d + 1$. Go to Step A25. If $d = s$, go to Step A19 if all the members have not completed the learner phase.

Step A27: Go to Step A17 so that the next group undergoes the learner phase. If all the groups have completed the teacher phase, then go to Step A28.

Step A28: Combine all the members of the group with the members in the archive.

Step A29: Perform ε-dominance on the combined members to determine the updated archive.

Step A30: Go to Step A4 if the termination criterion is not satisfied; else, terminate.

Variant II: Solutions generated using both the students

This algorithm is similar to Variant I, with the only difference being the selection of the appropriate equation to determine the new solution in the teacher and the learner phase. This variant employs both equations to generate two new solutions (per member of the group) without checking the dominance between the current member and its partner in the teacher phase (and learner phase) as opposed to the one equation selected in Variant I. This enables the generation of two solutions per member in the teacher phase as well as the learner phase of every generation. A detailed pseudocode of variant II is presented below.

Step B1: Define the size of the class (N), the number of groups (G) and the parameter ε to be used in ε-dominance method. Determine the number of students in a group $S = N/G$..

Step B2: Generate N population members uniformly within the bounds of the decision variables and evaluate the value of the various objective functions.

Step B3: Determine the non-dominated solutions based on ε-dominance from the initial population and store it in archive.

Step B4: Perform Step B5 to Step B42 till the required number of generations are completed.

Step B5: Randomly assign all the students such that there are S students in each of G groups.

Step B6: Perform Step B7 to Step B22 for each of the G groups.

Step B7: Randomly select a solution (X_{best}) from the archive to act as teacher for this group. Determine the mean of the group.

Step B8: Perform Step B9 to Step B21 for each member in the group.

Step B9: Determine the teaching factor using $T_{F,i} = round\left(1 + r_i\right) \quad \forall i = 1,2,..,V$ where r_i is a random number between 0 and 1

Step B10: Randomly select a partner from the group.

Step B11: Generate a potential student (*Xnew*) using Eq. (10.1) where $Y = \bar{X}$ and $F = T_F$.

Step B12: Bound *Xnew* using Eq. (10.2)

Step B13: Determine the objective function values of newly generated solution and set $d = 1$.

Step B14: Compare the newly generated member with the member d of the group. If the newly generated solution dominates the member d of the group, replace the member d of the group with the newly generated solution and go to Step B16, else continue to Step B15.

Step B15: If $d < s$, set $d = d + 1$. Go to Step B14.

Step B16: Generate another potential student (*Xnew*) using Eq. (10.1) where $Y = \bar{X}$ and $F = T_F$

Step B17: Bound *Xnew* using Eq. (10.2)

Step B18: Determine the objective function values of newly generated solution and set $d = 1$.

Step B19: Compare the newly generated member with the member d of the group. If the newly generated solution dominates the member d of the group, replace the member d of the group with the newly generated solution and go to Step B21, else continue to Step B20.

Step B20: If $d < s$, set $d = d + 1$. Go to Step B19. If $d = s$, go to Step B21.

Step B21: Go to Step B8 if all the members have not completed the teachers phase.

Step B22: Go to Step B6 so that the next group undergoes the teacher phase. If all the groups have completed the teacher phase, then go to Step B23.

Step B23: Perform Step B24 to Step B39 for each of the G groups.

Step B24: Randomly select a solution (X_{best}) from the archive to act as teacher for this group.

Step B25: Perform Step B26 to Step B38 for each member in the group.

Step B26: Determine the exploration factor using $E_F = round(1+r)$ where r is a random number between 0 and 1.

Step B27: Randomly select a partner from the group.

Step B28: Generate a potential student (*Xnew*) using Eq. (10.1) with $Y = X_g$ and $F = E_F$

Step B29: Bound *Xnew* using Eq. (10.2)

Step B30: Determine the objective function values of newly generated solution and set $d = 1$.

Step B31: Compare the newly generated member with the member d of the group. If the newly generated solution dominates the member d of the group, replace the member d of the group with the newly generated solution and go to Step B33, else continue to Step B32.

Step B32: If $d < s$, set $d = d + 1$. Go to Step B31.

Step B33: Generate another potential student (*Xnew*) using Eq. (10.1) with $Y = X_g$ and $F = E_F$

Step B34: Bound *Xnew* using Eq. (10.2)

Step B 35: Determine the objective function values of newly generated solution and set $d = 1$

Step B36: Compare the newly generated member with the member d of the group. If the newly generated solution dominates the member d of the group, replace the member d of the group with the newly generated solution and go to Step B38, else continue to Step B37.

Step B37: If $d < s$, set $d = d + 1$. Go to Step B 36. If $d = s$, go to Step B38.

Step B38: Go to Step B25 if all the members have not completed the learners phase.

Step B39: Go to Step B23 so that the next group undergoes the learners phase. If all the groups have completed the learner phase, then go to Step B40.

Step B40: Combine all the members of the group with the members in the archive.

Step B41: Perform ε-dominance on the combined members to determine the updated archive.

Step B42: Go to Step B4 if the termination criterion is not satisfied, else terminate.

In Variant I, Step A7 to Step A16 constitute the teacher phase, and Step A17 to Step A27 constitute the learner phase, whereas in Variant II, Step B6 to Step B22 constitute the teacher phase, and Step B23 to Step B39 constitute the learner phase. It should be noted that Step B11 to Step B15 is similar to Step B16 to Step B20 in the teacher phase (Step B28 to Step B32 is similar to Step B33 to Step B37 in the learner phase). Variant I requires $2N$ functional evaluations per generation, whereas for Variant II it is $4N$, where N is the class size. Thus, the second variant will perform close to half the number of generations for a constant number of functional evaluations than the first variant. If required, the archive size can be restricted to a pre-specified number. However, an appropriate strategy will have to be selected to eliminate some of the points (which are part of the Pareto points determined using the ε-dominance) if the archive size exceeds the pre-specified number. In both variants, the duplicate solutions are not removed and can be incorporated before Step A28 in Variant I (and Step B40 in Variant II), if required. It should be noted that the Variant I can be computationally expensive as it involves $2NG$ comparisons involving two solutions for non-dominance in the teacher phase (Step A11) as well as the learner phase (Step A22), where G is the number of generations.

10.3. Experimental settings

The ten multi-objective optimization problems used for the CEC2009 competition (Zhang et al., 2008) are solved using both the proposed variants. All the problems are scalable, and as per the CEC2009 competition requirement, the results are presented for 30 variables. The number of groups is set to 4, and the population size is set to 52 so as to provide an equal number of students in each group. The value of ε in the ε-dominance method is set to 0.007, which was also used for some problems in the literature (Deb et al., 2005). Each problem is run for 30 instances with a different random seed. In order to reproduce the results, the 'twister' algorithm in MATLAB 2015b is selected with a seed number of 1 to generate thirty random integers,

which are subsequently used as seed numbers in the 'twister' algorithm to realize the thirty runs. The termination criterion for all the ten problems (in all 30 runs) is 300000 functional evaluations, which is consistent with the CEC2009 competition requirements (Zhang et al., 2008) and also the literature (Patel & Savsani, 2016; Rao & Patel, 2014). The Inverse Generational Distance is determined as required in the CEC2009 competition and uses 100 points (for the bi-objective problems (UF1 – UF7)) and 150 points (for the three objective problems (UF8 – UF10)) from the global Pareto front. These points can be obtained using the CEC2009 test suite for MATLAB. All the simulations were performed on a computer with an i7 3.4GHz processor with 24GB RAM.

10.4. Results and discussion

The results obtained by both variants on solving the CEC 2009 benchmark functions are analyzed using the inverted generational distance (IGD) values. Additionally, the performance of these variants is studied through the convergence analysis of the Pareto fronts provided by them on each function.

10.4.1. Statistical analysis of IGD values

The values of the IGD for both variants are reported in Table 10.1. From the table, it can be inferred that the IGD of the proposed variants for all ten problems is inferior to those reported in the literature (Patel & Savsani, 2016; Rao & Patel, 2014). This confirms issues reported in the earlier part of the article are critical and need to be addressed so as to enable independent researchers to use MO- ITLBO as proposed in (Patel & Savsani, 2014a; Patel & Savsani, 2016; Rao & Patel, 2014). The best IGD for problems UF1 – UF10 were obtained with the random seed 23, 2, 24, 16, 1, 26, 24, 28, 28, 26, respectively, for Variant I, whereas the best IGD for problems UF1 – UF10 were obtained with the random seed 8, 19, 10, 20, 22, 27, 24, 13, 5, 14 respectively for Variant II. It should be noted that the archive size was not restricted to a specified number, as the number of Pareto points was significantly less.

Table 10.1 Comparison of IGD values

Problem ID	Reported IGD				IGD of Variant I				IGD of Variant II			
	Best	**Worst**	**Mean**	**Standard Deviation**	**Best**	**Worst**	**Mean**	**Standard Deviation**	**Best**	**Worst**	**Mean**	**Standard Deviation**
UF1	3.91E−03	4.87E−03	4.21E−03	8.04E−04	9.87E-02	4.22E-01	1.82E-01	8.26E-02	1.03E-01	4.32E-01	1.83E-01	9.3E-02
UF2	4.62E−03	5.93E−03	5.19E−03	1.73E−03	6.45E-02	2.52E-01	1.35E-01	5.14E-02	7.30E-02	2.32E-01	1.48E-01	4.7E-02
UF3	3.49E−02	6.17E−02	4.68E−02	6.48E−03	2.66E-01	5.28E-01	3.69E-01	7.35E-02	2.11E-01	6.03E-01	3.78E-01	8.2E-02
UF4	3.74E−02	4.61E−02	4.38E−02	1.07E−02	6.20E-02	8.79E-02	7.16E-02	7.10E-03	5.84E-02	8.02E-02	7.12E-02	6.6E-03
UF5	4.02E−02	9.92E−02	7.48E−02	8.62E−03	8.90E-01	1.7321	1.2881	2.21E-01	7.13E-01	1.7231	1.1947	2.8E-01
UF6	8.68E−03	3.25E−02	1.14E−02	1.01E−02	1.21E-01	1.1128	5.68E-01	1.81E-01	9.70E-02	9.45E-01	6.21E-01	1.6E-01
UF7	1.11E−02	8.48E−02	4.13E−02	2.38E−02	8.39E-02	4.46E-01	1.76E-01	1.06E-01	6.95E-02	4.09E-01	1.64E-01	7.99E-02
UF8	5.11E−02	5.83E−02	6.13E−02	1.65E−03	4.41E-01	1.0449	6.35E-01	2.09E-01	4.41E-01	8.26E-01	5.15E-01	1.15E-01
UF9	6.84E−02	2.00E−01	1.24E−01	8.97E−02	4.15E-01	6.66E-01	5.61E-01	6.90E-02	4.04E-01	6.65E-01	5.72E-01	7.09E-02
UF10	1.19E−01	1.90E−01	1.47E−01	1.29E−02	1.0449	1.0449	1.0449	0.0000	1.0449	1.0449	1.0449	0.0000

10.4.2. Convergence analysis of Pareto fronts

The convergence of non-dominated points obtained by both the variants with the global Pareto is analyzed based on their best run with respect to IGD value is given in Fig. 10.1 and Fig. 10.2. The following inferences can be made for ten problems based on their IGD values and figures. The pareto points obtained by both variants in all runs are provided in Appendix L.

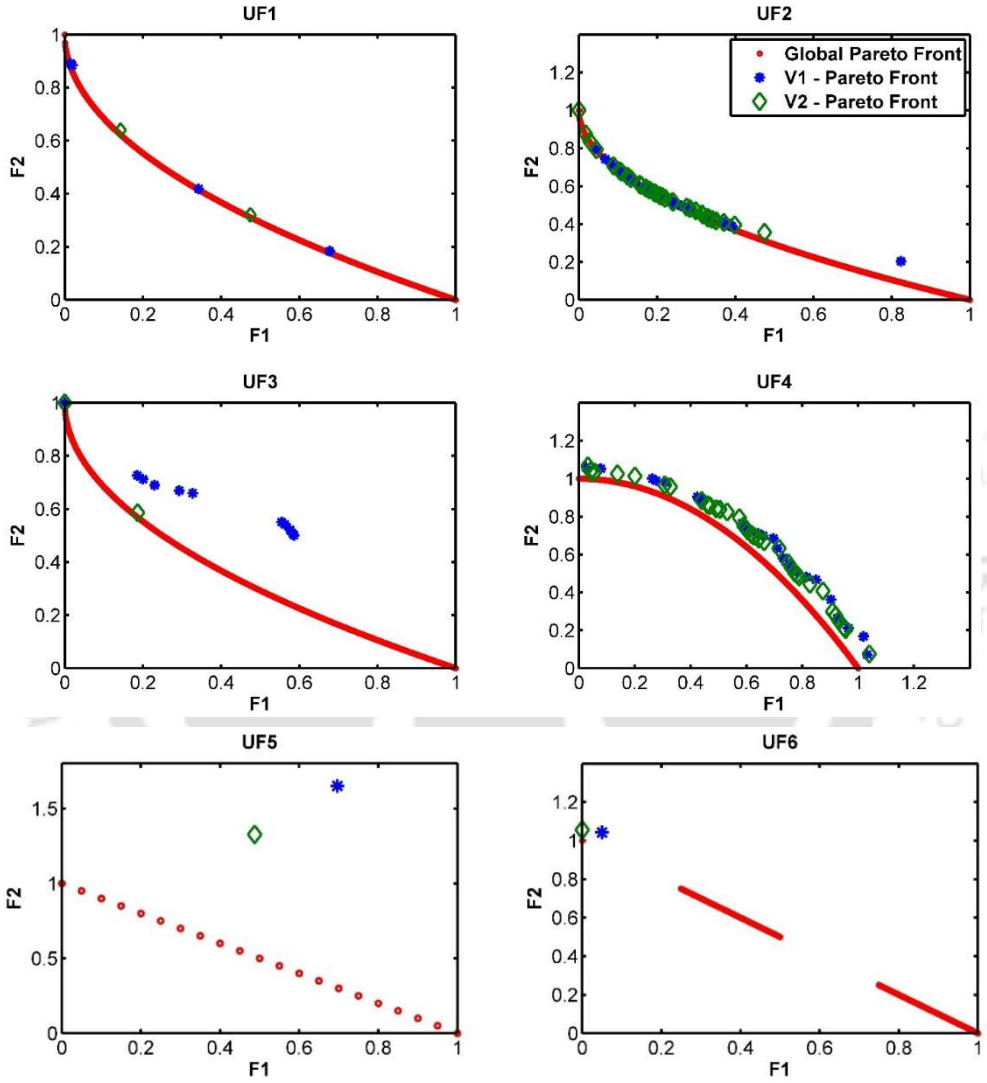


Fig. 10.1 Comparison of Pareto fronts for UF1 – UF6

- It can be observed that Variant I gives comparatively better values for the mean IGD than Variant II for all problems except UF4, UF5, UF7, and UF8.
- In the case of UF10, both variants have obtained similar IGD statistics as they are able to determine only one point in all sixty runs (both variants in all 30 runs).

- The mean and worst IGD values of both variants have a magnitude similar to that reported in the literature (Patel & Savsani, 2016; Rao & Patel, 2014) for the problems UF4 and UF9.
- UF6 has a discrete global Pareto front comprising three regions, but both variants are able to determine at least a single point in each region, as can be observed in Fig. 10.1.
- In all the runs, both variants are able to determine solutions that are closer to the global Pareto front for the problems UFI and UF2.
- The regions of Pareto points determined by both variants are almost similar in UF2 and UF4 for all the runs.
- In their best runs, Variant 2 provides Pareto points, which have a better IGD value than Variant 1 for UF3, UF4, UF5, UF6, UF7 and UF9
- The maximum number of Pareto points obtained in both variants is 34 for UF2, whereas the minimum number of Pareto points obtained in both variants is 1 for UF5 and UF10.

Thus, many of the steps that have not been clearly specified in the literature on MO-ITLBO and detailed in this note are crucial in harnessing the benefits of MO-ITLBO.

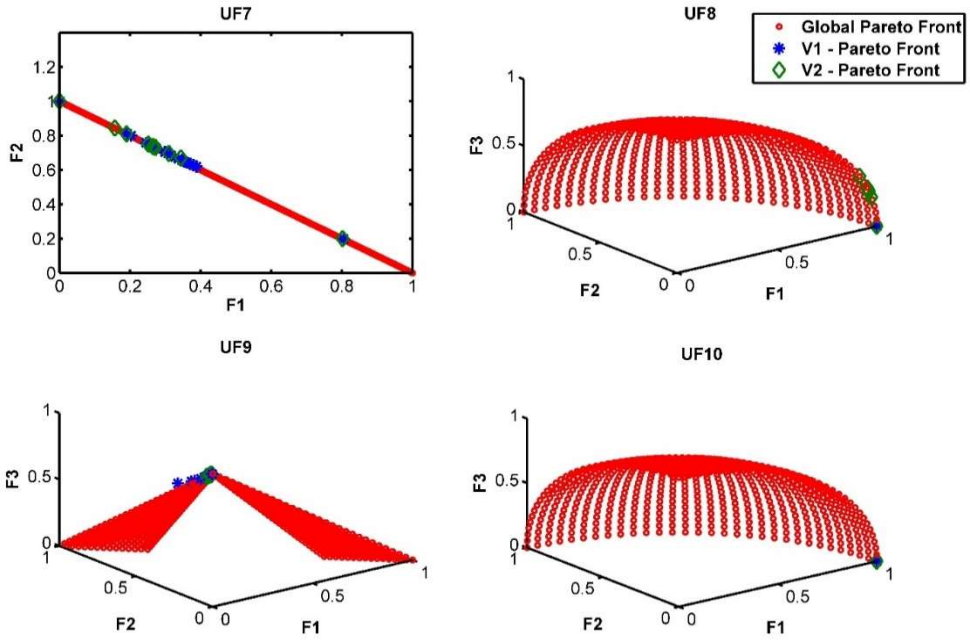


Fig. 10.2 Comparison of Pareto fronts for UF7 – UF10

10.5. Conclusion

MO-ITLBO has been recently proposed, in which an improved version of the basic TLBO has been used to solve MOOP. In these works [21,24], MO-ITLBO was reported to show competitive performance with many other state-of-the-art multi-objective algorithms. However, the extension of ITLBO (primarily developed to solve SOOP) to solve MOOP has not been adequately clarified in any of the three articles that have proposed it (Patel & Savsani, 2014a; Patel & Savsani, 2016; Rao & Patel, 2014). In this chapter, we have raised some such issues in MO-ITLBO, which have become serious impediments in independently applying MO-ITLBO to complex real-life optimization problems. Two variants of MO-ITLBO have been proposed to show some of the possible implementations and their performance confirmed issues reported are critical and should be addressed before using MO-ITLBO,

Chapter 11

Summary of the thesis and future scopes

Metaheuristic techniques, inspired by various optimization procedure that occurs in nature, have obtained wide acceptance from the research community. Metaheuristic techniques in their innate form might not successfully solve problems from multiple fields satisfactorily. Instead, these techniques are modified to improve performance, resulting in their variants and hybrid techniques. This thesis firstly focused on efficient optimization frameworks suitable for metaheuristic techniques in solving combinatorial problems like production planning and scheduling and non-linear models of refrigeration systems. Subsequently, the possibility of harnessing the benefits of parallel computing and the importance of using the exact implementation of a metaheuristic technique to obtain better results is studied. To improve the performance of metaheuristic techniques in solving different problems considered in this work, we have employed optimization frameworks involving better solution representation, modification of the potential solution using the heuristic mechanism, and efficient constraint-handling strategies. The efficacy of the proposed optimization frameworks was analyzed using multiple metaheuristic techniques, including recently proposed and variants of popular techniques.

11.1. Insights from the thesis

This thesis has analyzed three models of production planning involving unique process constraints, namely SLPP, MUIPP, and MUCPP, which have been used to determine an optimal production plan for the petrochemical industry. The strategies employed in the literature have limitations like the lack of compact solution representation and efficient constraint handling techniques. This thesis has proposed a better strategy for optimizing the models suitable for metaheuristic techniques as the respective strategy in the literature.

The proposed strategies adopted a solution structure that fulfills the requirement of unique process constraints, reducing the number of constraints to be handled. Unlike the strategy employed in the literature, which considers decision variables based on processes and the number of units, the proposed approach selects decision variables based on products. Such a choice of decision variables in the proposed strategies is the main factor in reducing the problem dimension. The efficacy of the proposed strategy is analyzed by solving the four case studies by varying resource availability. The reduction in the number of decision variables is approximately 55% in SLPP and MUIPP and a maximum of 98% for the MUCPP cases compared to the strategy in the literature. Similarly, a reduction in the number of constraints to be handled is observed compared to the strategy in the literature. The efficacy of the proposed strategies is analyzed using different metaheuristic techniques. In most instances, the proposed strategies are able to determine an equal or better optimal solution compared to their respective strategies in the literature. The proposed strategies remarkably improved the performance of all the metaheuristic techniques in identifying a better solution than the strategy in the literature, indicating its suitability. This thesis also performed the multi-objective study of the MUCPP for determining the trade-off between the investment budget and the profit. On analyzing the corner solutions, hypervolume ratio, and coverage metric of the obtained Pareto solutions, it is evident that the proposed strategy helped the metaheuristic techniques to obtain a well-converged and diversified Pareto front in all the cases.

The scheduling problems studied in this thesis consider multiple jobs that have to be scheduled on dissimilar parallel machines. In order to reduce the violation of constraints, this work proposed a no-wait time heuristic mechanism that rescheduled the processing of jobs assigned to a machine to the earliest possible time. Such a modification helped to gain more time for the succeeding jobs, which improved the possibilities for obtaining a solution satisfying the due date constraints. Based on the no-wait time heuristic mechanism, job processing begins

immediately after completing its preceding job, which helped satisfy the overlap constraints. Incorporating the heuristic mechanism along with the metaheuristic techniques helped to determine better optima. It is also observed that the heuristic mechanism also improved the convergence rate of all metaheuristic techniques by identifying feasible solutions in their earlier iterations. This study also analyzed the performance of the no-wait time heuristic mechanism in solving the multi-objective scheduling model with the minimization of processing cost and the minimization of makespan as the objectives. A multi-objective variant of the PTLBO, namely NSpTLBO is proposed that utilizes the non-dominated sorting and crowding distance measure for solving the multi-objective scheduling problems. Based on the obtained corner solutions and performance matrices such as hypervolume ratio and coverage metric, NSpTLBO has provided a competitive performance compared to other algorithms. Furthermore, this study also analyzed the performance of multi-objective metaheuristic techniques, including NSpTLBO, in solving scheduling instances with and without using the heuristic mechanism. Comparing the performance of NSpTLBO and the other metaheuristic techniques with and without this heuristic mechanism reveals that incorporating the no-wait time heuristic mechanism leads to the determination of diversified and converged non-dominated solutions.

The strategies employed in this thesis to solve combinatorial problems do not require any modifications to the structure of the metaheuristic techniques. In contrast to other studies in the literature that combine various methods, such as improved solution structures, heuristic mechanisms, and efficient constraint handling techniques with a single metaheuristic technique, this thesis considered multiple metaheuristic techniques and incorporated all the strategies with each of them to analyze their performance comprehensively. The obtained results from all the metaheuristic techniques demonstrated significant benefits compared to their innate forms.

The large computational complexity of metaheuristic techniques adversely affects their applicability in solving problems with computationally intensive objective functions. Parallel computing is observed to be a potential remedy for metaheuristic techniques to overcome this drawback. sTLBO is not inherently parallelizable due to its requirement of sequential evaluation of objective function in the iterative loop. Three parallelized variants of sTLBO are proposed in this thesis to overcome its limitations in solving computationally intensive problems. The performance of these three variants is analyzed on different benchmark functions and compared with sTLBO as well as some recent metaheuristic techniques and found to be providing competitive results. To analyze the efficiency of the three variants of sTLBO, studies based on varying availability of computational resources and increasing complexity of the objective function have been performed. The studies have shown that these proposed variants are competitive while harnessing the benefits of parallel computing as compared to sTLBO.

This thesis also emphasized the importance of providing the exact implementation of newly proposed metaheuristic techniques to ensure their claimed performance as reported in the literature and also for the easiness of their applicability in various fields. In this regard, the multi-objective variant of ITLBO is studied. Several ambiguities were observed in the various steps of MO-ITLBO, due to which multiple ways of implementing them were possible. In view of these possibilities, two variants of MO-ITLBO were proposed, and their performance was analyzed in solving the same set of benchmark functions as in the MO-ITLBO literature. The results provided by the variants did not match with the results provided in the MO-ITLBO literature and were observed to be inferior. Such an observation highlighted that the issues found in various steps were critical and necessitated the importance of providing the exact implementation in the public domain.

Lastly, this thesis proposed an efficient optimization procedure suitable for evolutionary optimization techniques to determine optimum operating variables and efficient absorbent solution-refrigerant-refrigerant pair in a single optimization attempt. A non-linear optimization model of the CACRS system is also solved using multiple metaheuristic techniques. Eleven refrigerants and three absorbent solution-refrigerant pairs were taken as working pairs in the VCRS and VARS, respectively. Unlike other studies in the literature, where multiple optimization attempts were performed to identify the best refrigerant-solution-refrigerant pair along with the operational decisions, the best pair is identified in a single optimization attempt.

11.2. Future scopes

This work primarily explored the applicability of metaheuristic techniques in solving combinatorial problems and non-linear models. In view of this, there are several potential areas for future research, among which a few possibilities are as follows.

Future research can focus on implementing the proposed metaheuristic techniques in real-world scenarios to assess their practical applicability. Collaborations with industry partners or case studies in production planning and scheduling, as well as refrigeration systems, can validate the effectiveness of the proposed methods in solving complex problems and provide insights into their practical challenges and limitations.

The thesis mentions the possibility of harnessing the benefits of parallel computing. Future work can investigate the parallel implementation of metaheuristic algorithms to improve their efficiency and speed up the solving process for large-scale combinatorial problems. This could involve exploring distributed computing frameworks, parallelization strategies, and hardware acceleration techniques to leverage the full potential of parallel computing.

Numerous metaheuristic techniques are introduced to the research community every year. Future research can explore benchmarking newly proposed metaheuristic techniques on the

problem domain presented in this thesis, which can provide valuable insights into their strengths and weaknesses. The results obtained from this study indicate a significant advancement in the field of metaheuristic techniques for combinatorial problems. The practical implications of this research could be far-reaching, enabling better solutions to real-world optimization challenges in various domains, including logistics, scheduling, transportation, and more. Furthermore, this work opens up new avenues for future research in exploring novel strategies and their integration with other metaheuristic algorithms to push the boundaries of optimization capabilities further.

Abdel-Basset, M., Mohamed, R., Abouhawwash, M., Askar, S. S., & Tantawy, A. A. (2023). An Efficient Multilevel Threshold Segmentation Method for Breast Cancer Imaging Based on Metaheuristics Algorithms: Analysis and Validations. *International Journal of Computational Intelligence Systems*, *16*(1), 101. https://doi.org/10.1007/s44196-023-00282-x

Abdel-Basset, M., Mohamed, R., Elkomy, O. M., & Abouhawwash, M. (2022). Recent metaheuristic algorithms with genetic operators for high-dimensional knapsack instances: A comparative study. *Computers & Industrial Engineering*, *166*, 107974. https://doi.org/10.1016/j.cie.2022.107974

Abdollahzadeh, B., Gharehchopogh, F. S., & Mirjalili, S. (2021). African vultures optimization algorithm: A new nature-inspired metaheuristic algorithm for global optimization problems. *Computers & Industrial Engineering*, *158*, 107408. https://doi.org/10.1016/j.cie.2021.107408

Abu-Marrul, V., Martinelli, R., Hamacher, S., & Gribkovskaia, I. (2021). Matheuristics for a parallel machine scheduling problem with non-anticipatory family setup times: Application in the offshore oil and gas industry. *Computers & Operations Research*, *128*, 105162. https://doi.org/10.1016/j.cor.2020.105162

Ahmad, S., Linnhoff, B., & Smith, R. (1990). Cost optimum heat exchanger networks—2. targets and design for detailed capital cost models. *Computers & Chemical Engineering*, *14*(7), 751-767. https://doi.org/10.1016/0098-1354(90)87084-3

Ahmed, R., Mahadzir, S., Erniza Mohammad Rozali, N., Biswas, K., Matovu, F., & Ahmed, K. (2021). Artificial intelligence techniques in refrigeration system modelling and optimization: A multi-disciplinary review. Sustainable Energy Technologies and Assessments, 47, 101488. https://doi.org/10.1016/j.seta.2021.101488

Ahmed, R., Rangaiah, G. P., Mahadzir, S., Mirjalili, S., Hassan, M. H., & Kamel, S. (2023). Memory, evolutionary operator, and local search based improved Grey Wolf Optimizer with linear population size reduction technique. *Knowledge-Based Systems*, *264*, 110297. https://doi.org/10.1016/j.knosys.2023.110297

Al-Betar, M. A., Khader, A. T., & Zaman, M. (2012). University Course Timetabling Using a Hybrid Harmony Search Metaheuristic Algorithm. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, *42*(5), 664-681. https://doi.org/10.1109/TSMCC.2011.2174356

Al-Noaimi, F., Khir, R., & Haouari, M. (2019). Optimal design of a district cooling grid: structure, technology integration, and operation. *Engineering Optimization*, *51*(1), 160-183. https://doi.org/10.1080/0305215X.2018.1446085

Alfares, H., & Al-Amer, A. (2002). An optimization model for guiding the petrochemical industry development in Saudi Arabia. *Engineering Optimization*, *34*(6), 671-687. https://doi.org/10.1080/03052150215722

Alinaghian, M., Tirkolaee, E. B., Dezaki, Z. K., Hejazi, S. R., & Ding, W. (2021). An augmented Tabu search algorithm for the green inventory-routing problem with time windows. *Swarm and Evolutionary Computation*, *60*, 100802. https://doi.org/10.1016/j.swevo.2020.100802

Alrezaamiri, H., Ebrahimnejad, A., & Motameni, H. (2020). Parallel multi-objective artificial bee colony algorithm for software requirement optimization. *Requirements Engineering*, *25*(3), 363-380. https://doi.org/10.1007/s00766-020-00328-y

Aminyavari, M., Najafi, B., Shirazi, A., & Rinaldi, F. (2014). Exergetic, economic and environmental (3E) analyses, and multi-objective optimization of a CO2/NH3 cascade refrigeration system. Applied Thermal Engineering, 65(1), 42-50. https://doi.org/10.1016/j.applthermaleng.2013.12.075

Arora, R., Tulshyan, R., & Deb, K. (2010). Parallelization of binary and real-coded genetic algorithms on GPU using CUDA. *IEEE Congress on Evolutionary Computation*, 1-8. https://doi.org/10.1109/CEC.2010.5586260

Arshad, M. U., Ghani, M. U., Ullah, A., Güngör, A., & Zaman, M. (2019). Thermodynamic analysis and optimization of double effect absorption refrigeration system using genetic algorithm. Energy Conversion and Management, 192, 292-307. https://doi.org/10.1016/j.enconman.2019.03.083

Askarzadeh, A. (2016). A novel metaheuristic method for solving constrained engineering optimization problems: Crow search algorithm. *Computers & Structures*, *169*, 1-12. https://doi.org/10.1016/j.compstruc.2016.03.001

Azad-Farsani, E., Zare, M., Azizipanah-Abarghooee, R., & Askarian-Abyaneh, H. (2014). A new hybrid CPSO-TLBO optimization algorithm for distribution network reconfiguration. *J. Intell. Fuzzy Syst.*, *26*(5), 2175-2184. https://doi.org/10.3233/ifs-130892

Azadeh, A., Goldansaz, S., & Zahedi-Anaraki, A. (2016). Solving and optimizing a bi-objective open shop scheduling problem by a modified genetic algorithm. *The International Journal of Advanced Manufacturing Technology*, *85*(5), 1603-1613. https://doi.org/10.1007/s00170-015-8069-z

Balakrishnan, K., Dhanalakshmi, R., & Seetharaman, G. (2022). S-shaped and V-shaped binary African vulture optimization algorithm for feature selection . *Expert Systems*, *39*(10), e13079. https://doi.org/10.1111/exsy.13079

Barnes, N. (2010). Publish your computer code: it is good enough. . *Nature*, *467(7317)*, 753-753.

Bayraktar, Z., & Komurcu, M. (2016). *Adaptive Wind Driven Optimization* Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS), New York City, United States. https://doi.org/10.4108/eai.3-12-2015.2262424

Bhosale, K. C., & Pawar, P. J. (2020). Production planning and scheduling problem of continuous parallel lines with demand uncertainty and different production capacities. *Journal of Computational Design and Engineering*, *7*(6), 761-774. https://doi.org/10.1093/jcde/qwaa055

Bing Yan, Mikhail A. Bragin, & Luh, P. B. (2020). *An Innovative Formulation Tightening Approach for Job-Shop Scheduling* https://doi.org/10.36227/techrxiv.12783893.v1

Błażewicz, J., Pesch, E., & Sterna, M. (2000). The disjunctive graph machine representation of the job shop scheduling problem. *European Journal of Operational Research*, *127*(2), 317-331. https://doi.org/10.1016/S0377-2217(99)00486-5

Boaventura, K. M., Peixoto, F. C., Fernandes, H. L. S., & Pessoa, F. L. P. (2022). Long-Range Investment Assessment in a Petrochemical Industry: Cost and Safety Considerations. *Computers & Chemical Engineering*, *161*, 107737. https://doi.org/10.1016/j.compchemeng.2022.107737

Brest, J., Maučec, M. S., & Bošković, B. (2020). Differential Evolution Algorithm for Single Objective Bound-Constrained Optimization: Algorithm j2020. 2020 IEEE Congress on Evolutionary Computation (CEC), https://doi.org/10.1109/CEC48606.2020.9185551

Buyukozkan, K., Kucukkoc, I., Satoglu, S. I., & Zhang, D. Z. (2016). Lexicographic bottleneck mixed-model assembly line balancing problem: Artificial bee colony and tabu search approaches with optimised parameters. *Expert Systems with Applications*, *50*, 151-166. https://doi.org/10.1016/j.eswa.2015.12.018

Camp, C. V., & Farshchin, M. (2014). Design of space trusses using modified teaching–learning based optimization. *Engineering Structures*, *62–63*, 87-97. http://www.sciencedirect.com/science/article/pii/S0141029614000236

Chauhan, S. S., & Kotecha, P. (2018). An efficient multi-unit production planning strategy based on continuous variables. *Applied Soft Computing*, *68*, 458-477. https://doi.org/10.1016/j.asoc.2018.03.012

Chauhan, S. S., & Kotecha, P. (2020). Single-Level Production Planning in Petrochemical Industries Using Novel Computational Intelligence Algorithms. In F. Bennis & R. K. Bhattacharjya (Eds.), *Nature-Inspired Methods for Metaheuristics Optimization: Algorithms and Applications in Science and Engineering* (pp. 215-243). Springer International Publishing.https://doi.org/10.1007/978-3-030-26458-1_13

Chauhan, S. S., Sivadurgaprasad, C., Kadambur, R., & Kotecha, P. (2018). A novel strategy for the combinatorial production planning problem using integer variables and performance evaluation of recent optimization algorithms. *Swarm and Evolutionary Computation*, *43*, 225-243. https://doi.org/10.1016/j.swevo.2018.04.004

Chen, D., Zou, F., Wang, J., & Yuan, W. (2016). SAMCCTLBO: a multi-class cooperative teaching–learning-based optimization algorithm with simulated annealing. *Soft Computing*, *20*(5), 1921-1943. https://doi.org/10.1007/s00500-015-1613-9

Chen, X., Mei, C., Xu, B., Yu, K., & Huang, X. (2018). Quadratic interpolation based teaching-learning-based optimization for chemical dynamic system optimization. *Knowledge-Based Systems*, *145*, 250-263. https://doi.org/10.1016/j.knosys.2018.01.021

Cheng, M.-Y., & Prayogo, D. (2014). Symbiotic Organisms Search: A new metaheuristic optimization algorithm. *Computers & Structures*, *139*, 98-112. https://doi.org/10.1016/j.compstruc.2014.03.007

Cheraghalipour, A., Hajiaghaei-Keshteli, M., & Paydar, M. M. (2018). Tree Growth Algorithm (TGA): A novel approach for solving optimization problems. *Engineering Applications of Artificial Intelligence*, *72*, 393-414. https://doi.org/10.1016/j.engappai.2018.04.021

Coello, C. A. C., Pulido, G. T., & Lechuga, M. S. (2004). Handling multiple objectives with particle swarm optimization. *IEEE Transactions on Evolutionary Computation*, *8*(3), 256-279. https://doi.org/10.1109/TEVC.2004.826067

Coello Coello, & A., C. (2002). Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art. *Computer Methods in Applied Mechanics and Engineering*, *191*(11), 1245-1287. https://doi.org/10.1016/S0045-7825(01)00323-1

Costa, A. (2015). Hybrid genetic optimization for solving the batch-scheduling problem in a pharmaceutical industry. *Computers & Industrial Engineering*, *79*, 130-147. https://doi.org/10.1016/j.cie.2014.11.001

Črepinšek, M., Liu, S.-H., & Mernik, L. (2012). A note on teaching–learning-based optimization algorithm. *Information Sciences*, *212*, 79-93. https://doi.org/10.1016/j.ins.2012.05.009

Cruz, N. C., Redondo, J. L., Álvarez, J. D., Berenguel, M., & Ortigosa, P. M. (2017). A parallel Teaching–Learning-Based Optimization procedure for automatic heliostat aiming. The Journal of Supercomputing, 73(1), 591-606. https://doi.org/10.1007/s11227-016-1914-5

D'Ariano, A., Pacciarelli, D., & Pranzo, M. (2007). A branch and bound algorithm for scheduling trains in a railway network. *European Journal of Operational Research*, *183*(2), 643-657. https://doi.org/10.1016/j.ejor.2006.10.034

Daolio, F., Liefooghe, A., Verel, S., Aguirre, H., & Tanaka, K. (2017). Problem Features versus Algorithm Performance on Rugged Multiobjective Combinatorial Fitness Landscapes. *Evolutionary Computation*, *25*(4), 555-585. https://doi.org/10.1162/evco_a_00193

de Miranda, J. L. (2019). The design and scheduling of chemical batch processes: Computational complexity studies. *Computers & Chemical Engineering*, *121*, 367-374. https://doi.org/10.1016/j.compchemeng.2018.11.011

Deb, K. (2001). *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, Inc.

Deb, K., Mohan, M., & Mishra, S. (2005). Evaluating the e-Domination Based Multi-Objective Evolutionary Algorithm for a Quick Computation of Pareto-Optimal Solutions. *Evol. Comput.*, *13*(4), 501-525. https://doi.org/10.1162/106365605774666895

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, *6*(2), 182-197. https://doi.org/10.1109/4235.996017

Deb, K., & Saha, A. (2012). Multimodal Optimization Using a Bi-Objective Evolutionary Algorithm. *Evolutionary Computation*, *20*(1), 27-62. https://doi.org/10.1162/EVCO_a_00042

Dede, T. (2014). Application of Teaching-Learning-Based-Optimization algorithm for the discrete optimization of truss structures *KSCE Journal of Civil Engineering*, *18*(6), 1759-1767. http://dx.doi.org/10.1007/s12205-014-0553-8

Dede, T., & Ayvaz, Y. (2015a). Combined size and shape optimization of structures with a new meta-heuristic algorithm. *Applied Soft Computing*, *28*, 250-258. http://www.sciencedirect.com/science/article/pii/S1568494614006383

Dede, T., & Ayvaz, Y. (2015b). Combined size and shape optimization of structures with a new meta-heuristic algorithm. *Applied Soft Computing*, *28*, 250-258. http://dx.doi.org/10.1016/j.asoc.2014.12.007

Dhiman, G., & Kumar, V. (2017). Spotted hyena optimizer: A novel bio-inspired based metaheuristic technique for engineering applications. *Advances in Engineering Software*, *114*, 48-70. https://doi.org/10.1016/j.advengsoft.2017.05.014

Dokeroglu, T. (2015). Hybrid teaching–learning-based optimization algorithms for the Quadratic Assignment Problem. *Computers & Industrial Engineering*, *85*, 86-101. http://www.sciencedirect.com/science/article/pii/S0360835215001096

Ehteram, M., Karami, H., Mousavi, S.-F., Farzin, S., & Kisi, O. (2017). Optimization of energy management and conversion in the multi-reservoir systems based on evolutionary algorithms. *Journal of Cleaner Production*, *168*, 1132-1142. https://doi.org/10.1016/j.jclepro.2017.09.099

Ewaschuk, C. M., Swartz, C. L. E., & Zhang, Y. (2018). An optimization framework for scheduling of converter aisle operation in a nickel smelting plant. *Computers & Chemical Engineering*, *119*, 195-214. https://doi.org/10.1016/j.compchemeng.2018.08.024

Faramarzi, A., Heidarinejad, M., Mirjalili, S., & Gandomi, A. H. (2020a). Marine Predators Algorithm: A nature-inspired metaheuristic. *Expert Systems with Applications*, *152*, 113377. https://doi.org/10.1016/j.eswa.2020.113377

Faramarzi, A., Heidarinejad, M., Stephens, B., & Mirjalili, S. (2020b). Equilibrium optimizer: A novel optimization algorithm. *Knowledge-Based Systems*, *191*, 105190. https://doi.org/10.1016/j.knosys.2019.105190

Fathi, M., Nourmohammadi, A., H.C. Ng, A., Syberfeldt, A., & Eskandari, H. (2020). An improved genetic algorithm with variable neighborhood search to solve the assembly line balancing problem. *Engineering Computations*, *37*(2), 501-521. https://doi.org/10.1108/EC-02-2019-0053

Fausto, F., Reyna-Orta, A., Cuevas, E., Andrade, Á. G., & Perez-Cisneros, M. (2020). From ants to whales: metaheuristics for all tastes. *Artificial Intelligence Review*, *53*(1), 753-810. https://doi.org/10.1007/s10462-018-09676-2

Fellows, P., & Ouaouich, A. (2004). *Small-scale Fruit and Vegetable Processing and Products: Production methods, equipment and quality assurance practices, User-manual, United Nations Industrial Development Organization, Vienna*.

Feng, Z.-k., Niu, W.-j., Liu, S., Luo, B., Miao, S.-m., & Liu, K. (2020). Multiple hydropower reservoirs operation optimization by adaptive mutation sine cosine algorithm based on neighborhood search and simplex search strategies. *Journal of Hydrology*, *590*, 125223. https://doi.org/10.1016/j.jhydrol.2020.125223

Ferreira, K. M., & de Queiroz, T. A. (2018). Two effective simulated annealing algorithms for the Location-Routing Problem. *Applied Soft Computing*, *70*, 389-422. https://doi.org/10.1016/j.asoc.2018.05.024

García, S., Molina, D., Lozano, M., & Herrera, F. (2008). A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the CEC'2005 Special Session on Real Parameter Optimization. *Journal of Heuristics*, *15*(6), 617. https://doi.org/10.1007/s10732-008-9080-4

Garg, V., Deep, K., & Bansal, S. (2023). Improved Teaching Learning Algorithm with Laplacian operator for solving nonlinear engineering optimization problems. *Engineering Applications of Artificial Intelligence*, *124*, 106549. https://doi.org/10.1016/j.engappai.2023.106549

Gibbons, J. D., & Chakraborti, S. (2011). Nonparametric Statistical Inference. In M. Lovric (Ed.), *International Encyclopedia of Statistical Science* (pp. 977-979). Springer Berlin Heidelberg.https://doi.org/10.1007/978-3-642-04898-2_420

Glover, F., & Laguna, M. (1997). *Tabu Search*. Springer New York, NY. https://doi.org/10.1007/978-1-4615-6089-0

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc.

Gong, M., Jiao, L., Du, H., & Bo, L. (2008). Multiobjective Immune Algorithm with Nondominated Neighbor-Based Selection. *Evolutionary Computation*, *16*(2), 225-255. https://doi.org/10.1162/evco.2008.16.2.225

Grosu, L., Benelmir, R., & Feidt, M. (1999). Technico-economic simulation and optimization of a compression refrigerating machine. *Energy Conversion and Management*, *40*(15), 1651-1660. https://doi.org/10.1016/S0196-8904(99)00058-8

Guzman, E., Andres, B., & Poler, R. (2022). Models and algorithms for production planning, scheduling and sequencing problems: A holistic framework and a systematic review. *Journal of Industrial Information Integration*, *27*, 100287. https://doi.org/10.1016/j.jii.2021.100287

Hajabdollahi, Z., Shafiey Dehaj, M., & Fu, P.-F. (2021). Multi-objective teaching-learning-based optimization of combined commercial fuel cells for electricity production. *Journal of Building Engineering*, *44*, 102643. https://doi.org/10.1016/j.jobe.2021.102643

Harikrishna, N. (2009). Parallel artificial bee colony (PABC) algorithm. 2009 World Congress on Nature & Biologically Inspired Computing (NaBIC), https://doi.org/10.1109/NABIC.2009.5393726

Harjunkoski, I., Jain, V., & Grossman, I. E. (2000). Hybrid mixed-integer/constraint logic programming strategies for solving scheduling and combinatorial optimization problems. *Computers & Chemical Engineering*, *24*(2), 337-343. https://doi.org/10.1016/S0098-1354(00)00470-1

He, L., Cao, Y., Li, W., Cao, J., & Zhong, L. (2022). Optimization of energy-efficient open shop scheduling with an adaptive multi-objective differential evolution algorithm. *Applied Soft Computing*, *118*, 108459. https://doi.org/10.1016/j.asoc.2022.108459

He, P., Hao, J.-K., & Wu, Q. (2021). Grouping memetic search for the colored traveling salesmen problem. *Information Sciences*, *570*, 689-707. https://doi.org/10.1016/j.ins.2021.04.090

Heidari, A. A., Mirjalili, S., Faris, H., Aljarah, I., Mafarja, M., & Chen, H. (2019). Harris hawks optimization: Algorithm and applications. *Future Generation Computer Systems*, *97*, 849-872. https://doi.org/10.1016/j.future.2019.02.028

Henn, S., & Schmid, V. (2013). Metaheuristics for order batching and sequencing in manual order picking systems. *Computers & Industrial Engineering*, *66*(2), 338-351. https://doi.org/10.1016/j.cie.2013.07.003

Herrán, A., de la Cruz, J. M., & de Andrés, B. (2012). Global Search Metaheuristics for planning transportation of multiple petroleum products in a multi-pipeline system. *Computers & Chemical Engineering*, *37*, 248-261. https://doi.org/10.1016/j.compchemeng.2011.10.003

Holland, J. H. (1992). *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. The MIT Press.

Homaee, O., Najafi, A., Dehghanian, M., Attar, M., & Falaghi, H. (2019). A practical approach for distribution network load balancing by optimal re-phasing of single phase customers using discrete genetic algorithm *International Transactions on Electrical Energy Systems*, *29*(5), e2834. https://doi.org/10.1002/2050-7038.2834

Hu, Z., & He, D. (2022). Operation scheduling optimization of gas–steam–power conversion systems in iron and steel enterprises. *Applied Thermal Engineering*, 118121. https://doi.org/10.1016/j.applthermaleng.2022.118121

Imran Hossain, S., Akhand, M. A. H., Shuvo, M. I. R., Siddique, N., & Adeli, H. (2019). Optimization of University Course Scheduling Problem using Particle Swarm Optimization with Selective Search. *Expert Systems with Applications*, *127*, 9-24. https://doi.org/10.1016/j.eswa.2019.02.026

Injeti, S. K., Thunuguntla, V. K., & Shareef, M. (2015). Optimal allocation of capacitor banks in radial distribution systems for minimization of real power loss and maximization of network savings using bio-inspired optimization algorithms. *International Journal of Electrical Power & Energy Systems*, *69*, 441-455. http://dx.doi.org/10.1016/j.ijepes.2015.01.040

J. J. Liang, B. Y. Q., P. N. Suganthan. (2013). *Problem Definitions and Evaluation Criteria for the CEC 2014 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization*

Jain, V., & Grossmann, I. E. (2001). Algorithms for Hybrid MILP/CP Models for a Class of Optimization Problems. *Informs Journal on Computing*, *13*(4), 258-276. https://doi.org/10.1287/ijoc.13.4.258.9733

Jain, V., Sachdeva, G., & Kachhwaha, S. S. (2015a). Energy, exergy, economic and environmental (4E) analyses based comparative performance study and optimization

of vapor compression-absorption integrated refrigeration system. *Energy*, *91*, 816-832. https://doi.org/10.1016/j.energy.2015.08.041

Jain, V., Sachdeva, G., & Kachhwaha, S. S. (2015b). NLP model based thermoeconomic optimization of vapor compression–absorption cascaded refrigeration system. *Energy Conversion and Management*, *93*, 49-62. https://doi.org/10.1016/j.enconman.2014.12.095

Jen-Ya, W. (2020). Minimizing the total weighted tardiness of overlapping jobs on parallel machines with a learning effect. *Journal of the Operational Research Society*, *71*(6), 910-927. https://doi.org/10.1080/01605682.2019.1590511

Ji, X., Ye, H., Zhou, J., Yin, Y., & Shen, X. (2017). An improved teaching-learning-based optimization algorithm and its application to a combinatorial optimization problem in foundry industry. *Applied Soft Computing*, *57*, 504-516. https://doi.org/10.1016/j.asoc.2017.04.029

Kadambur, R., & Kotecha, P. (2015). Multi-level production planning in a petrochemical industry using elitist Teaching–Learning-Based-Optimization. *Expert Systems with Applications*, *42*(1), 628-641. https://doi.org/10.1016/j.eswa.2014.08.006

Kadambur, R., & Kotecha, P. (2016). Optimal production planning in a petrochemical industry using multiple levels. *Computers & Industrial Engineering*, *100*, 133-143. https://doi.org/10.1016/j.cie.2016.08.008

Karaboga, D., & Basturk, B. (2007). A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. *Journal of Global Optimization*, *39*(3), 459-471. https://doi.org/10.1007/s10898-007-9149-x

Karatas, M. (2018). Optimal deployment of heterogeneous sensor networks for a hybrid point and barrier coverage application. *Computer Networks*, *132*, 129-144. https://doi.org/10.1016/j.comnet.2018.01.001

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. Proceedings of ICNN'95 - International Conference on Neural Networks, https://doi.org/10.1109/ICNN.1995.488968

Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by Simulated Annealing. *Science*, *220*, 671-680. https://doi.org/10.1126/science.220.4598.671

Kumar Roy, P., Sur, A., & Pradhan, D. K. (2013). Optimal short-term hydro-thermal scheduling using quasi-oppositional teaching learning based optimization. *Engineering Applications of Artificial Intelligence*, *26*(10), 2516-2524. http://dx.doi.org/10.1016/j.engappai.2013.08.002

Kwon, H., Ngan Do, T., & Kim, J. (2022). Optimization-based integrated decision model for smart resource management in the petrochemical industry. *Journal of Industrial and Engineering Chemistry*, *113*, 232-246. https://doi.org/10.1016/j.jiec.2022.05.051

Lansing, F. (1976). Computer modeling of a single-stage lithium bromide/water absorption refrigeration unit. *Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, Deep Space Network Progress Report*, *42*, 247-257. https://ipnpr.jpl.nasa.gov/progress_report/42-32/32AA.PDF

Latifi, S. E., Mohammadi, E., & Khakzad, N. (2017). Process plant layout optimization with uncertainty and considering risk. *Computers & Chemical Engineering*, *106*, 224-242. https://doi.org/10.1016/j.compchemeng.2017.05.022

Li, J.-q., Pan, Q.-k., & Mao, K. (2015). A discrete teaching-learning-based optimisation algorithm for realistic flowshop rescheduling problems. *Engineering Applications of Artificial Intelligence*, *37*, 279-292. https://doi.org/10.1016/j.engappai.2014.09.015

Li, M., & Wang, G.-G. (2022). A review of green shop scheduling problem. *Information Sciences*, *589*, 478-496. https://doi.org/10.1016/j.ins.2021.12.122

Li, S., Chen, H., Wang, M., Heidari, A. A., & Mirjalili, S. (2020). Slime mould algorithm: A new method for stochastic optimization. *Future Generation Computer Systems*, *111*, 300-323. https://doi.org/10.1016/j.future.2020.03.055

Li, Z., Dey, N., Ashour, A. S., & Tang, Q. (2018). Discrete cuckoo search algorithms for two-sided robotic assembly line balancing problem. *Neural Computing and Applications*, *30*(9), 2685-2696. https://doi.org/10.1007/s00521-017-2855-5

Liang, J., Wang, Y., Zhang, Z.-H., & Sun, Y. (2019). Energy efficient production planning and scheduling problem with processing technology selection. *Computers & Industrial Engineering*, *132*, 260-270. https://doi.org/10.1016/j.cie.2019.04.042

Liang, J. J., Qu, B. Y., Suganthan, P. N., & Hernández-Díaz, A. G. (2013). *Problem Definitions and Evaluation Criteria for the CEC 2013 Special Session on Real-Parameter Optimization*

Lunardi Willian T., Birgin Ernesto G., Laborie Philippe, Ronconi Débora P., & Voos Holger. (2020). Mixed Integer linear programming and constraint programming models for the online printing shop scheduling problem. *Computers & Operations Research*, *123*, 105020. https://doi.org/10.1016/j.cor.2020.105020

Ma, B. J., Pereira, J. L. J., Oliva, D., Liu, S., & Kuo, Y.-H. (2023). Manta ray foraging optimizer-based image segmentation with a two-strategy enhancement. *Knowledge-Based Systems*, *262*, 110247. https://doi.org/10.1016/j.knosys.2022.110247

Ma, Y., Zhang, X., Song, J., & Chen, L. (2021). A modified teaching–learning-based optimization algorithm for solving optimization problem. *Knowledge-Based Systems*, *212*, 106599. https://doi.org/10.1016/j.knosys.2020.106599

Maharana, D., Bhattacharya, T., Kotecha, P., & Anandalakshmi, R. (2022a). Exergetic optimization of simple and finned solar air collectors for humid subtropical regions. *Environmental Science and Pollution Research*, *29*(37), 56473-56489. https://doi.org/10.1007/s11356-021-18070-5

Maharana, D., Kommadath, R., & Kotecha, P. (2022b). An innovative approach to the supply-chain network optimization of biorefineries using metaheuristic techniques. *Engineering Optimization*, 1-18. https://doi.org/10.1080/0305215X.2022.2080204

Maharana, D., & Kotecha, P. (2016). Simultaneous Heat Transfer Search for single objective real-parameter numerical optimization problem. 2016 IEEE Region 10 Conference (TENCON), https://doi.org/10.1109/TENCON.2016.7848404

Makhadmeh, S. N., Khader, A. T., Al-Betar, M. A., Naim, S., Abasi, A. K., & Alyasseri, Z. A. A. (2021). A novel hybrid grey wolf optimizer with min-conflict algorithm for power scheduling problem in a smart home. *Swarm and Evolutionary Computation*, *60*, 100793. https://doi.org/10.1016/j.swevo.2020.100793

Martínez, H., Mauttone, A., & Urquhart, M. E. (2014). Frequency optimization in public transportation systems: Formulation and metaheuristic approach. *European Journal of Operational Research*, *236*(1), 27-36. https://doi.org/10.1016/j.ejor.2013.11.007

Medina, M. A., Das, S., Coello Coello, C. A., & Ramírez, J. M. (2014). Decomposition-based modern metaheuristic algorithms for multi-objective optimal power flow – A comparative study. *Engineering Applications of Artificial Intelligence*, *32*, 10-20. http://dx.doi.org/10.1016/j.engappai.2014.01.016

Meng, X., Liu, Y., Gao, X., & Zhang, H. (2014). A New Bio-inspired Algorithm: Chicken Swarm Optimization. Advances in Swarm Intelligence, Cham. https://doi.org/10.1007/978-3-319-11857-4_10

Merchan, A. F., Lee, H., & Maravelias, C. T. (2016). Discrete-time mixed-integer programming models and solution methods for production scheduling in multistage

facilities. *Computers & Chemical Engineering*, *94*, 387-410. https://doi.org/10.1016/j.compchemeng.2016.04.034

Mernik, M., Liu, S.-H., Karaboga, D., & Črepinšek, M. (2015). On clarifying misconceptions when comparing variants of the Artificial Bee Colony Algorithm by offering a new implementation. *Information Sciences*, *291*, 115-127. http://dx.doi.org/10.1016/j.ins.2014.08.040

Mirjalili, S. (2015). Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. *Knowledge-Based Systems*, *89*, 228-249. https://doi.org/10.1016/j.knosys.2015.07.006

Mirjalili, S. (2016). SCA: A Sine Cosine Algorithm for solving optimization problems. *Knowledge-Based Systems*, *96*, 120-133. https://doi.org/10.1016/j.knosys.2015.12.022

Mirjalili, S., Gandomi, A. H., Mirjalili, S. Z., Saremi, S., Faris, H., & Mirjalili, S. M. (2017). Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems. *Advances in Engineering Software*, *114*, 163-191. https://doi.org/10.1016/j.advengsoft.2017.07.002

Mirjalili, S., Mirjalili, S. M., & Hatamlou, A. (2016). Multi-Verse Optimizer: a nature-inspired algorithm for global optimization. *Neural Computing and Applications*, *27*(2), 495-513. https://doi.org/10.1007/s00521-015-1870-7

Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014a). Grey Wolf Optimizer. *Advances in Engineering Software*, *69*, 46-61. https://doi.org/10.1016/j.advengsoft.2013.12.007

Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014b). Grey Wolf Optimizer. *Advances in Engineering Software*, *69*, 46-61. https://doi.org/10.1016/j.advengsoft.2013.12.007

Mishra, A., & Shrivastava, D. (2018). A TLBO and a Jaya heuristics for permutation flow shop scheduling to minimize the sum of inventory holding and batch delay costs. *Computers & Industrial Engineering*, *124*, 509-522. https://doi.org/10.1016/j.cie.2018.07.049

Mohamed, A. W., Hadi, A. A., Mohamed, A. K., & Awad, N. H. (2020). Evaluating the Performance of Adaptive GainingSharing Knowledge Based Algorithm on CEC 2020 Benchmark Problems. 2020 IEEE Congress on Evolutionary Computation (CEC), https://doi.org/10.1109/CEC48606.2020.9185901

Mostafa, R. R., Gaheen, M. A., Abd ElAziz, M., Al-Betar, M. A., & Ewees, A. A. (2023). An improved gorilla troops optimizer for global optimization problems and feature selection. *Knowledge-Based Systems*, *269*, 110462. https://doi.org/10.1016/j.knosys.2023.110462

Mouret, S., Grossmann, I. E., & Pestiaux, P. (2009). Tightening the Linear Relaxation of a Mixed Integer Nonlinear Program Using Constraint Programming. Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-01929-6_16

Mummareddy, P. K., & Satapaty, S. C. (2015). An Hybrid Approach for Data Clustering Using K-Means and Teaching Learning Based Optimization. *Emerging ICT for Bridging the Future -Proceedings of the 49th Annual Convention of the Computer Society of India CSI Volume 2*, 165-171. http://dx.doi.org/10.1007/978-3-319-13731-5_19

Nagraj, S. M., Kommadath, R., Kotecha, P., & Anandalakshmi, R. (2022). Multi-objective optimization of vapor absorption refrigeration system for the minimization of annual operating cost and exergy destruction. *Journal of Building Engineering*, *49*, 103925. https://doi.org/10.1016/j.jobe.2021.103925

Nasir, M., Das, S., Maity, D., Sengupta, S., Halder, U., & Suganthan, P. N. (2012). A dynamic neighborhood learning based particle swarm optimizer for global numerical

optimization. *Information Sciences*, *209*, 16-36. https://doi.org/10.1016/j.ins.2012.04.028

Nearchou, A. C. (2004). A novel metaheuristic approach for the flow shop scheduling problem. *Engineering Applications of Artificial Intelligence*, *17*(3), 289-300. https://doi.org/10.1016/j.engappai.2004.02.008

Niknam, T., Golestaneh, F., & Sadeghi, M. S. (2012). θ -Multiobjective Teaching Learning-Based Optimization for Dynamic Economic Emission Dispatch. *IEEE Systems Journal*, *6*(2), 341-352. https://doi.org/10.1109/JSYST.2012.2183276

Niu, Q., Zhang, H., & Li, K. (2014). An improved TLBO with elite strategy for parameters identification of PEM fuel cell and solar cell models. *International Journal of Hydrogen Energy*, *39*(8), 3837-3854. http://dx.doi.org/10.1016/j.ijhydene.2013.12.110

Niu, W.-j., Feng, Z.-k., Cheng, C.-t., & Wu, X.-y. (2018). A parallel multi-objective particle swarm optimization for cascade hydropower reservoir operation in southwest China. *Applied Soft Computing*, *70*, 562-575. https://doi.org/10.1016/j.asoc.2018.06.011

Pahade, J. K., & Jha, M. (2022). A Hybrid Fuzzy-SCOOT Algorithm to Optimize Possibilistic Mean Semi-absolute Deviation Model for Optimal Portfolio Selection. *International Journal of Fuzzy Systems*, *24*(4), 1958-1973. https://doi.org/10.1007/s40815-022-01251-w

Patel, B., Desai, N. B., & Kachhwaha, S. S. (2017). Optimization of waste heat based organic Rankine cycle powered cascaded vapor compression-absorption refrigeration system. Energy Conversion and Management, 154, 576-590. https://doi.org/10.1016/j.enconman.2017.11.045

Patel, V., & Savsani, V. (2014). Optimization of a plate-fin heat exchanger design through an improved multi-objective teaching-learning based optimization (MO-ITLBO) algorithm. *Chemical Engineering Research and Design*, *92*(11), 2371-2382. http://dx.doi.org/10.1016/j.cherd.2014.02.005

Patel, V., & Savsani, V. J. (2016). A multi-objective improved teaching–learning based optimization algorithm (MO-ITLBO). *Information Sciences*, 357, 182-200 http://dx.doi.org/10.1016/j.ins.2014.05.049

Pezzella, F., Morganti, G., & Ciaschetti, G. (2008). A genetic algorithm for the Flexible Job-shop Scheduling Problem. *Computers & Operations Research*, *35*(10), 3202-3212. https://doi.org/10.1016/j.cor.2007.02.014

Pierezan, J., & Coelho, L. D. S. (2018, 8-13 July 2018). Coyote Optimization Algorithm: A New Metaheuristic for Global Optimization Problems. 2018 IEEE Congress on Evolutionary Computation (CEC), https://doi.org/10.1109/CEC.2018.8477769

Pinel, F., Dorronsoro, B., & Bouvry, P. (2018). The Virtual Savant: Automatic generation of parallel solvers. *Information Sciences*, *432*, 411-430. https://doi.org/10.1016/j.ins.2017.12.021

Prajapati, D., Chan, F. T. S., Daultani, Y., & Pratap, S. (2022). Sustainable vehicle routing of agro-food grains in the e-commerce industry. *International Journal of Production Research*, *60*(24), 7319-7344. https://doi.org/10.1080/00207543.2022.2034192

Premkumar, M., Jangir, P., Sowmya, R., Alhelou, H. H., Heidari, A. A., & Chen, H. (2021). MOSMA: Multi-Objective Slime Mould Algorithm Based on Elitist Non-Dominated Sorting. *IEEE Access*, *9*, 3229-3248. https://doi.org/10.1109/ACCESS.2020.3047936

Punnathanam, V., Kommadath, R., & Kotecha, P. (2016). Extension and performance evaluation of recent optimization techniques on mixed integer optimization problems. 2016 IEEE Congress on Evolutionary Computation (CEC), https://doi.org/10.1109/CEC.2016.7744348

Punnathanam, V., & Kotecha, P. (2016). Yin-Yang-pair Optimization: A novel lightweight optimization algorithm. *Engineering Applications of Artificial Intelligence*, *54*, 62-79. https://doi.org/10.1016/j.engappai.2016.04.004

Punnathanam, V., & Kotecha, P. (2017). Multi-objective optimization of Stirling engine systems using Front-based Yin-Yang-Pair Optimization. *Energy Conversion and Management*, *133*, 332-348. https://doi.org/10.1016/j.enconman.2016.10.035

R. Kotecha, P., Bhushan, M., & D. Gudi, R. (2009). Efficient Optimization Strategies with Constraint Programming. *56*, 387-404.

Rao, J., S. V., & P., V. D. (2012). Teaching–Learning-Based Optimization: An optimization method for continuous non-linear large scale problems. *Information Sciences*, *183*(1), 1-15. http://dx.doi.org/10.1016/j.ins.2011.08.006

Rao, R., & Patel, V. (2014). A multi-objective improved teaching-learning based optimization algorithm for unconstrained and constrained optimization problems. *International Journal of Industrial Engineering Computations* *5*(1), 1-22. https://doi.org/10.5267/j.ijiec.2013.09.007

Rao, R. V. (2016). Jaya: A simple and new optimization algorithm for solving constrained and unconstrained optimization problems. *International Journal of Industrial Engineering Computations*, *7*(1), 19-34. https://doi.org/10.5267/j.ijiec.2015.8.004

Rao, R. V. (2016.). Review of applications of TLBO algorithm and a tutorial for beginners to solve the unconstrained and constrained optimization problems. *Decision Science Letters* *5*(1), 1-30. https://doi.org/10.5267/j.dsl.2015.9.003

Rao, R. V., J., S. V., & P., V. D. (2012a). Teaching–Learning-Based Optimization: An optimization method for continuous non-linear large scale problems. *Information Sciences*, *183*(1), 1-15. http://dx.doi.org/10.1016/j.ins.2011.08.006

Rao, R. V., Kalyankar, V. D., & Waghmare, G. (2014). Parameters optimization of selected casting processes using teaching–learning-based optimization algorithm. *Applied Mathematical Modelling*, *38*(23), 5592-5608. http://dx.doi.org/10.1016/j.apm.2014.04.036

Rao, R. V., & Patel, V. (2012a). Comparative performance of an elitist teaching-learning-based optimization algorithm for solving unconstrained optimization problems. *International Journal of Industrial Engineering Computations*, *4*(1), 29-50. https://doi.org/10.5267/j.ijiec.2012.09.001

Rao, R. V., & Patel, V. (2012b). An elitist teaching-learning-based optimization algorithm for solving complex constrained optimization problems. *International Journal of Industrial Engineering Computations* *31*(4), 535-560. https://doi.org/10.5267/j.ijiec.2012.03.007

Rao, R. V., & Patel, V. (2013). An improved teaching-learning-based optimization algorithm for solving unconstrained optimization problems. *Scientia Iranica*, *20*(3), 710-720. https://doi.org/10.1016/j.scient.2012.12.005

Rao, R. V., & Patel, V. (2013). Multi-objective optimization of heat exchangers using a modified teaching-learning-based optimization algorithm. *Applied Mathematical Modelling*, *37*(3), 1147-1162. http://dx.doi.org/10.1016/j.apm.2012.03.043

Rao, R. V., Savsani, V. J., & Balic, J. (2012). Teaching–learning-based optimization algorithm for unconstrained and constrained real-parameter optimization problems. *Engineering Optimization*, *44*(12), 1447-1462. https://doi.org/10.1080/0305215X.2011.652103

Rao, R. V., Savsani, V. J., & Vakharia, D. P. (2011). Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems. *Computer-Aided Design*, *43*(3), 303-315. http://dx.doi.org/10.1016/j.cad.2010.12.015

Rao, R. V., Savsani, V. J., & Vakharia, D. P. (2012c). Teaching–Learning-Based Optimization: An optimization method for continuous non-linear large scale problems. *Information Sciences*, *183*(1), 1-15. http://dx.doi.org/10.1016/j.ins.2011.08.006

Rao, R. V., & Waghmare, G. G. (2015). Multi-objective design optimization of a plate-fin heat sink using a teaching-learning-based optimization algorithm. *Applied Thermal Engineering*, *76*, 521-529. https://doi.org/10.1016/j.applthermaleng.2014.11.052

Rao, V. R., & Patel, V. (2013). Multi-objective optimization of two stage thermoelectric cooler using a modified teaching–learning-based optimization algorithm. *Engineering Applications of Artificial Intelligence*, *26*(1), 430-445. http://dx.doi.org/10.1016/j.engappai.2012.02.016

Rao, V. R., & Waghmare, G. G. (2014). A comparative study of a teaching–learning-based optimization algorithm on multi-objective unconstrained and constrained functions. *Journal of King Saud University - Computer and Information Sciences*, *26*(3), 332-346. https://doi.org/10.1016/j.jksuci.2013.12.004

Reynoso-Meza, G., Sanchis, J., Blasco, X., Martínez, M. (2010). Design of Continuous Controllers Using a Multiobjective Differential Evolution Algorithm with Spherical Pruning. Applications of Evolutionary Computation, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-12239-2_55

Roshanaei, V., Azab, A., & ElMaraghy, H. (2013). Mathematical modelling and a meta-heuristic for flexible job shop scheduling. *International Journal of Production Research*, *51*(20), 6247-6274. https://doi.org/10.1080/00207543.2013.827806

Roshanaei, V., Naderi, B., Jolai, F., & Khalili, M. (2009). A variable neighborhood search for job shop scheduling with set-up times to minimize makespan. *Future Generation Computer Systems*, *25*(6), 654-661. https://doi.org/10.1016/j.future.2009.01.004

Rossi, F. L., Nagano, M. S., & Neto, R. F. T. (2016). Evaluation of high performance constructive heuristics for the flow shop with makespan minimization. *The International Journal of Advanced Manufacturing Technology*, *87*(1), 125-136. https://doi.org/10.1007/s00170-016-8484-9

Rothlauf, F. (2011). Optimization Methods. In *Design of Modern Heuristics: Principles and Application* (pp. 45-102). Springer Berlin Heidelberg.https://doi.org/10.1007/978-3-540-72962-4_3

Roy, P. K., & Bhui, S. (2013). Multi-objective quasi-oppositional teaching learning based optimization for economic emission load dispatch problem. *International Journal of Electrical Power & Energy Systems*, *53*, 937-948. http://www.sciencedirect.com/science/article/pii/S0142061513002755

Roy, P. K., Pradhan, M., & Paul, T. (2018). Krill herd algorithm applied to short-term hydrothermal scheduling problem. *Ain Shams Engineering Journal*, *9*(1), 31-43. https://doi.org/10.1016/j.asej.2015.09.003

Sadollah, A., Eskandar, H., & Kim, J. H. (2015). Water cycle algorithm for solving constrained multi-objective optimization problems. *Applied Soft Computing*, *27*, 279-298. https://doi.org/10.1016/j.asoc.2014.10.042

Sáenz-Alanís, C. A., V.D, J., Salazar-Aguilar, M. A., & Boyer, V. (2016). A parallel machine batch scheduling problem in a brewing company. *The International Journal of Advanced Manufacturing Technology*, *87*(1), 65-75. https://doi.org/10.1007/s00170-016-8477-8

Sallam, K. M., Chakrabortty, R. K., & Ryan, M. J. (2020). A two-stage multi-operator differential evolution algorithm for solving Resource Constrained Project Scheduling

problems. *Future Generation Computer Systems*, *108*, 432-444. https://doi.org/10.1016/j.future.2020.02.074

Sallam, K. M., Elsayed, S. M., Chakrabortty, R. K., & Ryan, M. J. (2020). Improved Multi-operator Differential Evolution Algorithm for Solving Unconstrained Problems. 2020 IEEE Congress on Evolutionary Computation (CEC), https://doi.org/10.1109/CEC48606.2020.9185577

Sarkhel, R., Das, N., Saha, A. K., & Nasipuri, M. (2018). An improved Harmony Search Algorithm embedded with a novel piecewise opposition based learning algorithm. *Engineering Applications of Artificial Intelligence*, *67*, 317-330. https://doi.org/10.1016/j.engappai.2017.09.020

Shen, Y., Zhang, C., Soleimanian Gharehchopogh, F., & Mirjalili, S. (2023). An improved whale optimization algorithm based on multi-population evolution for global optimization and engineering design problems. *Expert Systems with Applications*, *215*, 119269. https://doi.org/10.1016/j.eswa.2022.119269

Sherali, H. D., & Driscoll, P. J. (2002). On Tightening the Relaxations of Miller-Tucker-Zemlin Formulations for Asymmetric Traveling Salesman Problems. *Operations Research*, *50*(4), 656–669. https://doi.org/10.1287/opre.50.4.656.2865

Shih-Wei, L., Chen-Yang, C., Pourya, P., & Kuo-Ching, Y. (2021). Multi-temperature simulated annealing for optimizing mixed-blocking permutation flowshop scheduling problems. *Expert Systems with Applications*, *165*, 113837. https://doi.org/10.1016/j.eswa.2020.113837

Shukla, A. K., Singh, P., & Vardhan, M. (2019). A new hybrid wrapper TLBO and SA with SVM approach for gene expression data. *Information Sciences*, *503*, 238-254. https://doi.org/10.1016/j.ins.2019.06.063

Singh, M. R., & Mahapatra, S. S. (2016). A quantum behaved particle swarm optimization for flexible job shop scheduling. *Computers & Industrial Engineering*, *93*, 36-44. https://doi.org/10.1016/j.cie.2015.12.004

Siwi, R. G., Aljumah, F., Li, J., & Xao, X. (2018). Optimal Strategic Planning of Integrated Petroleum and Petrochemical Supply Chain. In A. Friedl, J. J. Klemeš, S. Radl, P. S. Varbanov, & T. Wallek (Eds.), *Computer Aided Chemical Engineering* (Vol. 43, pp. 1201-1206). Elsevier.https://doi.org/10.1016/B978-0-444-64235-6.50209-6

Storn, R., & Price, K. (1997). Differential Evolution – A Simple and Efficient Heuristic for global Optimization over Continuous Spaces. *Journal of Global Optimization*, *11*(4), 341-359. https://doi.org/10.1023/A:1008202821328

Stuart Kozola, Improving Optimization Performance with Parallel Computing, Technical Articles and Newsletters, MathWorks, Last accessed: 28th July 2020. (2009).

Suganthan, P. N., Hansen, N., Liang, J. J., Deb, K., Chen, Y.-P., Auger, A., & Tiwari, S. (2005). *Problem Definitions and Evaluation Criteria for the CEC 2005 Special Session on Real-Parameter Optimization*

Talbi, E.G. (2009). *Metaheuristics: From Design to Implementation*. Wiley Publishing.

Tan, W., Yuan, X., Huang, G., & Liu, Z. (2021). Low-carbon joint scheduling in flexible open-shop environment with constrained automatic guided vehicle by multi-objective particle swarm optimization. *Applied Soft Computing*, *111*, 107695. https://doi.org/10.1016/j.asoc.2021.107695

Tasgetiren, M. F., Liang, Y.-C., Sevkli, M., & Gencyilmaz, G. (2007). A particle swarm optimization algorithm for makespan and total flowtime minimization in the permutation flowshop sequencing problem. *European Journal of Operational Research*, *177*(3), 1930-1947. https://doi.org/10.1016/j.ejor.2005.12.024

Tozzo, E., Cabral Seixas Costa, A. P., & Lins, I. D. (2020). A hybrid multi-objective genetic algorithm for scheduling heterogeneous workover rigs on onshore oil fields. *Journal of Petroleum Science and Engineering*, *195*, 107935. https://doi.org/10.1016/j.petrol.2020.107935

Tsutsumi, D., Gyulai, D., Kovács, A., Tipary, B., Ueno, Y., Nonaka, Y., & Fujita, K. (2020). Joint optimization of product tolerance design, process plan, and production plan in high-precision multi-product assembly. *Journal of Manufacturing Systems*, *54*, 336-347. https://doi.org/10.1016/j.jmsy.2020.01.004

Tumbalam Gooty, R., Agrawal, R., & Tawarmalani, M. (2019). An MINLP formulation for the optimization of multicomponent distillation configurations. *Computers & Chemical Engineering*, *125*, 13-30. https://doi.org/10.1016/j.compchemeng.2019.02.013

Waghmare, G. (2013). Comments on "A note on teaching–learning-based optimization algorithm". *Information Sciences*, *229*, 159-169. http://dx.doi.org/10.1016/j.ins.2012.11.009

Wang, G. G., Gao, D., & Pedrycz, W. (2022). Solving Multiobjective Fuzzy Job-Shop Scheduling Problem by a Hybrid Adaptive Differential Evolution Algorithm. *IEEE Transactions on Industrial Informatics*, *18*(12), 8519-8528. https://doi.org/10.1109/TII.2022.3165636

Wang, Y.-Z. (2003). Using genetic algorithm methods to solve course scheduling problems. *Expert Systems with Applications*, *25*(1), 39-50. https://doi.org/10.1016/S0957-4174(03)00004-6

Weinand, J. M., Sörensen, K., San Segundo, P., Kleinebrahm, M., & McKenna, R. (2022). Research trends in combinatorial optimization *International Transactions in Operational Research*, *29*(2), 667-705. https://doi.org/10.1111/itor.12996

Wilcoxon, F. (1992). Individual Comparisons by Ranking Methods. In S. Kotz & N. L. Johnson (Eds.), *Breakthroughs in Statistics: Methodology and Distribution* (pp. 196-202). Springer New York.https://doi.org/10.1007/978-1-4612-4380-9_16

Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, *1*(1), 67-82. https://doi.org/10.1109/4235.585893

Wu, G., Mallipeddi, R., Suganthan, P. N., Wang, R., & Chen, H. (2016). Differential evolution with multi-population based ensemble of mutation strategies. *Information Sciences*, *329*, 329-345. https://doi.org/10.1016/j.ins.2015.09.009

Xie, L., & Cao, C. (2020). Multi-modal and multi-route transportation problem for hazardous materials under uncertainty. *Engineering Optimization*, 1-21. https://doi.org/10.1080/0305215X.2020.1850708

Xu, Y., Yang, Z., Li, X., Kang, H., & Yang, X. (2020). Dynamic opposite learning enhanced teaching–learning-based optimization. *Knowledge-Based Systems*, *188*, 104966. https://doi.org/10.1016/j.knosys.2019.104966

Yan, B., Luh, P. B., Zheng, T., Schiro, D. A., Bragin, M. A., Zhao, F., Zhao, J., & Lelic, I. (2020). A Systematic Formulation Tightening Approach for Unit Commitment Problems. *IEEE Transactions on Power Systems*, *35*(1), 782-794. https://doi.org/10.1109/TPWRS.2019.2935003

Yang, X.-S. (2012). Flower Pollination Algorithm for Global Optimization. Unconventional Computation and Natural Computation, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-32894-7_27

Yang, X.-S., & Deb, S. (2013). Multiobjective cuckoo search for design optimization. *Computers & Operations Research*, *40*(6), 1616-1624. https://doi.org/10.1016/j.cor.2011.09.026

Yao, L., Wang, X., Ding, T., Wang, Y., Wu, X., & Liu, J. (2019). Stochastic Day-Ahead Scheduling of Integrated Energy Distribution Network With Identifying Redundant Gas Network Constraints. *IEEE Transactions on Smart Grid*, *10*(4), 4309-4322. https://doi.org/10.1109/TSG.2018.2856825

Yepes-Borrero, J. C., Villa, F., Perea, F., & Caballero-Villalobos, J. P. (2020). GRASP algorithm for the unrelated parallel machine scheduling problem with setup times and additional resources. *Expert Systems with Applications*, *141*, 112959. https://doi.org/10.1016/j.eswa.2019.112959

Zapotecas-Martínez, S., García-Nájera, A., & López-Jaimes, A. (2019). Multi-objective grey wolf optimizer based on decomposition. *Expert Systems with Applications*, *120*, 357-371. https://doi.org/10.1016/j.eswa.2018.12.003

Zervoudakis, K., & Tsafarakis, S. (2020). A mayfly optimization algorithm. *Computers & Industrial Engineering*, *145*, 106559. https://doi.org/10.1016/j.cie.2020.106559

Zhang, Q., Zhou, A., Zhaoy, S., Suganthan, P. N., Liu, W., & Tiwari, S. (2008). Multiobjective optimization test instances for the CEC 2009 *Special Session on Performance Assessment of Multi-Objective Optimization Algorithms, Technical Report,* 1-30 .

Zhao, F., Jiang, T., & Wang, L. (2022). A Reinforcement Learning Driven Cooperative Meta-Heuristic Algorithm for Energy-Efficient Distributed No-Wait Flow-Shop Scheduling with Sequence-Dependent Setup Time. *IEEE Transactions on Industrial Informatics*, 1-12. https://doi.org/10.1109/TII.2022.3218645

Zhao, F., Zhang, L., Cao, J., & Tang, J. (2021). A cooperative water wave optimization algorithm with reinforcement learning for the distributed assembly no-idle flowshop scheduling problem. *Computers & Industrial Engineering*, *153*, 107082. https://doi.org/10.1016/j.cie.2020.107082

Zhao, F., Zhao, L., Wang, L., & Song, H. (2020). An ensemble discrete differential evolution for the distributed blocking flowshop scheduling with minimizing makespan criterion. *Expert Systems with Applications*, *160*, 113678. https://doi.org/10.1016/j.eswa.2020.113678

Zhao, W., Wang, L., & Zhang, Z. (2019). Atom search optimization and its application to solve a hydrogeologic parameter estimation problem. *Knowledge-Based Systems*, *163*, 283-304. https://doi.org/10.1016/j.knosys.2018.08.030

Zhu, Z., & Zhou, X. (2020). Flexible job-shop scheduling problem with job precedence constraints and interval grey processing time. *Computers & Industrial Engineering*, *149*, 106781. https://doi.org/10.1016/j.cie.2020.106781

Zou, F., Wang, L., Hei, X., Chen, D., & Wang, B. (2013). Multi-objective optimization using teaching-learning-based optimization algorithm. *Engineering Applications of Artificial Intelligence*, *26*(4), 1291-1300. http://dx.doi.org/10.1016/j.engappai.2012.11.006

Zou, F., Wang, L., Hei, X., Chen, D., Jiang, Q., & Li, H. (2014a). Bare-Bones Teaching-Learning-Based Optimization. *The Scientific World Journal*, *2014*, 17, Article 136920. http://dx.doi.org/10.1155/2014/136920

Zou, F., Wang, L., Hei, X., Chen, D., & Yang, D. (2014b). Teaching–learning-based optimization with dynamic group strategy for global optimization. *Information Sciences*, *273*, 112-131. https://doi.org/10.1016/j.ins.2014.03.038

Žulj, I., Kramer, S., & Schneider, M. (2018). A hybrid of adaptive large neighborhood search and tabu search for the order-batching problem. *European Journal of Operational Research*, *264*(2), 653-664. https://doi.org/10.1016/j.ejor.2017.06.056

Appendix A: Features of metaheuristic techniques used

Table A1. Features of single objective metaheuristic techniques

Algorithm	Inspiration	Solution & objective value	Generation of solution	Population updating strategy
ABC (2005)	Food searching behavior of honey bees	Food source & Nector amount	In the employee and onlooker bee phase, the new solution is generated using a partner solution. In the scout bee phase, a random solution is generated within the bounds	Greedy selection in employed and onlooker phase and (μ, λ) in scout phase
AVOA (2021)	Foraging and navigation behavior of African vultures	Vulture & fitness	Population is divided into two groups with the best solution in one group and the second best solution in the other group. Each solution performs either exploration or exploitation of search space based on the rate of starvation of vultures. The new solution in exploration phase is generated either using best solution, random position and rate of starvation factor or by randomly moving around the best solution, with respect to a probability. The search space is exploited in four different strategies decided by a factor P2.	(μ, λ)
AWDO (2016)	Equations governing the air parcel trajectory under the natural forces	Air parcel Pressure value	The velocity of an air parcel is updated using its previous velocity, previous position, the best solution, the rank of the current solution, and four user-defined parameters. The new position is updated using the previous position and its new velocity.	(μ, λ)
COA (2018)	Social structure and experience sharing of coyotes	Coyote and Social condition	A new solution is generated using two randomly selected solutions and the median of the same pack	Greedy selection
CSA (2016)	Intelligent behaviour of crows	Position of crow and fitness	Based on the awareness probability, the position of each crow is updated either using a randomly selected position from the memory, flight length and current position or using the bounds of the search space.	(μ, λ)
CSO (2014)	Hierarchal behavior of chickens in their social life	Chickens and Fitness value	Roosters modify their position with the help of Gaussian distribution. All hens update their position with the help of the personal best of the hen, rooster and random solution along with their respective the fitness. Chicks are updated using their mother hen and current position.	(μ, λ)
DNLPSO (2012)	Movement of bird flocks in search of food	Particle & Objective function value	The velocity of a particle is updated using its current velocity, personal best (determined through neighborhood learning strategy), and global best. The particle position is updated using the current position and updated velocity.	(μ, λ)
FPA (2012)	Pollination process of flowers	Pollens & objective value	Pollens are modified either using the global post position or with the help of two random pollens from the population.	Greedy selection

Table A1 cont. Features of single objective metaheuristic techniques

Algorithm	**Inspiration**	**Solution & objective value**	**Generation of solution**	**Population updating strategy**
GA (1989)	Natural genetics and survival of the fittest	Chromosome & fitness	Two new offspring are generated using the selected crossover operator and a single offspring is generated using the selected mutation operator.	(μ+ λ)
GWO (2014)	Leadership hierarchy and hunting behavior of grey wolves	Search agents and Objective function value	The best three solutions in the population are α, β and δ solutions, respectively. Every solution updates its position using the α, β, and δ solutions and their current position.	(μ + λ)
HHO (2019)	Hunting behavior of Harris' Hawks	Location of Harris' hawks & Fitness	Based on the escaping energy of the rabbit, the location of the harris hawks are updated using any of the three ways like with the help of other family member or randomly with the help of mean population or by surprise pounce or by random dives.	(μ, λ)
Jaya (2016)	Reaching the best solution by moving away from worse solutions	Candidate solution & function value	A potential solution is generated with the help of the best and worst solutions in the population.	Greedy selection
MA (2020)	Flight behaviour and mating process of mayflies	Position of mayfly & objective function value	Population is divided equally into male and female flies. The velocity of male flies is determined using its current velocity and position, personal best and global best position. The velocity of female flies is modified with the help a male fly if the objective value of the partner is better than the current solution, otherwise its velocity is modified randomly. The position of male and female flies is modified using their respective updated velocity. A linear crossover operation is also performed with a male and female flies.	(μ, λ)
MFO (2015)	Traverse orientation of moths towards light	Moth and Fitness	Best moths identified till the current iteration are considered as flames. Moths are updated with the help of their corresponding flame.	(μ + λ) for flame population and (μ, λ) for moth population
MPEDE (2016)	Parallel direct search method using parameter vectors	Vector & Cost function value	Population divided into three equal sized indicator population modified using one of the three mutation strategies. The remaining reward population is assigned to the best performing mutation strategy. The trial vectors are generated by performing a uniform crossover between the target and the mutant vectors.	Greedy selection
MVO (2016)	Concept of the white hole, black hole, and wormhole of cosmology	Universe and Fitness	By exchanging decision variables from better solutions to worse solutions.	(μ, λ)

Table A1 cont. Features of single objective metaheuristic techniques

Algorithm	**Inspiration**	**Solution & objective value**	**Generation of solution**	**Population updating strategy**
SCA (2016)	Mathematical model based on sine cosine function	Position & Fitness	With an equal probability, the position of solutions are modified with the help of destination position and either by sine or by the cosine of a random number.	(μ, λ)
SHO (2017)	Social and collaborative behavior of spotted hyenas	Search agents & Objective function value	New potential solution is generated using a group of elite solutions and absolute distance between each elite solution with the current solution.	(μ + λ)
SHTS (2016)	Heat transfer among molecules	Molecule & energy level	Solutions in the conduction phase are modified based on energy transfer between higher energetic to lower energetic molecules. The convection phase is the interaction between the best solution with the population. In the radiation phase, each molecule interacts with the surrounding as well as with other molecules.	Greedy selection
SOS (2014)	Symbiotic interaction between organisms	Organism & Fitness value	In the mutualism phase, two new solutions are generated with two randomly selected solutions, the mean of the selected solutions and the best solution. In the commensalism phase, the new solution is generated with the help of using a randomly selected solution from the population. In the parasitism phase, a randomly selected variable is mutated within its bound.	Greedy selection
SPMGLFTLO	Knowledge transferring process in different groups of a classroom	Student & Objective function value	Population is divided into multiple groups, and each solution in the group undergoes the teacher phase or learner phase with equal probability. In the teacher phase, a potential new solution is generated with the best solution, teaching factor and the population mean. In the learner phase, a potential new solution is created using a randomly selected partner. The worst solution in each group performs Lévy flight.	Greedy selection
SSA (2017)	Navigating and foraging behavior of salps	Salp & Fitness	The leaders (best half of the population) update their position using the food position and the bounds of the decision variables, and the followers (worst half of the population) utilize the neighboring salps to update their position.	(μ, λ)
sTLBO	Knowledge transferring process in a classroom environment	Student & Objective function value	In the teacher phase, a potential new solution is generated with the best solution, teaching factor and the population mean. In the learner phase, a potential new solution is created using a randomly selected partner.	Greedy selection
TGA (2018)	Competition of trees to obtain food and light	Tree & Objective function value	Population is modified in four groups. In best tree group, a local search is performed from the current position of the tree. In completion for light phase, a new solution is generated using current position and two different trees which are least distant from the current tree. In remove and replace group,	(μ, λ)
YYPO (2016)	Harmony of conflicting and complimenting nature of dualities	Points & objective function value	The two solutions generates a number of solutions in a hyper volume around them depending on the dimension of the problem.	(μ + λ)

Table A2. Features of multi-objective metaheuristic techniques

Algorithm	**Inspiration**	**Solution & objective value**	**Generation of solution**	**Population updating strategy**
FYYPO (2017)	Harmony of conflicting and complimenting nature of dualities	Points & objective function value	Initially, the number of solutions randomly generated in the search space is equal to the twice the number of objectives. Among these solutions, a pair of solutions generates a number of solutions in a hyper volume around them depending on the dimension of the problem. In specific interval, an archive stage is performed by considering all the paired points, the solutions generated around them and archive points.	The crowding distance measure and non-dominated sorting method are employed for density measure around each point. The non-dominated solutions in the first front are stored in the archive. If the number of first front solutions are higher than the archive size, then least crowded solutions are considered in the archive.
MOCS (2013)	Obligate parasitic behavior of cuckoos and the Levy flight behavior of birds	Cuckoo eggs & objective value	K new solutions are generated using Lévy flight from a single solution, for a problem with K objectives. Worst solutions are replaced by generating new solutions using levy flight and difference between current solution and another solution.	The population for the next iteration is selected from the combined solutions of the newly generated solutions and the population of the current iteration using the nondominated sorting and crowding distance measures.
MOPSO (2004)	Bird flocking or fish schooling	Position of particles & fitness	The position of a solution is updated using the current position and its updated velocity. The velocity is updated with of personal best solution and global best solution. An external repository is kept to save the non-dominated solutions from which a solution is selected as the global best solution.	The population for next generation is updated using (μ, λ) strategy. If the newly generated solution dominates the current solution, it replaces the personal best of the current solution. If the new solution dominates any of the solutions in the repository, then it replaces the dominating solution. When the new solution is non-dominating all the repository solutions, the new solution is added to the repository. When the archive exceeds its size limit, only the least crowded solutions are retained.
MOSMA (2021)	Oscillation behavior of slime mould in laboratory experiments	Slime mould & fitness	Corresponding each solution, a weight factor is determined using best solution, and parameters named smell index and smell order. New solutions are generated using best fitness, weight factor, and difference between two randomly selected solutions from the population.	The newly generated population is combined with the population of the previous iteration. The non-dominated sorting and crowding distance approach is employed to determine the population for the next iteration.

Table A2 cont. Features of multi-objective metaheuristic techniques

Algorithm	**Inspiration**	**Solution & objective value**	**Generation of solution**	**Population updating strategy**
MOWCA (2015)	Water cycle process that occurs in nature	Streams & objective function value	The population is divided into sea, rivers and streams. The sea and river constitutes the best solutions and the rest of the solutions are treated as streams. Least crowded solution from the first front solutions are identified as sea and rivers. The new solutions are generated using the sea, rivers and random streams.	The newly generated solutions are always checked for dominance with sea and river solutions. If new solution dominates river or se, it replaces that specific solution. An external archive is maintained to save the non-dominated solutions identified so far.
MSSA (2017)	Navigating and foraging behavior of salps	Salp & Fitness	The leaders (best half of the population from the least crowded region) update their position using the food position and the bounds of the decision variables, and the followers utilize the neighboring salps to update their position	An archive is maintained to store the non-dominated solutions identified so far. If the number of non-dominated solutions exceed the archive size the solutions from the most crowded region are discarded. When a new solution is generated it is compared with each solution in the archive. When the new solution dominates any of the solution in the archive then the new solution replaces that solution.
NNIA (2008)	Immune system	Antibodies & fitness	Dominant population stores the non-dominated solutions determined so far. Dominant population and active population are the same, if the size of dominant population is less than or equal to maximum size of active population; otherwise the crowding distance measure is used to select the less dense solutions to the active population. In proportional cloning, a group of identical solutions depending on the self-adaptive parameter are descended from each solution. The self-adaptive parameter depends on the crowding distance value of each solution. The cloned population undergoes recombination with the active population after proportional cloning.	An archive is maintained to store the non-dominated solutions determined so far. If the number of non-dominated solutions in the archive exceeds the archive size, the crowding distance approach is employed to select the least crowded solutions. The non-dominated sorting method and crowding distance approach is performed on the combined archive population and the newly generated solutions to obtain the population for the next iteration.

Table A2 cont. Features of multi-objective metaheuristic techniques

Algorithm	**Inspiration**	**Solution & objective value**	**Generation of solution**	**Population updating strategy**
NSGA II (2002)	Natural genetics and survival of the fittest	Chromosome & fitness	New solutions are generated using SBX crossover and polynomial mutation.	The non-dominated sorting method is employed on the combined parent and the offspring population to categorize them into various non-dominated fronts. The solutions from the first front onwards are selected for the next generation until the total number of selected solutions is greater than or equal to the required population size. If the selected solutions are greater than the population size, then crowding distance method is employed to select the least crowded solutions from the last selected front.
spMODE (2010)	Vectors & fitness	Parallel direct search method using parallel vectors	New solutions are generated using difference vector based mutation (DE/rand/1) and binomial crossover. The vectors required to perform the mutation and crossover operator are selected randomly from the external archive and current population.	Spherical pruning is employed to determine the best approximation of the Pareto front. The non-dominated solution with the least Euclidean distance from the ideal point is preferred in the spherical pruning approach. The search direction for the non-dominated solution is varied using discrete arc increments. The external archive stores the non-dominated solutions, and the diversity among these non-dominated solutions is ensured using the spherical pruning approach. The newly generated solution replaces its parent solution only if it dominates the parent solution; otherwise, it is discarded. The newly generated solution is also analyzed for dominance with the solutions in the archive. If it is accepted to the archive, then the spherical pruning is employed to determine the archive for next iteration.

Table A3. Source link for the implementation of single objective metaheuristic techniques

Algorithm	**Source link**
ABC (2005)	https://in.mathworks.com/matlabcentral/fileexchange/74122-artificial-bee-colony-optimization?s_tid=prof_contriblnk
AVOA (2021)	https://in.mathworks.com/matlabcentral/fileexchange/94820-african-vultures-optimization-algorithm?s_tid=prof_contriblnk
AWDO (2016)	https://in.mathworks.com/matlabcentral/fileexchange/57924-the-adaptive-wind-driven-optimization-algorithm
COA (2018)	https://in.mathworks.com/matlabcentral/fileexchange/68373-coa
CSA (2016)	https://in.mathworks.com/matlabcentral/fileexchange/56127-crow-search-algorithm
CSO (2014)	https://in.mathworks.com/matlabcentral/fileexchange/48204-cso
DNLPSO (2012)	https://github.com/P-N-Suganthan/CODES/blob/master/2012-INS-dnlpso.rar
FPA (2012)	https://in.mathworks.com/matlabcentral/fileexchange/45112-flower-pollination-algorithm
GA (1989)	https://in.mathworks.com/matlabcentral/fileexchange/74132-genetic-algorithm?s_tid=prof_contriblnk
GWO (2014)	https://seyedalimirjalili.com/gwo
HHO (2019)	https://in.mathworks.com/matlabcentral/fileexchange/70577-harris-hawks-optimization-hho-algorithm-and-applications
Jaya (2016)	https://goo.gl/rRwcjS
MA (2020)	https://in.mathworks.com/matlabcentral/fileexchange/76902-mayfly-optimization-algorithm
MFO (2015)	https://seyedalimirjalili.com/mfo
MPEDE (2016)	https://github.com/P-N-Suganthan/CODES/blob/master/2016-INS-MPEDE.rar
MVO (2016)	https://seyedalimirjalili.com/mvo
SCA (2016)	https://seyedalimirjalili.com/sca
SHO (2017)	https://in.mathworks.com/matlabcentral/fileexchange/73887-spotted-hyena-optimizer-sho
SHTS (2016)	Obtained from authors
SOS (2014)	https://in.mathworks.com/matlabcentral/fileexchange/47465-sos-m
SSA (2017)	https://seyedalimirjalili.com/ssa
TGA (2018)	Obtained from authors
YYPO (2016)	https://in.mathworks.com/matlabcentral/fileexchange/65558-yin-yang-pair-optimization-yypo?s_tid=prof_contriblnk

Table A4. Source link for the implementation of multi-objective metaheuristic techniques

Algorithm	**Source link**
FYYPO (2017)	Obtained from authors
MOCS (2013)	https://in.mathworks.com/matlabcentral/fileexchange/74752-multiobjective-cuckoo-search-mocs
MOPSO (2004)	https://in.mathworks.com/matlabcentral/fileexchange/62074-multi-objective-particle-swarm-optimization-mopso
MOSMA (2021)	https://in.mathworks.com/matlabcentral/fileexchange/85003-mosma-multi-objective-slime-mould-algorithm
MOWCA (2015)	https://ali-sadollah.com/water-cycle-algorithm-wca/
MSSA (2017)	https://seyedalimirjalili.com/ssa
NNIA (2008)	https://see.xidian.edu.cn/iiip/mggong/Projects/NNIA.htm
NSGA II (2002)	https://in.mathworks.com/matlabcentral/fileexchange/10429-nsga-ii-a-multi-objective-optimization-algorithm
spMODE (2010)	https://in.mathworks.com/matlabcentral/fileexchange/39215-multi-objective-differential-evolution-algorithm-with-spherical-pruning

Appendix B: Data for the production planning problem

Table B1. Data for propylene products and processes (Alfares & Al-Amer, 2002)

Product name	Product ID	Sale price ($/ton)	Process *ID*	Process name	Propylene used/ton	Capacity (10^3 ton/yr)			Production cost ($10^6/yr)			Investment cost ($10^6)		
						l_j	m_j	h_j	C_{lj}	C_{mj}	C_{hj}	V_{lj}	V_{mj}	V_{hj}
Polypropylene copolymer	i_1	975	j_1	Amoco/Chisso	0.948	70	135	270	50.7	90.1	170.7	55	81.1	131.6
			j_2	BASF	0.9432	75	150	300	56.8	103.8	196.2	58	85.1	132.4
			j_3	Himont	0.949	77.5	155	310	56.9	103.7	195.7	60.2	86.8	134.1
Polypropylene block copolymer	i_2	975	j_4	Sumitomo (gas phase)	0.9546	70	145	290	51.7	97.6	184.8	55.1	83.1	132
			j_5	UCC/Shell	0.955	47.5	95	190	38.2	69.8	130.4	43.3	66.8	104.3
Polypropylene homopolymer	i_3	780	j_6	Borealis	1.045	40	80	160	38.5	65.2	120.7	66.2	92.8	153.2
			j_7	UCC/Shell	1.05	40	80	160	31.8	57.1	105.5	40	61.4	95.1
Phenol	i_4	735	j_8	From C_6H_6/C_3H_8 via cumene	0.5103	45	90	180	37.8	57.7	94.9	106.6	151.7	231.5
Acrylic acid (ester grade)	i_5	1450	j_9	Two-stage oxidation	0.6289	40	80	160	38.5	65.6	119.1	82.8	125.4	207
Propylene oxide	i_6	1130	j_{10}	Arco process (styrene product)	0.8648	90	180	360	92.2	159.2	290.9	233.5	390.7	698.7
			j_{11}	Texaco (T butanol byproduct)	0.9546	90	180	360	86.7	154.1	287.7	185.8	304.5	537.1
			j_{12}	Chlorohydrin	0.8265	90	180	360	95.8	175	330.9	119	179.4	289.2
			j_{13}	Acro process (T butanol byproduct)	0.7875	90	180	360	87.5	157.2	294.9	212.3	362.7	657.7
			j_{14}	Cell liquor neutralization	0.8101	90	180	360	105.9	196.6	375.2	109.8	164.3	263.1
			j_{15}	Shell process (styrene byproduct)	0.8782	90	180	360	93.1	131.1	239.4	221.7	376.1	672.7
N-butanol	i_7	830	j_{16}	Via Cobalt hydrocarbonyl catalyst	0.815	50	100	200	41.4	68.7	117.2	115.5	180.4	287.4
			j_{17}	Via N butryaldehyde Rh catalyst	0.6994	50	100	200	34.9	62	111.6	63.7	100.2	156.3
Cumene	i_8	450	j_{18}	From C_6H_6 and Propylene	0.3784	60	120	240	36.6	62.1	120.8	23.1	33.2	50.7

Table B2. Data for ethylene products and processes (Alfares & Al-Amer, 2002)

Product name	**Product ID**	**Sale price ($/ton)**	**Process *ID***	**Process name**	**Ethylene used/ton**	**Capacity (10^3 ton/yr)**			**Production cost (10^6/yr)**			**Investment cost (10^6)**		
						l_j	m_j	h_j	C_{lj}	C_{mj}	C_{hj}	V_{lj}	V_{mj}	V_{hj}
Poly vinyl Chloride	i_9	740	j_{19}	Suspension polymerization		100	200	400	67.6	125.2	237.2	117.6	186	307.5
			j_{20}	Bulk polymerization		50	100	300	33	63.1	163.8	62.5	114	209.6
Poly vinyl Chloride (dispersion)	i_{10}	1250	j_{21}	Batch emulsion polymerization		25	50	100	28.7	48.3	86	73.1	101.1	148
			j_{22}	Continuous emulsion polymerization		25	50	100	24	43.1	79.5	46.5	70.7	110.1
Vinyl chloride	i_{11}	430	j_{23}	TOSOH technology		125	250	500	63.8	123.5	241	49.2	74.4	112.8
			j_{24}	Pyrolysis		125	250	500	68.5	134.5	264	79.1	144.2	258.1
			j_{25}	Chlorination/ Oxychlorination	0.4678	250	500	1000	101.5	195	377	134	229.9	392.2
Ethylene glycol	i_{12}	600	j_{26}	Hydration of EO all EO for EG	0.7267	90	180	360	50.3	90	165.6	142.6	234.8	397.5
Vinyl acetate	i_{13}	690	j_{27}	From ethylene and acetic acid	0.393	67.5	135	200	53.9	101.2	146.4	82.7	133.6	181.3
High density polyethylene	i_{14}	860	j_{28}	UCC process	1.02	70	135	270	42.1	75.1	141.8	56.9	84.5	131.5
			j_{29}	Du Pont process	1.02	70	135	270	44.6	77.5	147.7	63.4	84.5	136.9
			j_{30}	Philips process	1.02	70	135	270	44.6	78.8	148	66.5	96.2	147.7
Linear low density polyethylene	i_{15}	900	j_{31}	Dry mode gas phase univation process	0.9461	100	200	400	55.7	106.8	208.4	51.4	83	144.5
			j_{32}	Bimodal grade by mixed mettallocene / Ziegler catalyst	0.9387	75	150	300	48.3	90.2	172.8	46.9	66	98.6
			j_{33}	Bimodal grade by unipol process	0.943	122.5	245	490	92	174	336.2	82.4	116.6	175.6
Low density polyethylene	i_{16}	870	j_{34}	High pressure tubular reactor	1.06	50	100	200	34.9	63.9	120.4	72	117.7	199.7

Table B3. Data for synthesis gas products and processes (Alfares & Al-Amer, 2002)

Product name	**Product *ID***	**Sale price ($/ton)**	**Process *ID***	**Process name**	**Methane used/ton**	**Capacity (10^3 ton/yr)**			**Production cost (10^6/yr)**			**Investment cost (10^6)**		
						l_j	m_j	h_j	C_{lj}	C_{mj}	C_{hj}	V_{lj}	V_{mj}	V_{hj}
Acetic acid	i_{17}	480	j_{35}	Low-pressure carbonylation (Rh. Catalyst solution)		182.5	365	540	63.2	111.4	156.6	125.6	195.9	259.6
			j_{36}	Low-pressure carbonylation supported Rh. Catalyst		182.5	365	540	60.3	103	142.6	116.4	168.2	213.5
			j_{37}	Low-pressure carbonylation Rh. Halide catalyst		180	360	550	64.7	110.2	154.6	133.2	196.3	248.9
Ammonia	i_{18}	160	j_{38}	ICI AMV process	6.3500	300	430	590	48.3	65	85	210.9	278.2	356
			j_{39}	MW Kellog process	5.9280	300	430	590	52.8	71.4	92.7	243.5	322.4	412.6
			j_{40}	ICI LCA process	6.6780	105	170	340	19.4	27.4	47.3	87	119.5	196.7
Formaldehyde 1	i_{19}	500	j_{41}	From methanol using silver catalyst	6.6780	15	25	50	6.6	9.7	17.7	15.3	20.2	32.7
			j_{42}	From methanol using FeMo catalyst	6.6780	15	25	50	6.9	10.6	19.4	17.9	26.2	44.9
Methanol	i_{20}	150	j_{43}	Lurgi process	7.8670	415	830	1660	55.2	96.3	184.3	224.6	365.5	682.1
			j_{44}	ICI process copper catalyst	7.7780	415	830	1660	56.5	100.5	194.3	228.5	384.6	727.6
			j_{45}	ICI LCM process	7.6610	415	830	1660	51.9	98	187.6	199.1	371.5	702.9

Table B4. Data for aromatics (BTX) products and processes (Alfares & Al-Amer, 2002)

Product name	**Product *ID***	**Sale price ($/ton)**	**Process *ID***	**Process name**	**Ethylene used/ton**	**Capacity (10^3 ton/yr)**			**Production cost (10^6/yr)**			**Investment cost (10^6)**		
						l_j	m_j	h_j	C_{lj}	C_{mj}	C_{hj}	V_{lj}	V_{mj}	V_{hj}
Styrene	i_{21}	760	j_{46}	Liquid phase alkyl/ adiabatic dehydrogenation	0.2891	225	450	680	105.8	204.8	306	116.9	190	265.1
			j_{47}	Liquid phase alkyl Oxidative reheating	0.2878	225	450	680	108	209.7	313.5	115.6	191.8	266.8
			j_{48}	Vapor phase alkyl/adiabatic dehydrogenation	0.2843	225	450	680	105.6	202.5	302.6	125.2	192.7	269
			j_{49}	Vapor phase alkyl/Isothermal dehydrogenation	0.2874	225	450	680	106.7	206.1	308.1	125.2	202	285.5
Pthalic anhydride	i_{22}	700	j_{50}	Attochem/Nippon		12.5	25	50	9.4	16.4	28.4	26	40.8	63.9
			j_{51}	From O-Xylene by Alsuisse Italia process		12.5	25	50	9	15.4	27.3	27.7	39.6	56.9
Phenol	i_{23}	735	j_{52}	Liquid phase oxidation of toluene		45	90	180	36.8	64	118.7	108.8	157.2	251.6
PTA	i_{24}	680	j_{53}	Hydrolysis of dimethyl terephthalate		125	250	500	81.4	145.8	275.5	208.1	308.6	515.5
			j_{54}	From P-xylene by Bromine promoted air oxidation		125	250	500	78.4	145	277	170.5	267.3	452.7

Appendix C: Sensitivity analysis of parameters in metaheuristic techniques on solving MUCPP problems

The eight metaheuristic techniques used to solve the multi-unit production planning problem (MUPP) include multi-population based ensemble of mutation strategies differential evolution (MPEDE), sanitized teaching learning based optimization (sTLBO), artificial bee colony (ABC), dynamic neighborhood learning particle swarm optimizer (DNLPSO), sine cosine algorithm (SCA), symbiotic organisms search (SOS), single phase multi-group teaching learning based optimization with lévy flight (SPMGLFTLO) and harris hawk optimization (HHO). This Wilcoxon signed rank test and the statistical analysis of the results are utilized to identify the appropriate parameter values for solving the MUPP cases using the proposed strategy. The selected settings for the parameters of each technique are used for solving four case studies of the MUPP problem using the proposed strategy and the strategy in the literature.

The sensitivity of the algorithm-specific parameters of ABC, DNLPSO, MPEDE, SCA, and SPMGLFTLO are analyzed using the settings mentioned in Table C1. As techniques such as HHO, SOS, and sTLBO do not possess any tunable parameters, these are not considered in the study regarding parameter tuning. The parameter values used for their sensitivity analysis of DNLPSO and MPEDE are selected based on choices specified in the literature (Nasir et al., 2012; Wu et al., 2016). The values for parameter N_g in SPMGLFTLO are selected in such a way that each group contains a maximum of half the population and a minimum of two solutions. The population size and the termination criterion for every metaheuristic technique are identical for all the parameter settings. To analyze the impact of tunable parameters on the performance of the metaheuristic techniques in solving the four cases using the proposed strategy, we used the Wilcoxon signed-rank test at 0.05 significance (α) and the statistical measures such as *best*, *worst*, *mean,* and *median* values of the results obtained by using different settings of the algorithm parameters.

Table C1. Settings for the parameter tuning

Technique	Parameter	Values
ABC	Limit (l)	$l = 0.1n_eD$, $0.5n_eD$, and n_eD, where n_e is the number of employed bees and D is the problem dimension
DNLPSO	Refreshing gap (m) and regrouping period (g)	(m, g) = (3, 5) and (7, 10)
MPEDE	Ratio of indicator population to the whole population (λ_1) and generation gap (n_g).	(λ_1, n_g) = (0.2, 10), (0.2, 20), (0.2, 30), (0.2, 50), (0.2, 80), (0.1, 20), (0.15, 20), (0.25, 20), and (0.3, 20),
SCA	Constant (a)	a = 0.5, 1, and 2
SPMGLFTLO	Number of groups (N_g)	N_g = 2, 4, 5, 10, 20, 25, and 50

The Wilcoxon signed-rank test (García et al., 2008; Wilcoxon, 1992) is a commonly used non-parametric statistical test for analyzing the performance of metaheuristic techniques in solving multiple optimization problems. The Wilcoxon test does not consider a difference in the performance of the settings of an algorithm when the obtained p-value is greater than the significance value of 0.05. The R+ value is the sum of ranks given to the positive difference among the objective values obtained using the two settings. Similarly, R− value is the sum of ranks given for the negative differences. As the objective of the MUPP problem is maximization, the parameter setting with a high R+ value indicates its superior performance compared to the other setting, and the higher R− value specifies the opposite scenario. In order to tune the parameters of different techniques, we have solved four cases of the MUPP problem using the proposed strategy. The fitness value of the best solution in 26 runs is considered as the sample for the Wilcoxon signed-rank test.

The p-value of the Wilcoxon signed rank test on ABC with different settings for the parameter limit is provided in Table C2 of the Appendix. The p-value obtained based on the results of ABC with different settings is greater than the significance value of 0.05, implying the results are not statistically different. The statistical analysis of the results obtained using different settings is provided in Table C3. It is observed that ABC with a limit of $0.1n_eD$

provided an overall better performance based on the statistical measures such as *best*, *worst*, *mean*, and *median* values. Hence in Chapter 3, the value for the limit is set to $0.1n_eD$.

Table C2. *p*-value of the Wilcoxon signed-rank test on different settings of ABC

	Case 1				**Case 2**		
	S1	S2	S3		S1	S2	S3
S1	–	8.03E-02	8.03E-02	S1	–	8.03E-02	8.03E-02
S2	8.03E-02	–	1.00E+00	S2	8.03E-02	–	1.00E+00
S3	8.03E-02	1.00E+00	–	S3	8.03E-02	1.00E+00	–
	Case 3				**Case 4**		
	S1	S2	S3		S1	S2	S3
S1	–	8.03E-02	8.03E-02	S1	–	8.03E-02	8.03E-02
S2	8.03E-02	–	1.00E+00	S2	8.03E-02	–	1.00E+00
S3	8.03E-02	1.00E+00	–	S3	8.03E-02	1.00E+00	–

S1: $l = 0.1n_eD$, S2: $l = 0.5n_eD$, S3: $l = n_eD$

Table C3. Statistical analysis of ABC with different settings

	Best			**Worst**		
	S1	S2	S3	S1	S2	S3
Case 1	**512.83**	**512.83**	**512.83**	**138.16**	38.11	38.11
Case 2	**568.24**	500.92	500.92	**189.19**	101.64	101.64
Case 3	**787.57**	**787.57**	**787.57**	**252.56**	178.79	178.79
Case 4	**1004.58**	**1004.58**	**1004.58**	**214.33**	**214.33**	**214.33**
	Mean			**Median**		
	S1	S2	S3	S1	S2	S3
Case 1	**299.55**	273.45	273.45	**279.41**	254.08	254.08
Case 2	**360.11**	303.42	303.42	**360.42**	300.04	300.04
Case 3	**485.48**	434.64	434.64	**520.88**	429.85	429.85
Case 4	**501.47**	496.03	496.03	**467.60**	**467.60**	**467.60**

S1: $l = 0.1n_eD$, S2: $l = 0.5n_eD$, S3: $l = n_eD$

The *p*-values of the Wilcoxon test and the statistical results provided by DNLPSO with different parameter settings are provided in Table C4 and Table C5, respectively. The *p*-value obtained with different settings of DNLPSO is greater than the significance value of 0.05, implying the results are not statistically different. On analyzing the *best* statistical measure, it has been found that the setting with m=3 and g=5 has provided a better value than the other setting in three out of four cases. Based on this observation, we have set m and g to 3 and 5, respectively, in the rest of the analysis.

Table C4. *p*-value of the Wilcoxon signed-rank test on different settings of DNLPSO

Case 1			Case 2			Case 3			Case 4		
	S1	S2		S1	S2		S1	S2		S1	S2
S1	–	0.75	S1	–	0.37	S1	–	0.36	S1	–	0.61
S2	0.75	–	S2	0.37	–	S2	0.36	–	S2	0.61	–
S1: (m, g) = (3, 5), S2: (m, g) = (7, 10)											

Table C5. Statistical analysis of DNLPSO with different settings

	Best		**Worst**	
	S1	S2	S1	S2
Case 1	713.76	**719.29**	**-9.06E+22**	-1.77E+23
Case 2	**792.37**	699.56	**-8.28E+22**	-1.71E+23
Case 3	**1188.68**	1082.86	-5.62E+23	**-5.28E+23**
Case 4	**1413.41**	1409.82	-5.46E+23	**-5.14E+23**
	Mean		**Median**	
	S1	S2	S1	S2
Case 1	**-1.30E+22**	-1.84E+22	137.20	**224.05**
Case 2	**-1.11E+22**	-1.66E+22	180.69	**182.46**
Case 3	-7.36E+22	**-7.26E+22**	**-7.65E+20**	-8.26E+20
Case 4	-8.43E+22	**-7.51E+22**	-1.05E+21	**-7.65E+20**
S1: (m, g) = (3, 5), S2: (m, g) = (7, 10)				

The *p*-value of the Wilcoxon signed-rank test performed on the results obtained with different parameter settings of the MPEDE is provided in Table C6, indicating that the results obtained by any of the settings are not statistically different in solving the four case studies of the MUPP problem. Also, from the statistical analysis of results given in Table C7, it is not possible to conclude that a particular parameter setting is consistently better than others in all the cases. Hence, based on the suggestion given by the inventors of MPEDE, we considered parameter settings for λ_1 and n_g as 0.2 and 20, respectively.

Table C6. *p*-value of the Wilcoxon signed-rank test on different settings of MPEDE

Case 1									
	S1	S2	S3	S4	S5	S6	S7	S8	S9
S1	–	2.9E-01	5.4E-01	4.7E-01	9.0E-01	2.0E-01	4.8E-01	2.1E-01	9.9E-01
S2	2.9E-01	–	4.1E-01	1.3E-01	2.0E-01	2.1E-01	3.9E-01	2.3E-01	8.5E-01
S3	5.4E-01	4.1E-01	–	4.7E-01	6.1E-01	2.5E-01	4.7E-01	2.2E-01	8.7E-01
S4	4.7E-01	1.3E-01	4.7E-01	–	8.6E-01	2.6E-01	4.8E-01	1.9E-01	9.1E-01
S5	9.0E-01	2.0E-01	6.1E-01	8.6E-01	–	2.8E-01	4.5E-01	1.9E-01	9.3E-01
S6	2.0E-01	2.1E-01	2.5E-01	2.6E-01	2.8E-01	–	5.3E-01	**5.4E-03**	2.4E-01
S7	4.8E-01	3.9E-01	4.7E-01	4.8E-01	4.5E-01	5.3E-01	–	7.3E-02	4.5E-01
S8	2.1E-01	2.3E-01	2.2E-01	1.9E-01	1.9E-01	**5.4E-03**	7.3E-02	–	1.7E-01
S9	9.9E-01	8.5E-01	8.7E-01	9.1E-01	9.3E-01	2.4E-01	4.5E-01	1.7E-01	–
Case 2									
	S1	S2	S3	S4	S5	S6	S7	S8	S9
S1	–	2.2E-01	1.0E+00	7.4E-01	7.4E-01	8.1E-01	4.2E-01	1.9E-01	2.9E-01
S2	2.2E-01	–	6.0E-01	3.9E-01	5.0E-01	7.5E-01	5.0E-01	1.8E-01	2.4E-01
S3	1.0E+00	6.0E-01	–	6.6E-01	8.8E-01	7.3E-01	5.2E-01	1.8E-01	2.5E-01
S4	7.4E-01	3.9E-01	6.6E-01	–	8.8E-01	5.2E-01	6.0E-01	1.7E-01	1.5E-01
S5	7.4E-01	5.0E-01	8.8E-01	8.8E-01	–	5.5E-01	6.2E-01	1.7E-01	1.6E-01
S6	8.1E-01	7.5E-01	7.3E-01	5.2E-01	5.5E-01	–	2.7E-01	2.7E-01	7.7E-01
S7	4.2E-01	5.0E-01	5.2E-01	6.0E-01	6.2E-01	2.7E-01	–	5.8E-02	1.7E-01
S8	1.9E-01	1.8E-01	1.8E-01	1.7E-01	1.7E-01	2.7E-01	5.8E-02	–	6.8E-01
S9	2.9E-01	2.4E-01	2.5E-01	1.5E-01	1.6E-01	7.7E-01	1.7E-01	6.8E-01	–
Case 3									
	S1	S2	S3	S4	S5	S6	S7	S8	S9
S1	–	5.5E-01	6.9E-01	8.9E-01	7.5E-01	1.9E-01	7.3E-02	4.5E-01	1.5E-01
S2	5.5E-01	–	5.0E-01	6.9E-02	4.1E-01	1.7E-01	6.9E-02	6.2E-01	1.1E-01
S3	6.9E-01	5.0E-01	–	9.3E-01	6.9E-01	2.9E-01	**2.8E-02**	6.9E-01	1.1E-01
S4	8.9E-01	6.9E-02	9.3E-01	–	7.5E-01	3.3E-01	**2.3E-02**	7.3E-01	7.3E-02
S5	7.5E-01	4.1E-01	6.9E-01	7.5E-01	–	2.9E-01	5.5E-02	6.2E-01	1.2E-01
S6	1.9E-01	1.7E-01	2.9E-01	3.3E-01	2.9E-01	–	**2.3E-02**	8.4E-01	**4.9E-02**
S7	7.3E-02	6.9E-02	**2.8E-02**	**2.3E-02**	5.5E-02	**2.3E-02**	–	**4.6E-02**	9.7E-01
S8	4.5E-01	6.2E-01	6.9E-01	7.3E-01	6.2E-01	8.4E-01	**4.6E-02**	–	1.7E-01
S9	1.5E-01	1.1E-01	1.1E-01	7.3E-02	1.2E-01	**4.9E-02**	9.7E-01	1.7E-01	–
Case 4									
	S1	S2	S3	S4	S5	S6	S7	S8	S9
S1	–	2.0E-01	5.2E-01	3.4E-01	**2.6E-02**	3.3E-01	5.3E-01	1.9E-01	9.1E-02
S2	2.0E-01	–	9.3E-01	9.5E-01	1.7E-01	7.1E-01	8.7E-01	3.7E-01	1.9E-01
S3	5.2E-01	9.3E-01	–	5.2E-01	5.5E-02	7.9E-01	8.7E-01	3.9E-01	2.1E-01
S4	3.4E-01	9.5E-01	5.2E-01	–	6.9E-02	8.1E-01	8.9E-01	4.2E-01	2.7E-01
S5	**2.6E-02**	1.7E-01	5.5E-02	6.9E-02	–	9.1E-01	7.9E-01	4.4E-01	3.2E-01
S6	3.3E-01	7.1E-01	7.9E-01	8.1E-01	9.1E-01	–	8.7E-01	6.2E-01	6.6E-01
S7	5.3E-01	8.7E-01	8.7E-01	8.9E-01	7.9E-01	8.7E-01	–	4.5E-01	3.0E-01
S8	1.9E-01	3.7E-01	3.9E-01	4.2E-01	4.4E-01	6.2E-01	4.5E-01	–	8.5E-01
S9	9.1E-02	1.9E-01	2.1E-01	2.7E-01	3.2E-01	6.6E-01	3.0E-01	8.5E-01	–

S1:(λ1, ng) = (0.2, 10), S2:(λ1, ng) = (0.2, 20), S3:(λ1, ng) = (0.2, 30), S4:(λ1, ng) = (0.2, 50), S5:(λ1, ng) = (0.2, 80), S6:(λ1, ng) = (0.1, 20), S7:(λ1, ng) = (0.15, 20), S8:(λ1, ng) = (0.25, 20), S9:(λ1, ng) = (0.3, 20)

Table C7. Statistical analysis of MPEDE with different settings

	S1	S2	S3	S4	S5	S6	S7	S8	S9
Best									
Case 1	728.73	736.21	736.21	721.96	**736.21**	717.67	719.37	728.73	728.73
Case 2	774.33	810.40	810.40	826.72	826.76	**826.76**	816.95	801.68	810.40
Case 3	1204.89	1204.89	1204.89	1204.89	1204.89	1186.99	**1226.87**	1193.56	1224.30
Case 4	1437.79	**1437.79**	1437.79	1437.79	1437.79	1437.79	1437.79	1437.79	1437.79
Worst									
	S1	S2	S3	S4	S5	S6	S7	S8	S9
Case 1	462.77	462.71	462.54	462.48	462.63	443.48	447.19	**508.31**	480.72
Case 2	**638.21**	**638.21**	**638.21**	**638.21**	**638.21**	620.39	**638.21**	620.54	593.49
Case 3	593.46	593.46	593.46	593.46	593.46	319.45	820.37	609.61	**834.22**
Case 4	1007.11	1007.09	1007.00	1007.27	1007.37	1029.20	1024.41	**1034.22**	1027.74
Mean									
	S1	S2	S3	S4	S5	S6	S7	S8	S9
Case 1	598.01	599.03	599.25	597.24	597.67	572.76	583.89	**623.78**	597.59
Case 2	723.35	724.39	725.01	727.12	727.03	718.45	**733.60**	705.42	712.62
Case 3	1005.87	1004.30	997.05	995.02	1001.59	980.02	**1051.87**	992.06	1045.34
Case 4	1232.52	1246.39	1248.35	1253.39	1255.34	1264.13	1250.91	1277.94	**1300.66**
Median									
	S1	S2	S3	S4	S5	S6	S7	S8	S9
Case 1	581.68	588.56	584.12	585.59	584.12	562.14	594.57	**620.33**	593.18
Case 2	732.47	**732.56**	732.56	732.56	732.56	712.33	727.71	708.13	723.22
Case 3	1042.91	1038.56	1034.21	1036.23	1038.64	1002.59	**1065.61**	982.17	1047.20
Case 4	1205.08	1220.25	1240.71	1226.06	1230.94	1236.71	1217.35	1242.79	**1394.75**

S1:(λ1, ng) = (0.2, 10), S2:(λ1, ng) = (0.2, 20), S3:(λ1, ng) = (0.2, 30), S4:(λ1, ng) = (0.2, 50), S5:(λ1, ng) = (0.2, 80), S6:(λ1, ng) = (0.1, 20), S7:(λ1, ng) = (0.15, 20), S8:(λ1, ng) = (0.25, 20), S9:(λ1, ng) = (0.3, 20)

Table C8 provides the *p*-values obtained based on the Wilcoxon test of the results determined using different parameter values of SCA. It should be noted that the *p*-value obtained in all cases with the different parameter settings is lesser than the significance value, specifying the high dependency of SCA on the parameter value while solving the MUPP cases. The higher R+ value of the setting with a = 2 specified that the result obtained with this setting had outperformed others. Hence, the value of parameter a = 2 is selected for the performance analysis of SCA in Chapter 3.

Table C8. Results of the Wilcoxon signed-rank test on different settings of SCA

	Case 1								
	S1			S2			S3		
	p-value	R+	R−	*p*-value	R+	R−	*p*-value	R+	R−
S1	–	–	–	**8.3E-06**	0	351	**8.3E-06**	0	351
S2	**8.3E-06**	351	0	–	–	–	**2.7E-04**	32	319
S3	**8.3E-06**	351	0	**2.7E-04**	319	32	–	–	–
	Case 2								
	S1			S2			S3		
	p-value	R+	R−	*p*-value	R+	R−	*p*-value	R+	R−
S1	–	–	–	**8.3E-06**	0	351	**8.3E-06**	0	351
S2	**8.3E-06**	351	0	–	–	–	**1.2E-01**	114	237
S3	**8.3E-06**	351	0	**1.2E-01**	237	114	–	–	–
	Case 3								
	S1			S2			S3		
	p-value	R+	R−	*p*-value	R+	R−	*p*-value	R+	R−
S1	–	–	–	**8.3E-06**	0	351	**8.3E-06**	0	351
S2	**8.3E-06**	351	0	–	–	–	**1.3E-05**	4	347
S3	**8.3E-06**	351	0	**1.3E-05**	347	4	–	–	–
	Case 4								
	S1			S2			S3		
	p-value	R+	R−	*p*-value	R+	R−	*p*-value	R+	R−
S1	–	–	–	**8.3E-06**	0	351	**8.3E-06**	0	351
S2	**8.3E-06**	351	0	–	–	–	**9.3E-06**	1	350
S3	**8.3E-06**	351	0	**9.3E-06**	350	1	–	–	–

S1: a=0.5, S2: a=1, S3: a=2

The *p*-values among the various settings of SPMLFTLO provided in Table C9 indicate that none of the settings are statistically different from the others in all the cases. On analyzing the statistical results provided in Table C10, it is observed that the *best* value determined with the parameter value, N_g = 4 is better in Case 3 and Case 4, while the parameter value N_g = 25 is better in Case 1 and Case 2. Moreover, the *best* determined using N_g = 25 in Case 4 is close to the value determined using N_g=4. Additionally, based on all other statistic measures, the

value determined with $N_g = 25$ is better than $N_g = 4$ in all the cases except the median value in Case 4. So, we considered $N_g = 25$ for SPMGLFTLO in the rest of the analysis provided in Chapter 3.

Table C9. *p*-value of the Wilcoxon signed-rank test on different settings of SPMGLFTLO

				Case 1			
	S1	S2	S3	S4	S5	S6	S7
S1	–	2.48E-01	4.39E-01	3.95E-01	**1.10E-02**	5.20E-02	6.94E-02
S2	2.48E-01	–	6.57E-01	7.13E-01	**2.01E-02**	4.09E-01	9.90E-01
S3	4.39E-01	6.57E-01	–	7.98E-01	**6.33E-03**	4.85E-01	4.54E-01
S4	3.95E-01	7.13E-01	7.98E-01	–	**3.19E-02**	2.38E-01	3.16E-01
S5	**1.10E-02**	**2.01E-02**	**6.33E-03**	**3.19E-02**	–	2.00E-01	9.12E-02
S6	5.20E-02	4.09E-01	4.85E-01	2.38E-01	2.00E-01	–	6.75E-01
S7	6.94E-02	9.90E-01	4.54E-01	3.16E-01	9.12E-02	6.75E-01	–
				Case 2			
	S1	S2	S3	S4	S5	S6	S7
S1	–	2.48E-01	2.11E-01	6.94E-02	5.52E-02	**3.35E-03**	3.16E-01
S2	2.48E-01	–	8.51E-01	1.51E-01	1.37E-01	**4.90E-02**	6.20E-01
S3	2.11E-01	8.51E-01	–	8.09E-01	8.09E-01	3.16E-01	3.41E-01
S4	6.94E-02	1.51E-01	8.09E-01	–	9.70E-01	2.69E-01	5.20E-02
S5	5.52E-02	1.37E-01	8.09E-01	9.70E-01	–	1.31E-01	5.85E-02
S6	**3.35E-03**	**4.90E-02**	3.16E-01	2.69E-01	1.31E-01	–	**8.57E-03**
S7	3.16E-01	6.20E-01	3.41E-01	5.20E-02	5.85E-02	**8.57E-03**	–
				Case 3			
	S1	S2	S3	S4	S5	S6	S7
S1	–	6.56E-02	2.09E-01	**3.39E-02**	**6.97E-04**	**3.96E-04**	**3.64E-03**
S2	6.56E-02	–	6.38E-01	9.90E-01	8.65E-02	2.00E-01	8.09E-01
S3	2.09E-01	6.38E-01	–	3.81E-01	9.12E-02	8.65E-02	4.39E-01
S4	**3.39E-02**	9.90E-01	3.81E-01	–	**1.76E-02**	1.51E-01	8.29E-01
S5	**6.97E-04**	8.65E-02	9.12E-02	**1.76E-02**	–	6.96E-01	**3.64E-03**
S6	**3.96E-04**	2.00E-01	8.65E-02	1.51E-01	6.96E-01	–	**2.30E-02**
S7	**3.64E-03**	8.09E-01	4.39E-01	8.29E-01	**3.64E-03**	**2.30E-02**	–
				Case 4			
	S1	S2	S3	S4	S5	S6	S7
S1	–	2.92E-01	8.49E-01	1.12E-01	1.51E-01	2.69E-01	1.44E-01
S2	2.92E-01	–	5.85E-01	**9.94E-03**	**2.46E-02**	**3.85E-02**	**1.64E-02**
S3	8.49E-01	5.85E-01	–	**3.85E-02**	6.19E-02	1.51E-01	1.44E-01
S4	1.12E-01	**9.94E-03**	**3.85E-02**	–	6.57E-01	6.20E-01	4.85E-01
S5	1.51E-01	**2.46E-02**	6.19E-02	6.57E-01	–	8.69E-01	8.89E-01
S6	2.69E-01	**3.85E-02**	1.51E-01	6.20E-01	8.69E-01	–	9.90E-01
S7	1.44E-01	**1.64E-02**	1.44E-01	4.85E-01	8.89E-01	9.90E-01	–

S1: $N_g = 2$, S2: $N_g = 4$, S3: $N_g = 5$, S4: $N_g = 10$, S5: $N_g = 20$, S6: $N_g = 25$, S7: $N_g = 50$

Table C10. Statistical analysis of SPMGLFTLO with different settings

	S1	S2	S3	S4	S5	S6	S7
Best							
Case 1	732.19	727.90	728.37	728.84	734.50	**735.99**	713.45
Case 2	816.94	816.94	816.95	816.94	816.88	**825.38**	781.77
Case 3	1171.28	**1212.62**	1205.63	1186.97	1193.52	1204.30	1141.93
Case 4	1435.30	**1437.78**	1437.78	1437.78	1437.78	1437.41	1394.21
Worst							
Case 1	423.79	497.08	446.40	476.91	**578.17**	534.61	526.19
Case 2	489.71	574.20	593.49	645.03	593.49	593.49	**683.44**
Case 3	681.40	659.88	744.35	853.10	**975.66**	971.51	938.85
Case 4	970.16	941.02	847.15	**1109.80**	1042.54	1013.27	1086.69
Mean							
Case 1	605.19	629.50	624.77	621.73	**670.73**	646.18	639.53
Case 2	702.42	730.41	742.41	751.25	748.65	**764.90**	732.60
Case 3	981.13	1038.98	1023.53	1052.23	**1093.75**	1084.52	1044.64
Case 4	1253.08	1210.49	1235.04	**1320.63**	1307.07	1296.24	1304.85
Median							
Case 1	577.00	627.42	649.93	582.96	**679.24**	638.45	637.96
Case 2	733.79	733.23	765.97	754.38	754.75	**775.02**	733.09
Case 3	989.63	1076.39	1049.57	1072.41	**1088.16**	1061.56	1043.95
Case 4	1275.63	1143.10	1237.29	**1394.75**	1394.75	1387.13	1325.91

S1: N_g = 2, S2: N_g = 4, S3: N_g = 5, S4: N_g = 10, S5: N_g = 20, S6: N_g = 25, S7: N_g = 50

Appendix D: Data for the scheduling problem

The details regarding the number of orders and machines corresponding to each problem are provided in Table D1.

Table D1. Details of the number of orders and machines in each problem

Problem	**P1**	**P2**	**P3**	**P4**	**P5**
Number of orders	3	7	12	15	20
Number of machines	2	3	3	5	5

Each of the five problems has two different data corresponding to the processing time, whereas, for both the data set, the release date, the due date, and the processing cost of the orders are the same. The processing time required by all machines of data set 1 is longer than that of data set 2, and hence a larger feasible region is available corresponding to data set 2 than data set 1. The resulting ten problem instances are named according to the problem number and the dataset number. For example, P4S2 indicates problem four with the second dataset. All the required data (Jain & Grossmann, 2001) corresponding to the ten problems is provided in Table D2 to Table D18.

Table D2. Data for P1S1 and P1S2

Orders (*i*)	r_i	d_i	**Cost**		**Duration**			
					S1		**S2**	
			***M*1**	***M*2**	***M*1**	***M*2**	***M*1**	***M*2**
1	2	16	10	6	10	14	5	7
2	3	13	8	5	6	8	3	4
3	4	21	12	7	11	16	5	7

Table D3. Data for P2S1 and P2S2

Orders (*i*)	r_i	d_i	Cost			Duration					
						S1			S2		
			*M*1	*M*2	*M*3	*M*1	*M*2	*M*3	*M*1	*M*2	*M*3
1	2	16	10	6	8	10	14	12	5	7	6
2	3	13	8	5	6	6	8	7	3	4	3
3	4	21	12	7	10	11	16	13	2	4	3
4	5	28	10	6	8	6	12	8	3	6	4
5	10	24	8	5	7	10	16	12	2	4	3
6	1	28	12	7	10	7	12	10	1	3	2
7	2	23	12	7	10	10	8	10	1	2	1

Table D4. Data for P3S1 and P3S2

Orders (*i*)	r_i	d_i	Cost			Duration					
						S1			S2		
			*M*1	*M*2	*M*3	*M*1	*M*2	*M*3	*M*1	*M*2	*M*3
1	2	36	10	6	8	10	14	12	5	7	6
2	3	33	8	5	6	6	8	7	3	4	3
3	4	31	12	7	10	11	16	13	2	4	3
4	5	38	10	6	8	6	12	8	3	6	4
5	10	34	8	5	7	10	16	12	2	4	3
6	1	38	12	7	10	7	12	10	1	3	2
7	2	33	12	10	11	10	13	10	1	2	1
8	4	25	9	5	7	4	10	8	2	5	4
9	10	38	10	6	8	2	4	3	4	6	6
10	1	39	8	5	7	7	14	11	3	5	2
11	5	30	15	9	12	8	16	12	2	3	2
12	2	20	13	7	10	3	6	5	2	6	4

Table D5. Data regarding the release date, due date, and cost for P4S1 and P4S2

Orders (*i*)	r_i	d_i	**Cost**				
			***M*1**	***M*2**	***M*3**	***M*4**	***M*5**
1	2	33	10	6	8	9	9
2	3	34	8	5	6	7	7
3	4	31	12	7	10	11	10
4	5	33	10	6	8	9	8
5	10	34	8	5	6	7	7
6	1	34	12	7	10	11	10
7	2	33	12	10	11	12	11
8	4	25	9	5	7	9	8
9	10	38	10	6	8	9	8
10	1	37	8	5	6	7	6
11	5	30	15	9	12	14	13
12	2	20	13	7	10	12	11
13	4	32	9	5	6	8	7
14	6	20	10	6	8	10	9
15	2	25	8	5	6	7	7

Table D6. Data regarding the duration for P4S1 and P4S2

Orders (*i*)	**Duration**									
	S1					**S2**				
	***M*1**	***M*2**	***M*3**	***M*4**	***M*5**	***M*1**	***M*2**	***M*3**	***M*4**	***M*5**
1	10	14	12	11	13	5	7	6	5	6
2	6	8	7	6	7	3	4	3	3	4
3	11	16	13	11	12	2	4	3	2	3
4	6	12	8	7	8	3	6	4	3	4
5	10	16	12	12	13	2	4	3	2	2
6	7	12	10	8	9	1	3	2	2	2
7	10	13	10	11	12	1	2	1	1	1
8	4	10	8	5	6	2	5	4	3	3
9	2	4	3	2	3	4	6	6	5	5
10	7	14	11	8	10	2	5	3	2	3
11	8	16	12	10	11	2	3	2	2	2
12	3	6	5	4	5	2	6	4	3	3
13	4	10	7	5	6	1	3	3	2	2
14	2	4	4	3	3	2	5	5	2	3
15	7	14	13	10	11	4	7	6	4	5

Table D7. Data regarding the release date, due date, and cost for P5S1 and P4S1

Orders (*i*)	r_i	d_i	**Cost**				
			***M*1**	***M*2**	***M*3**	***M*4**	***M*5**
1	2	33	10	6	8	9	9
2	3	34	8	5	6	7	7
3	4	31	12	7	10	11	10
4	5	33	10	6	8	9	8
5	10	34	8	5	6	7	7
6	1	34	12	7	10	11	10
7	2	33	12	10	11	12	11
8	4	25	9	5	7	9	8
9	10	38	10	6	8	9	8
10	1	37	8	5	6	7	6
11	5	30	15	9	12	14	13
12	2	20	13	7	10	12	11
13	4	32	9	5	6	8	7
14	6	20	10	6	8	10	9
15	2	25	8	5	6	7	7
16	2	34	9	5	7	9	8
17	3	37	10	6	8	9	8
18	7	38	8	5	6	7	6
19	6	32	15	9	12	14	13
20	0	30	13	7	10	12	11

Table D8. Data regarding the duration for P5S1 and P5S2

Orders (*i*)	**Duration**									
	S1					**S2**				
	***M*1**	***M*2**	***M*3**	***M*4**	***M*5**	***M*1**	***M*2**	***M*3**	***M*4**	***M*5**
1	10	14	12	11	13	5	7	6	5	6
2	6	8	7	6	7	3	4	3	3	4
3	11	16	13	11	12	2	4	3	2	3
4	6	12	8	7	8	3	6	4	3	4
5	10	16	12	12	13	2	4	3	2	2
6	7	12	10	8	9	1	3	2	2	2
7	10	13	10	11	12	1	2	1	1	1
8	4	10	8	5	6	2	5	4	3	3
9	2	4	3	2	3	4	6	6	5	5
10	7	14	11	8	10	2	5	3	2	3
11	8	16	12	10	11	2	3	2	2	2
12	3	6	5	4	5	2	6	4	3	3
13	4	10	7	5	6	1	3	3	2	2
14	2	4	4	3	3	2	5	5	2	3
15	7	14	13	10	11	4	7	6	4	5
16	3	8	7	5	6	2	4	3	2	3
17	6	12	10	7	8	3	6	4	3	4
18	2	8	6	13	4	2	4	3	2	2
19	4	7	6	5	5	1	3	2	2	2
20	5	7	7	6	6	1	2	1	1	1

Table D9. Data regarding the release date and due date for P6S1 and P6S1

Orders (*i*)	r_i	d_i	**Orders (*i*)**	r_i	d_i	**Orders (*i*)**	r_i	d_i	**Orders (*i*)**	r_i	d_i
1	3	52	26	2	52	51	0	38	76	1	55
2	4	49	27	3	55	52	2	52	77	5	48
3	5	51	28	6	38	53	3	52	78	6	38
4	2	51	29	2	38	54	4	49	79	10	56
5	4	43	30	4	50	55	5	51	80	2	51
6	0	38	31	2	43	56	12	52	81	1	52
7	2	52	32	7	56	57	1	52	82	10	52
8	3	55	33	0	38	58	2	51	83	2	38
9	6	50	34	2	52	59	4	43	84	7	56
10	2	51	35	3	52	60	7	56	85	4	50
11	1	55	36	4	49	61	1	55	86	2	43
12	5	48	37	5	51	62	5	48	87	0	46
13	6	38	38	12	52	63	3	55	88	4	48
14	10	56	39	1	52	64	6	50	89	6	38
15	2	51	40	2	51	65	2	51	90	2	43
16	1	52	41	4	43	66	6	50	91	2	52
17	10	52	42	7	56	67	2	38	92	3	55
18	2	38	43	1	55	68	7	56	93	6	38
19	7	56	44	5	48	69	2	51	94	2	38
20	4	50	45	3	55	70	4	43	95	4	50
21	2	43	46	6	50	71	0	38	96	2	43
22	0	46	47	2	51	72	2	52	97	7	56
23	4	48	48	6	50	73	3	55	98	0	38
24	6	38	49	2	38	74	6	50	99	2	52
25	2	43	50	7	56	75	2	51	100	3	52

Table D10. Data regarding the cost for order 1 to 34 of P6S1 and P6S1

Order	M 1	M 2	M 3	M 4	M 5	M 6	M 7	M 8	M 9	M 10	M 11	M 12	M 13	M 14	M 15	M 16	M 17	M 18	M 19	M 20
1	13	6	9	12	9	9	6	8	7	12	8	12	9	7	10	12	13	10	11	7
2	13	7	11	8	10	11	11	12	13	10	8	12	7	12	7	13	12	10	10	9
3	14	7	7	11	12	7	11	11	8	10	11	14	9	11	11	13	7	7	11	7
4	14	7	8	8	10	11	14	8	7	9	12	7	12	7	7	9	8	12	9	12
5	14	7	10	7	13	7	13	12	13	11	12	14	11	12	11	9	9	10	8	7
6	13	6	7	10	10	12	8	11	11	12	11	13	7	12	6	13	9	10	7	9
7	13	6	7	11	13	8	8	7	7	12	12	7	7	7	8	13	10	11	7	12
8	14	6	9	11	12	12	14	12	12	11	11	14	8	12	7	12	6	7	6	12
9	12	7	9	12	12	12	12	7	9	12	10	11	12	8	9	7	10	8	10	12
10	14	6	10	8	6	7	8	12	11	9	12	7	10	12	14	7	7	7	8	7
11	14	7	11	9	12	11	8	12	7	11	10	14	9	11	14	13	11	10	10	9
12	13	6	11	7	6	9	13	7	10	9	6	12	8	10	13	7	12	7	7	8
13	14	6	12	10	12	11	14	12	8	8	9	10	6	10	6	11	7	7	11	12
14	13	6	12	13	9	8	6	7	6	12	6	13	6	10	10	10	11	9	11	10
15	15	7	10	14	9	8	9	10	11	7	11	9	7	11	13	12	12	12	8	9
16	13	6	8	13	13	11	11	13	9	9	8	12	9	10	7	11	8	12	12	10
17	14	7	10	7	11	9	7	12	10	10	10	8	11	8	14	8	13	8	9	10
18	15	6	9	14	13	7	6	12	9	7	9	14	12	12	13	11	7	12	10	8
19	9	5	6	7	7	5	5	5	6	5	6	7	7	7	7	7	6	7	5	6
20	12	6	10	8	11	11	8	8	6	11	7	12	6	9	9	7	6	10	11	8
21	15	6	6	9	8	12	10	9	12	12	10	14	6	12	14	9	10	7	11	7
22	14	6	10	7	13	9	14	9	8	12	6	7	11	11	8	12	13	8	7	11
23	14	6	11	13	10	8	6	10	13	7	8	14	8	9	13	13	9	12	9	10
24	9	5	7	6	5	7	7	7	7	5	6	6	6	7	5	7	7	6	7	7
25	14	6	10	8	7	8	6	8	11	7	10	10	13	12	14	11	8	12	7	9
26	14	6	8	13	8	8	11	12	6	12	7	9	6	8	14	8	11	11	11	12
27	14	6	9	13	11	12	12	10	7	7	8	11	6	10	9	12	6	12	7	9
28	13	6	6	12	10	12	6	10	11	9	7	12	13	10	13	8	11	7	9	7
29	13	6	12	9	7	9	13	10	7	10	6	10	12	11	6	7	7	11	8	12
30	13	7	8	7	11	7	7	7	9	12	10	8	8	9	11	7	13	12	7	11
31	13	6	10	13	7	9	7	8	6	11	9	12	11	10	12	9	6	8	6	7
32	14	6	9	9	11	7	8	13	6	8	6	14	9	9	12	13	7	7	6	10
33	8	5	7	5	6	6	5	5	6	5	5	6	6	5	6	6	7	7	6	5
34	14	7	10	14	9	7	13	11	13	10	9	12	8	11	10	13	9	7	10	11

Table D11. Data regarding the cost for order 35 to 68 of P6S1 and P6S1

Order	M 1	M 2	M 3	M 4	M 5	M 6	M 7	M 8	M 9	M 10	M 11	M 12	M 13	M 14	M 15	M 16	M 17	M 18	M 19	M 20
35	13	6	8	13	10	10	9	9	12	10	12	9	11	10	6	9	8	10	12	10
36	9	5	7	7	6	7	5	5	5	5	5	7	5	5	5	6	6	7	6	7
37	14	7	10	9	11	8	14	7	8	7	9	11	11	9	12	7	13	12	8	7
38	13	6	10	13	6	12	10	12	8	10	6	11	7	9	7	8	7	8	11	9
39	14	6	10	14	9	11	9	7	10	7	6	11	7	9	7	10	9	7	8	7
40	14	7	10	14	8	8	7	12	7	12	9	8	13	11	9	12	9	11	12	8
41	14	6	9	14	9	12	13	13	7	12	12	10	6	7	6	8	11	12	8	12
42	14	6	12	14	11	11	13	12	7	9	9	14	6	11	7	10	13	9	9	10
43	14	6	10	14	6	7	13	8	10	7	7	9	6	12	10	10	13	9	10	7
44	13	7	7	13	10	10	10	8	13	11	9	8	12	8	9	12	10	12	10	8
45	13	7	8	13	12	11	8	11	13	7	9	8	10	10	11	10	12	8	11	9
46	13	6	10	7	11	7	6	13	12	8	10	10	11	10	11	11	6	7	12	10
47	14	6	12	12	12	12	14	12	11	10	11	12	13	9	6	8	13	11	11	11
48	14	6	10	9	12	12	9	13	10	10	10	9	10	9	13	8	8	7	8	8
49	8	5	6	7	5	6	5	7	5	7	6	7	7	7	7	7	7	5	7	7
50	8	5	6	5	5	7	6	7	5	5	5	6	5	5	7	7	5	7	6	7
51	13	6	11	10	9	10	10	10	9	12	10	7	6	9	10	7	10	11	6	11
52	13	6	11	13	12	7	6	7	6	7	7	8	8	10	9	12	11	8	11	12
53	13	6	6	7	11	12	10	11	13	9	6	10	7	10	13	12	9	11	10	12
54	13	6	11	8	12	11	13	9	7	12	12	13	8	12	7	7	7	11	7	11
55	14	6	6	9	6	7	14	10	8	8	10	12	6	12	12	11	7	8	8	10
56	8	5	6	5	6	6	6	6	6	5	7	5	5	5	5	7	6	7	7	6
57	9	5	6	5	5	6	5	5	5	5	5	5	6	7	7	7	7	5	7	7
58	15	6	11	14	8	8	7	12	6	9	9	13	13	9	11	8	7	8	6	12
59	12	6	9	11	12	10	10	8	11	8	12	10	6	7	11	7	10	12	7	8
60	8	5	7	5	7	6	7	6	7	6	7	6	5	6	6	6	7	5	6	7
61	15	6	7	7	8	7	6	7	7	8	7	13	13	12	13	12	13	7	12	9
62	13	6	6	10	7	11	7	12	12	11	11	12	9	8	10	13	7	10	6	11
63	14	6	7	13	6	10	12	11	8	12	6	7	6	11	6	13	12	7	8	7
64	14	6	7	13	12	8	8	10	6	8	9	14	7	12	9	8	13	7	9	7
65	14	6	10	8	13	8	13	13	12	12	7	14	6	9	9	8	12	9	12	7
66	14	6	7	14	10	10	9	7	11	11	9	10	10	11	6	8	6	8	6	8
67	14	6	8	14	7	7	10	10	9	12	7	12	7	8	10	12	9	8	9	11
68	13	7	9	8	8	8	8	8	7	8	9	7	9	7	13	9	10	12	8	7

Table D12. Data regarding the cost for order 69 to 100 of P6S1 and P6S1

Order	M 1	M 2	M 3	M 4	M 5	M 6	M 7	M 8	M 9	M 10	M 11	M 12	M 13	M 14	M 15	M 16	M 17	M 18	M 19	M 20
69	13	6	11	8	13	12	12	7	6	9	6	7	13	11	6	11	13	8	6	12
70	13	6	11	9	11	12	9	9	10	9	9	11	7	7	6	9	13	8	9	10
71	13	7	7	9	13	8	10	11	11	10	7	9	11	11	11	11	8	10	10	12
72	13	7	10	12	10	12	12	12	9	9	8	10	13	11	11	10	11	9	8	11
73	13	6	8	12	13	9	6	13	6	12	8	8	9	10	7	7	6	10	7	8
74	15	7	7	8	10	12	13	10	8	7	11	8	13	11	13	8	7	11	9	12
75	14	6	10	9	7	8	6	7	9	8	7	14	9	8	6	10	11	7	12	11
76	12	6	10	12	12	10	9	11	9	8	11	10	12	8	11	9	6	12	12	7
77	13	7	9	13	11	11	8	10	10	7	9	12	7	8	7	8	8	9	9	7
78	15	7	11	8	13	7	12	13	7	7	7	7	11	12	10	9	12	11	10	10
79	14	7	11	9	7	12	14	7	8	7	8	8	10	11	9	8	8	11	9	11
80	13	6	9	9	7	7	6	11	7	7	6	11	7	9	13	7	10	9	12	8
81	11	7	7	7	8	9	7	11	8	10	10	11	10	8	10	7	11	7	10	7
82	13	6	6	8	9	10	6	7	7	8	8	10	8	8	12	8	13	10	9	11
83	14	6	6	9	10	8	8	9	8	11	7	10	9	8	14	13	12	9	11	12
84	15	6	12	14	6	7	11	7	13	7	8	10	9	7	8	13	11	9	7	10
85	13	6	12	12	12	10	6	8	13	12	6	10	10	11	10	8	13	7	12	11
86	13	6	10	8	9	8	10	7	8	7	12	9	10	9	12	13	6	11	7	7
87	14	6	7	11	10	8	14	9	12	12	7	10	7	10	8	8	6	10	9	8
88	14	6	10	12	7	7	14	11	8	12	10	9	6	9	7	11	7	12	12	11
89	14	6	8	9	10	11	9	11	8	10	6	14	10	9	12	9	11	7	7	12
90	14	6	12	12	8	8	9	10	11	12	6	10	10	10	8	11	7	8	7	11
91	14	6	7	8	6	7	6	12	9	11	9	13	12	11	14	13	9	10	7	8
92	13	6	6	13	8	10	9	11	11	8	8	10	12	7	8	9	6	9	12	12
93	12	6	6	12	9	9	6	11	7	9	6	10	8	8	8	9	7	10	6	9
94	9	5	7	6	7	6	5	6	6	6	5	5	7	6	7	6	5	5	5	7
95	8	5	6	6	5	7	7	7	7	5	5	5	6	6	7	7	7	7	7	7
96	13	6	12	8	6	12	6	8	7	11	6	11	12	11	13	8	12	8	6	12
97	15	6	6	14	11	10	14	7	8	10	8	7	11	9	14	7	12	8	12	12
98	13	6	7	13	9	10	12	11	6	10	10	11	6	10	6	10	10	11	9	10
99	13	6	6	12	6	7	7	11	11	7	8	7	8	8	9	10	8	9	6	9
100	14	6	8	14	11	8	9	7	10	9	11	9	6	11	7	9	7	11	12	8

Table D13. Data regarding the duration for order 1 to 34 of P6S1

Order	M 1	M 2	M 3	M 4	M 5	M 6	M 7	M 8	M 9	M 10	M 11	M 12	M 13	M 14	M 15	M 16	M 17	M 18	M 19	M 20
1	3	13	8	9	5	11	12	8	10	13	7	11	12	10	5	13	4	3	4	10
2	2	9	7	6	5	2	7	7	6	6	5	6	7	2	4	6	5	4	7	5
3	3	16	6	3	4	10	4	10	4	5	9	7	6	6	13	9	5	7	6	3
4	3	13	13	9	13	8	9	3	10	6	12	3	7	11	9	6	9	13	10	6
5	2	9	7	5	5	6	4	6	7	4	7	3	5	6	7	7	5	2	5	7
6	3	16	5	9	10	12	13	7	11	4	5	11	10	4	3	8	4	4	9	13
7	3	14	10	10	9	12	8	6	8	12	11	3	4	6	9	12	3	12	5	4
8	4	14	6	8	12	6	6	11	12	11	11	9	7	6	4	4	9	5	4	5
9	2	8	7	7	6	6	6	6	4	7	7	7	4	5	5	5	7	6	4	7
10	3	13	8	7	9	3	13	6	10	3	3	9	11	8	12	4	13	4	3	7
11	3	13	9	12	4	9	13	13	10	9	11	8	4	9	13	13	13	11	3	5
12	2	9	5	7	6	2	5	2	5	6	7	6	5	5	7	7	7	6	5	6
13	4	13	8	8	5	4	8	7	4	4	13	4	7	6	7	8	4	6	9	6
14	3	13	3	13	11	11	11	9	4	5	11	3	13	13	13	9	11	13	3	11
15	2	9	5	2	5	7	6	5	4	4	5	6	3	3	5	7	7	6	5	5
16	2	8	4	6	5	3	7	5	6	2	7	6	6	5	5	7	5	7	6	6
17	2	9	7	6	5	4	7	5	5	3	5	2	5	6	4	4	7	7	5	7
18	3	13	13	12	5	11	8	13	6	7	6	9	13	13	12	6	7	3	7	8
19	2	8	5	4	6	6	5	5	4	5	6	2	7	6	3	7	6	7	7	6
20	3	16	5	3	6	8	5	5	11	5	4	7	10	5	3	8	6	12	9	12
21	2	9	6	6	3	6	7	5	7	7	7	3	6	4	5	2	6	7	6	6
22	2	9	6	6	6	2	6	4	7	5	5	5	5	5	5	6	3	5	6	4
23	3	13	12	4	3	11	13	12	13	10	12	4	5	12	11	10	10	5	7	10
24	3	15	5	11	12	3	3	8	3	3	6	4	12	10	3	5	6	7	7	5
25	2	9	7	2	7	7	6	6	5	2	6	6	5	4	3	5	6	3	5	7
26	3	14	13	10	3	9	7	12	10	6	9	8	5	3	13	10	9	7	13	7
27	3	14	5	10	10	10	12	9	3	11	12	10	3	3	11	4	3	7	13	12
28	4	16	8	10	11	12	5	7	5	7	12	13	13	4	12	8	8	7	11	4
29	2	9	6	2	6	3	6	7	6	4	5	3	7	5	4	5	6	6	4	6
30	2	9	7	5	6	7	4	3	7	6	6	7	4	7	7	7	7	2	6	4
31	3	16	13	10	9	3	10	9	8	4	4	4	9	8	3	4	10	12	13	5
32	2	9	7	6	5	6	5	3	6	2	5	5	5	2	5	6	4	7	5	5
33	2	8	6	7	7	5	5	7	5	7	5	2	7	5	5	6	5	5	7	5
34	3	15	8	7	5	3	5	11	8	6	3	5	7	11	12	10	13	5	8	4

Table D14. Data regarding the duration for order 34 to 68 of P6S1

Order	M 1	M 2	M 3	M 4	M 5	M 6	M 7	M 8	M 9	M 10	M 11	M 12	M 13	M 14	M 15	M 16	M 17	M 18	M 19	M 20
35	2	8	5	7	7	6	5	3	5	3	7	6	7	6	7	5	7	4	6	6
36	2	9	5	6	3	6	7	5	7	6	6	5	4	4	7	7	5	6	7	6
37	2	9	4	6	5	6	7	5	7	7	6	6	5	6	6	6	5	2	4	6
38	2	9	5	6	5	6	6	4	7	5	7	5	6	6	5	7	3	2	6	4
39	3	14	10	4	3	6	6	5	6	5	13	4	9	3	10	4	3	3	10	8
40	2	8	5	7	6	6	6	2	5	5	7	5	5	7	4	5	6	3	7	4
41	2	8	7	6	3	5	7	4	7	5	6	5	7	4	5	2	3	6	4	6
42	3	13	3	5	6	8	13	5	11	5	3	13	11	10	4	9	8	8	11	13
43	3	12	12	4	3	10	6	11	12	7	3	12	8	4	9	12	9	6	12	8
44	2	8	3	5	7	7	5	4	3	2	3	6	6	5	5	4	7	4	6	5
45	2	8	5	7	5	6	5	5	7	7	5	6	5	2	5	3	5	6	7	5
46	3	13	8	5	10	6	6	3	10	11	8	12	11	9	12	6	13	8	3	9
47	2	9	6	5	6	5	6	5	5	5	3	2	7	7	7	7	6	6	5	7
48	3	13	10	6	13	10	11	3	11	3	3	7	7	7	9	13	10	10	6	7
49	3	13	6	7	11	13	13	3	4	12	12	12	11	4	10	12	12	5	13	12
50	2	8	5	7	7	3	4	2	7	6	5	7	7	6	5	6	6	7	7	6
51	2	9	7	5	7	7	5	7	6	7	5	4	5	4	7	5	6	6	7	3
52	2	9	5	4	6	3	5	7	6	7	7	5	4	3	6	2	4	6	5	6
53	3	12	5	12	6	4	3	5	3	6	7	12	3	3	4	6	5	7	12	8
54	2	9	3	6	5	7	7	6	6	4	5	6	7	5	5	5	5	5	5	2
55	2	9	6	5	5	6	6	6	6	4	4	5	5	7	6	2	4	3	5	5
56	2	9	6	2	7	4	5	5	5	4	7	3	5	7	7	5	3	5	7	7
57	2	9	7	3	7	6	7	3	5	5	7	6	7	5	3	5	7	6	7	2
58	3	13	8	9	13	4	9	3	11	13	11	8	8	12	4	11	4	6	8	7
59	4	13	12	7	6	4	8	10	7	12	13	12	4	9	8	12	6	11	7	7
60	2	9	7	6	5	6	7	7	4	5	6	5	5	7	7	7	7	5	7	7
61	2	9	6	6	7	5	5	6	6	2	4	6	7	5	7	6	6	7	3	5
62	2	9	6	2	5	5	5	5	5	5	6	7	3	6	7	5	7	7	5	5
63	3	13	7	13	7	12	11	9	3	7	6	7	10	3	10	3	8	10	7	12
64	3	13	4	8	8	7	13	5	13	8	6	3	3	11	5	4	9	3	6	11
65	2	8	6	4	7	4	7	2	5	5	4	4	6	2	3	7	4	6	7	5
66	2	9	6	6	7	7	7	6	3	6	7	5	5	4	5	6	4	6	6	2
67	4	15	11	4	13	9	9	10	10	6	4	13	4	10	8	9	9	5	8	5
68	2	8	5	5	5	5	7	7	6	5	6	2	5	7	4	7	7	7	4	4

Table D15. Data regarding the duration for order 69 to 100 of P6S1

Order	M 1	M 2	M 3	M 4	M 5	M 6	M 7	M 8	M 9	M 10	M 11	M 12	M 13	M 14	M 15	M 16	M 17	M 18	M 19	M 20
69	2	8	6	2	7	4	7	7	5	6	7	6	5	5	6	6	5	7	7	4
70	2	8	5	7	5	5	6	2	3	7	7	7	7	6	3	4	7	5	7	5
71	2	8	6	6	5	6	7	6	6	2	5	7	7	5	6	5	5	7	5	6
72	4	13	11	7	4	4	13	12	7	4	4	4	9	12	5	11	10	10	9	11
73	4	16	6	5	11	13	4	8	4	11	12	7	10	7	13	5	4	10	11	11
74	3	13	10	10	7	10	6	13	3	10	5	11	5	9	8	11	9	5	8	5
75	2	8	7	2	5	2	6	3	7	5	6	7	7	6	3	6	5	6	5	6
76	3	13	13	13	7	11	13	5	13	11	13	3	4	4	6	10	8	10	6	9
77	2	8	3	2	7	7	6	5	7	5	5	6	6	6	5	6	5	5	7	6
78	2	9	5	2	7	7	5	7	5	7	5	5	6	6	3	5	7	7	5	4
79	3	15	9	5	4	10	7	3	9	6	5	11	7	13	13	3	10	8	12	6
80	3	13	4	12	6	3	3	13	8	8	5	8	13	4	4	11	8	10	7	7
81	3	13	6	3	11	9	6	3	10	3	12	13	7	4	11	9	6	3	8	6
82	2	9	5	7	4	2	5	5	3	4	7	5	6	7	7	5	3	7	7	7
83	2	8	6	6	6	6	7	2	6	7	5	5	7	6	6	2	5	5	6	7
84	3	13	8	10	9	7	7	3	7	12	11	12	11	12	5	3	3	13	6	7
85	2	8	4	6	5	2	3	3	5	6	5	6	5	2	5	7	5	3	7	6
86	2	9	7	7	5	7	3	5	7	7	5	2	6	7	7	2	4	6	6	5
87	2	8	3	2	6	7	7	7	5	2	5	7	5	6	5	3	5	7	6	5
88	2	9	7	5	7	6	5	7	6	6	3	7	7	3	6	6	7	5	5	6
89	3	13	13	10	9	10	7	4	5	5	3	12	6	10	11	13	13	10	3	5
90	2	9	6	6	6	4	4	5	7	6	5	2	5	7	6	5	3	4	5	5
91	3	12	11	7	9	3	9	3	12	3	3	12	11	8	8	7	9	10	4	5
92	2	9	6	6	7	7	6	2	7	2	5	6	7	7	6	5	3	7	7	6
93	3	13	7	13	8	4	9	3	4	3	4	6	5	7	11	10	12	5	3	3
94	3	13	11	3	7	3	11	8	3	6	9	9	5	5	4	12	13	11	12	12
95	2	9	7	3	7	6	6	7	3	7	3	3	5	7	5	2	6	2	5	4
96	3	12	6	5	8	3	8	10	8	11	4	6	12	5	4	6	6	8	3	12
97	3	13	5	5	9	12	3	12	11	11	12	7	13	5	10	10	7	10	12	9
98	3	12	9	5	10	12	5	3	5	7	12	3	9	8	5	6	10	10	11	4
99	2	8	3	7	3	6	5	7	7	7	3	3	6	6	5	7	5	6	5	7
100	2	8	5	5	7	6	5	5	7	7	5	2	7	4	6	6	7	6	7	6

Table D16. Data regarding the duration for order 1 to 34 of P6S2

Order	M 1	M 2	M 3	M 4	M 5	M 6	M 7	M 8	M 9	M 10	M 11	M 12	M 13	M 14	M 15	M 16	M 17	M 18	M 19	M 20
1	13	6	9	12	9	9	6	8	7	12	8	12	9	7	10	12	13	10	11	7
2	13	7	11	8	10	11	11	12	13	10	8	12	7	12	7	13	12	10	10	9
3	14	7	7	11	12	7	11	11	8	10	11	14	9	11	11	13	7	7	11	7
4	14	7	8	8	10	11	14	8	7	9	12	7	12	7	7	9	8	12	9	12
5	14	7	10	7	13	7	13	12	13	11	12	14	11	12	11	9	9	10	8	7
6	13	6	7	10	10	12	8	11	11	12	11	13	7	12	6	13	9	10	7	9
7	13	6	7	11	13	8	8	7	7	12	12	7	7	7	8	13	10	11	7	12
8	14	6	9	11	12	12	14	12	12	11	11	14	8	12	7	12	6	7	6	12
9	12	7	9	12	12	12	12	7	9	12	10	11	12	8	9	7	10	8	10	12
10	14	6	10	8	6	7	8	12	11	9	12	7	10	12	14	7	7	7	8	7
11	14	7	11	9	12	11	8	12	7	11	10	14	9	11	14	13	11	10	10	9
12	13	6	11	7	6	9	13	7	10	9	6	12	8	10	13	7	12	7	7	8
13	14	6	12	10	12	11	14	12	8	8	9	10	6	10	6	11	7	7	11	12
14	13	6	12	13	9	8	6	7	6	12	6	13	6	10	10	10	11	9	11	10
15	15	7	10	14	9	8	9	10	11	7	11	9	7	11	13	12	12	12	8	9
16	13	6	8	13	13	11	11	13	9	9	8	12	9	10	7	11	8	12	12	10
17	14	7	10	7	11	9	7	12	10	10	10	8	11	8	14	8	13	8	9	10
18	15	6	9	14	13	7	6	12	9	7	9	14	12	12	13	11	7	12	10	8
19	9	5	6	7	7	5	5	5	6	5	6	7	7	7	7	7	6	7	5	6
20	12	6	10	8	11	11	8	8	6	11	7	12	6	9	9	7	6	10	11	8
21	15	6	6	9	8	12	10	9	12	12	10	14	6	12	14	9	10	7	11	7
22	14	6	10	7	13	9	14	9	8	12	6	7	11	11	8	12	13	8	7	11
23	14	6	11	13	10	8	6	10	13	7	8	14	8	9	13	13	9	12	9	10
24	9	5	7	6	5	7	7	7	7	5	6	6	6	7	5	7	7	6	7	7
25	14	6	10	8	7	8	6	8	11	7	10	10	13	12	14	11	8	12	7	9
26	14	6	8	13	8	8	11	12	6	12	7	9	6	8	14	8	11	11	11	12
27	14	6	9	13	11	12	12	10	7	7	8	11	6	10	9	12	6	12	7	9
28	13	6	6	12	10	12	6	10	11	9	7	12	13	10	13	8	11	7	9	7
29	13	6	12	9	7	9	13	10	7	10	6	10	12	11	6	7	7	11	8	12
30	13	7	8	7	11	7	7	7	9	12	10	8	8	9	11	7	13	12	7	11
31	13	6	10	13	7	9	7	8	6	11	9	12	11	10	12	9	6	8	6	7
32	14	6	9	9	11	7	8	13	6	8	6	14	9	9	12	13	7	7	6	10
33	8	5	7	5	6	6	5	5	6	5	5	6	6	5	6	6	7	7	6	5
34	14	7	10	14	9	7	13	11	13	10	9	12	8	11	10	13	9	7	10	11

Table D17. Data regarding the duration for order 35 to 68 of P6S2

Order	M 1	M 2	M 3	M 4	M 5	M 6	M 7	M 8	M 9	M 10	M 11	M 12	M 13	M 14	M 15	M 16	M 17	M 18	M 19	M 20
35	13	6	8	13	10	10	9	9	12	10	12	9	11	10	6	9	8	10	12	10
36	9	5	7	7	6	7	5	5	5	5	5	7	5	5	5	6	6	7	6	7
37	14	7	10	9	11	8	14	7	8	7	9	11	11	9	12	7	13	12	8	7
38	13	6	10	13	6	12	10	12	8	10	6	11	7	9	7	8	7	8	11	9
39	14	6	10	14	9	11	9	7	10	7	6	11	7	9	7	10	9	7	8	7
40	14	7	10	14	8	8	7	12	7	12	9	8	13	11	9	12	9	11	12	8
41	14	6	9	14	9	12	13	13	7	12	12	10	6	7	6	8	11	12	8	12
42	14	6	12	14	11	11	13	12	7	9	9	14	6	11	7	10	13	9	9	10
43	14	6	10	14	6	7	13	8	10	7	7	9	6	12	10	10	13	9	10	7
44	13	7	7	13	10	10	10	8	13	11	9	8	12	8	9	12	10	12	10	8
45	13	7	8	13	12	11	8	11	13	7	9	8	10	10	11	10	12	8	11	9
46	13	6	10	7	11	7	6	13	12	8	10	10	11	10	11	11	6	7	12	10
47	14	6	12	12	12	12	14	12	11	10	11	12	13	9	6	8	13	11	11	11
48	14	6	10	9	12	12	9	13	10	10	10	9	10	9	13	8	8	7	8	8
49	8	5	6	7	5	6	5	7	5	7	6	7	7	7	7	7	7	5	7	7
50	8	5	6	5	5	7	6	7	5	5	5	6	5	5	7	7	5	7	6	7
51	13	6	11	10	9	10	10	10	9	12	10	7	6	9	10	7	10	11	6	11
52	13	6	11	13	12	7	6	7	6	7	7	8	8	10	9	12	11	8	11	12
53	13	6	6	7	11	12	10	11	13	9	6	10	7	10	13	12	9	11	10	12
54	13	6	11	8	12	11	13	9	7	12	12	13	8	12	7	7	7	11	7	11
55	14	6	6	9	6	7	14	10	8	8	10	12	6	12	12	11	7	8	8	10
56	8	5	6	5	6	6	6	6	6	5	7	5	5	5	5	7	6	7	7	6
57	9	5	6	5	5	6	5	5	5	5	5	5	6	7	7	7	7	5	7	7
58	15	6	11	14	8	8	7	12	6	9	9	13	13	9	11	8	7	8	6	12
59	12	6	9	11	12	10	10	8	11	8	12	10	6	7	11	7	10	12	7	8
60	8	5	7	5	7	6	7	6	7	6	7	6	5	6	6	6	7	5	6	7
61	15	6	7	7	8	7	6	7	7	8	7	13	13	12	13	12	13	7	12	9
62	13	6	6	10	7	11	7	12	12	11	11	12	9	8	10	13	7	10	6	11
63	14	6	7	13	6	10	12	11	8	12	6	7	6	11	6	13	12	7	8	7
64	14	6	7	13	12	8	8	10	6	8	9	14	7	12	9	8	13	7	9	7
65	14	6	10	8	13	8	13	13	12	12	7	14	6	9	9	8	12	9	12	7
66	14	6	7	14	10	10	9	7	11	11	9	10	10	11	6	8	6	8	6	8
67	14	6	8	14	7	7	10	10	9	12	7	12	7	8	10	12	9	8	9	11
68	13	7	9	8	8	8	8	8	7	8	9	7	9	7	13	9	10	12	8	7

Table D18. Data regarding the duration for order 69 to 100 of P6S2

Order	M 1	M 2	M 3	M 4	M 5	M 6	M 7	M 8	M 9	M 10	M 11	M 12	M 13	M 14	M 15	M 16	M 17	M 18	M 19	M 20
69	13	6	11	8	13	12	12	7	6	9	6	7	13	11	6	11	13	8	6	12
70	13	6	11	9	11	12	9	9	10	9	9	11	7	7	6	9	13	8	9	10
71	13	7	7	9	13	8	10	11	11	10	7	9	11	11	11	11	8	10	10	12
72	13	7	10	12	10	12	12	12	9	9	8	10	13	11	11	10	11	9	8	11
73	13	6	8	12	13	9	6	13	6	12	8	8	9	10	7	7	6	10	7	8
74	15	7	7	8	10	12	13	10	8	7	11	8	13	11	13	8	7	11	9	12
75	14	6	10	9	7	8	6	7	9	8	7	14	9	8	6	10	11	7	12	11
76	12	6	10	12	12	10	9	11	9	8	11	10	12	8	11	9	6	12	12	7
77	13	7	9	13	11	11	8	10	10	7	9	12	7	8	7	8	8	9	9	7
78	15	7	11	8	13	7	12	13	7	7	7	7	11	12	10	9	12	11	10	10
79	14	7	11	9	7	12	14	7	8	7	8	8	10	11	9	8	8	11	9	11
80	13	6	9	9	7	7	6	11	7	7	6	11	7	9	13	7	10	9	12	8
81	11	7	7	7	8	9	7	11	8	10	10	11	10	8	10	7	11	7	10	7
82	13	6	6	8	9	10	6	7	7	8	8	10	8	8	12	8	13	10	9	11
83	14	6	6	9	10	8	8	9	8	11	7	10	9	8	14	13	12	9	11	12
84	15	6	12	14	6	7	11	7	13	7	8	10	9	7	8	13	11	9	7	10
85	13	6	12	12	12	10	6	8	13	12	6	10	10	11	10	8	13	7	12	11
86	13	6	10	8	9	8	10	7	8	7	12	9	10	9	12	13	6	11	7	7
87	14	6	7	11	10	8	14	9	12	12	7	10	7	10	8	8	6	10	9	8
88	14	6	10	12	7	7	14	11	8	12	10	9	6	9	7	11	7	12	12	11
89	14	6	8	9	10	11	9	11	8	10	6	14	10	9	12	9	11	7	7	12
90	14	6	12	12	8	8	9	10	11	12	6	10	10	10	8	11	7	8	7	11
91	14	6	7	8	6	7	6	12	9	11	9	13	12	11	14	13	9	10	7	8
92	13	6	6	13	8	10	9	11	11	8	8	10	12	7	8	9	6	9	12	12
93	12	6	6	12	9	9	6	11	7	9	6	10	8	8	8	9	7	10	6	9
94	9	5	7	6	7	6	5	6	6	6	5	5	7	6	7	6	5	5	5	7
95	8	5	6	6	5	7	7	7	7	5	5	5	6	6	7	7	7	7	7	7
96	13	6	12	8	6	12	6	8	7	11	6	11	12	11	13	8	12	8	6	12
97	15	6	6	14	11	10	14	7	8	10	8	7	11	9	14	7	12	8	12	12
98	13	6	7	13	9	10	12	11	6	10	10	11	6	10	6	10	10	11	9	10
99	13	6	6	12	6	7	7	11	11	7	8	7	8	8	9	10	8	9	6	9
100	14	6	8	14	11	8	9	7	10	9	11	9	6	11	7	9	7	11	12	8

Appendix E: Additional results of scheduling instances

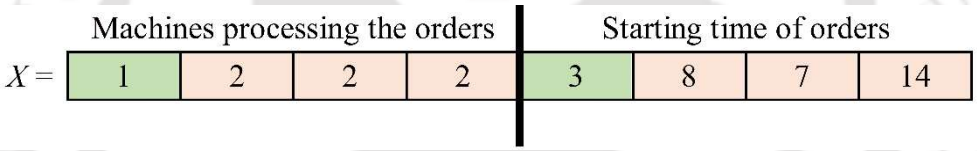
Numerical illustration of the no-wait time heuristic mechanism

Numerical examples of a scheduling problem with two machines and four jobs are provided to understand the proposed heuristic mechanism. The parameter values regarding the release date, due date, processing time, and cost of jobs on each machine are provided in Table E1.

Table E1 Parameter values required for the examples

Jobs	Release date	Due date	Processing time		LB	UB
			M_1	M_2		
J_1	2	10	7	8	2	3
J_2	2	15	6	5	2	10
J_3	3	13	8	4	3	9
J_4	13	26	9	6	13	20

Example 1*:* Let the potential solution provided by a metaheuristic technique be as given below.



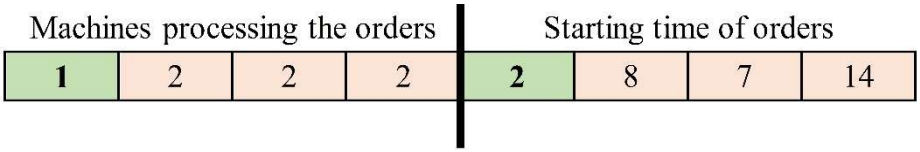
In the schedule, J_1 is assigned to M_1, and its starting time is 3. The other jobs, J_2, J_3, and J_4 are allotted to M_2, starting at 8, 7, and 14, respectively.

Consider machine 1 (M_1)

S1. As per schedule X, J_1 is processed by M_1. Hence $I_1 = \{1\}$

S2. Total number of orders in M_1, $K = |I_1| = 1$

S3. Since K = 1, the starting time of J_1 is replaced with its release date, which is equal to 2.



The modified solution after the completion of J_1 on M_1 would be as given above.

Consider machine 2 (M_2)

S1. As per schedule, J_2, J_3, and J_4 are is processed by M_2. Hence $I_2 = \{2,3,4\}$

S2. Total number of jobs in M_2, $K = |I_2| = 3$

S3. Sort the jobs assigned to M_2 in the increasing order of their starting time.

Jobs	J_3	J_2	J_4
Starting time	7	8	14

S4. After sorting, J_3 has to be processed first on M_2. Hence, the starting time of J_3 is replaced with its release date.

Jobs	J_3	J_2	J_4
Starting time	**3**	8	14

S5. Calculate the end time of J_3 using Eq.(6.1), which is equal to 7 (= 3 + 4). Determine the starting time of J_2 using Eq. (6.12).

Jobs	J_3	J_2	J_4
Starting time	3	**7**	14

S6. Determine the starting time of J_4 using Eq. (6.12), which is max(2, 7) = 7. The end time of J_2 is 12 (= 7 + 5), which is less than the release date of J_4 (= 13). Hence, the starting time of J_4 is set to its release date.

Jobs	J_3	J_2	J_4
Starting time	3	7	**13**

S7. Check for the upper bound violation at the starting time of all the jobs. In the modified schedule, none of the variables has violated its upper bound.

The repaired schedule after undergoing the no-wait time heuristic mechanism is provided

	Machines processing the orders				Starting time of orders			
X_r =	**1**	**2**	**2**	**2**	**2**	**7**	**3**	**13**

As per the example, the potential schedule provided by the metaheuristic technique violates the overlapping constraint as J_3 on M_2 starts at 7 and ends at 11, and subsequently, J_2 has to start at 8. However, the modified solution does not violate any of the overlapping constraints.

Example 2: In *Example* 1, the starting time of none of the orders has violated the upper bound. However, a scenario in which the upper bound violation can occur in a potential schedule like the one given below

	Machines processing the orders				Starting time of orders			
$X =$	2	1	1	1	3	8	7	14

Consider machine 1 (M_1)

S1. Jobs J_2, J_3, and J_4 are allotted to M_1. Hence $I_1 = \{2,3,4\}$.

S2. Total number of jobs in M_1, $K = |I_2| = 3$.

S3. Sort jobs assigned to M_1 in the increasing order of their starting time.

Jobs	J_3	J_2	J_4
Starting time	**7**	8	14

S4. Replace the starting time of the first order (J_3) with its release date (= 3) and calculate the end time of J_3, which is equal to 11 (= 3 + 8).

S5. Determine the starting time of J_2 using Eq. (6.12), and it is equal to 11 (= max(8, 11)). The end time of J_2 is equal to 17 (=11 + 6).

Jobs	J_3	J_2	J_4
Starting time	3	**11**	14

S6. Determine the starting time of J_4 using Eq.(6.12), which is equal to 17 (= max(14, 17)).

Jobs	J_3	J_2	J_4
Starting time	3	11	**17**

S7. Identify set of jobs assigned to M_1 whose starting time exceeds the upper bound, $I_{ub} = \{2\}$

S8. As $|I_{ub}| \geq 1$, replace the starting time of J_2 to its upper bound, i.e., equal to 10.

Jobs	J_3	J_2	J_4
Starting time	3	**10**	17

S9. Sort the jobs assigned to M_1 in the ascending order of their starting time

Jobs	J_3	J_2	J_4
Starting time	3	**10**	17

S10. Determine the end time of all jobs assigned to M_1 as per the updated starting time.

Jobs	J_3	J_2	J_4
End time	11	**16**	26

Consider machine 2 (M_2)

S11. As per schedule X, the first order J_1 is assigned to M_2. As it is the only order allotted to the M_2, replace the starting time of J_1 with its release date (= 2).

The repaired solution (X_r) of X is as given below.

X_r =

Machines processing the orders				Starting time of orders			
2	**1**	**1**	**1**	**2**	**10**	**3**	**17**

The determination of any constraint violation of solutions X and X_r is performed as below.

X		End Time	Violation		X_r		End Time	Violation	
M	J		Due date constraint	Overlap constraint	M	J		Due date constraint	Overlap constraint
1	3	7 + 8 = 15	15 − 13 = 2	0	1	3	3 + 8 = 11	0	0
1	2	8 + 6 = 14	0	15 – 8 = 7	1	2	10 + 6 = 16	16 – 15 = 1	11−10 = 1
1	4	14 + 9= 23	0	0	1	4	17 + 9 = 26	0	0
2	1	3 + 8 = 11	11 − 10 = 1	0	2	1	2 + 8 = 10	0	0
Total violation			3	7	Total violation			1	1

In *Example* 2, J_3 has to start at 7 as per the potential schedule X, and in that case, the due date constraint corresponding to this order gets violated. At the same time, for the repaired solution, the starting time of J_3 is replaced with its release date so that early processing of the first order

J_3 occurs and also satisfies its due date constraint. It can also be observed that if J_3 has started at 7, then its end time on M_1 would be 15, and the succeeding order, as per the potential schedule, has to begin its processing at 8, which violates the overlap constraint. In the case of the repaired schedule, the end time of J_3 is 11, which is higher than the upper bound of the starting time of subsequent order J_2. Hence, the starting time of J_2 is replaced with the upper bound of its starting time. This modification has resulted in the violation of overlap constraints in M_1 while processing J_2 and J_3. However, the overlap constraint violation is lesser in the repaired schedule than in the potential schedule provided by the metaheuristic technique. It is due to shifting the starting time to the release date of the first order in a machine, and the strategy followed in handling the bound violation. It can be observed from *Example* 2 that neither the schedule provided by the metaheuristic technique nor the repaired schedule is feasible. The total violations on the due date and the overlap constraints of schedule X are higher than X_r.

Table E2. Best fitness value of algorithms with the horizon (WH) and without the horizon constraints (WoH)

Problems	ABC		AWDO		COA		CSO		DNLPSO		GWO		MFO		MPEDE	
	WoH	WH	WoH	WH	WoH	WH	WoH	WH	WoH	WH	WoH	WH	WoH	WH	WoH	WH
P1S1	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**
P1S2	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**
P2S1	**60**	**60**	**130**	131	**60**	**60**	**60**	**60**	**60**	**60**	**60**	**60**	**60**	**60**	**60**	**60**
P2S2	**44**	**44**	**44**	**44**	**44**	**44**	**44**	**44**	**44**	**44**	**44**	**44**	**44**	**44**	**44**	**44**
P3S1	**101**	**101**	**234**	236	**101**	**101**	**101**	**101**	**101**	102	**101**	**101**	**101**	102	**101**	**101**
P3S2	**83**	**83**	**84**	85	**83**	**83**	**83**	**83**	**83**	**83**	**83**	**83**	**83**	**83**	**83**	**83**
P4S1	**116**	117	**117**	119	**115**	**115**	**115**	**115**	120	**119**	**115**	116	**118**	119	**115**	**115**
P4S2	**103**	**103**	**103**	**103**	**102**	**102**	**102**	**102**	**104**	**104**	**102**	**102**	**103**	**103**	**102**	**102**
P5S1	**161**	**161**	**164**	167	**159**	**159**	160	**159**	169	**168**	161	**160**	**165**	167	**159**	**159**
P5S2	142	**141**	**141**	**141**	**140**	**140**	**140**	**140**	**145**	**145**	**140**	**140**	**141**	142	**140**	**140**

Problems	MVO		SHO		SHTS		SOS		SSA		sTLBO		YYPO	
	WoH	WH	WoH	WH	WoH	WH	WoH	WH	WoH	WH	WoH	WH	WoH	WH
P1S1	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**
P1S2	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**
P2S1	**60**	**60**	**60**	**60**	**60**	**60**	**60**	**60**	**60**	**60**	**60**	**60**	**60**	**60**
P2S2	**44**	**44**	**44**	**44**	**44**	**44**	**44**	**44**	**44**	**44**	**44**	**44**	**44**	**44**
P3S1	**101**	**101**	**233**	237	**101**	**101**	**101**	**101**	**101**	**101**	**101**	**101**	**101**	**101**
P3S2	**83**	**83**	**84**	**84**	**83**	**83**	**83**	**83**	**83**	**83**	**83**	**83**	**83**	**83**
P4S1	**116**	117	**124**	**124**	**116**	117	**115**	**115**	**116**	117	**115**	**115**	**116**	**116**
P4S2	103	**102**	**109**	**109**	**102**	**102**	**102**	**102**	**102**	**102**	**102**	**102**	**103**	**103**
P5S1	**159**	160	**388**	389	161	**160**	**158**	159	164	**163**	159	**158**	**161**	162
P5S2	**141**	**141**	**150**	**150**	**140**	**140**	**140**	**140**	**142**	**142**	**140**	**140**	142	**141**

Table E3. Worst fitness value of algorithms with the horizon (WH) and without the horizon constraints (WoH)

Problems	ABC		AWDO		COA		CSO		DNLPSO		GWO		MFO		MPEDE	
	WoH	WH	WoH	WH	WoH	WH	WoH	WH	WoH	WH	WoH	WH	WoH	WH	WoH	WH
P1S1	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**58**	**58**	**26**	**26**	58	**26**	**26**	58
P1S2	**18**	**18**	**21**	**21**	**18**	**18**	**18**	**18**	**21**	**21**	**18**	**18**	**18**	**18**	**18**	**18**
P2S1	**60**	**60**	**138**	143	**130**	131	**60**	132	133	**132**	132	**130**	**131**	132	**130**	**130**
P2S2	**44**	**44**	**48**	**48**	**44**	**44**	**45**	**45**	**46**	**46**	**45**	**45**	**45**	**45**	**44**	**44**
P3S1	**104**	105	**247**	250	108	**104**	107	**105**	**238**	**238**	**105**	109	232	**231**	105	**104**
P3S2	**85**	**85**	**89**	**89**	84	**83**	**84**	**84**	**87**	**87**	**84**	**84**	86	**85**	**84**	**84**
P4S1	**120**	**120**	131	**130**	**118**	119	123	**122**	127	**126**	**120**	**120**	**125**	126	**118**	**118**
P4S2	**105**	**105**	108	**107**	**102**	103	**104**	**104**	**113**	**113**	**104**	**104**	**106**	107	104	**103**
P5S1	**166**	168	387	**386**	**164**	165	**170**	171	**388**	392	**169**	170	379	**184**	**164**	**164**
P5S2	**145**	146	**151**	**151**	**141**	**141**	**142**	**142**	**155**	**155**	**145**	**145**	150	**147**	**141**	**141**

Problems	MVO		SHO		SHTS		SOS		SSA		sTLBO		YYPO	
	WoH	WH	WoH	WH	WoH	WH	WoH	WH	WoH	WH	WoH	WH	WoH	WH
P1S1	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**
P1S2	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**
P2S1	**130**	133	**139**	141	133	**132**	**60**	**60**	**130**	131	**130**	133	**60**	**60**
P2S2	**45**	**45**	**47**	**47**	**44**	**44**	**44**	**44**	**45**	**45**	**45**	**45**	**44**	**44**
P3S1	232	**108**	**246**	251	**104**	105	**103**	**103**	235	**110**	**103**	105	**107**	228
P3S2	85	**84**	**90**	**90**	**84**	**84**	**84**	**84**	**85**	**85**	**84**	**84**	**84**	**84**
P4S1	**123**	**123**	**290**	**290**	**121**	122	**117**	**117**	**124**	**124**	118	**117**	**121**	122
P4S2	**108**	109	**115**	**115**	**105**	**105**	**104**	**104**	**108**	**108**	104	**103**	**105**	**105**
P5S1	**171**	**171**	**403**	404	**173**	176	**163**	164	**378**	**378**	166	**164**	172	**171**
P5S2	150	**147**	**159**	**159**	**144**	**144**	142	**141**	**148**	**148**	**141**	**141**	**145**	**145**

Table E4. Median of the fitness value of all algorithms with the horizon (WH) and without the horizon constraints (WoH)

Problems	ABC		AWDO		COA		CSO		DNLPSO		GWO		MFO		MPEDE	
	WoH	WH	WoH	WH	WoH	WH	WoH	WH	WoH	WH	WoH	WH	WoH	WH	WoH	WH
P1S1	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**
P1S2	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**
P2S1	**60**	**60**	**134**	134.5	**60**	**60**	**60**	**60**	**60**	**60**	**60**	**60**	**60**	**60**	**60**	**60**
P2S2	**44**	**44**	**46**	**46**	**44**	**44**	**44**	**44**	**44**	**44**	**44**	**44**	**44**	**44**	**44**	**44**
P3S1	**103**	**103**	**242**	244.5	**102**	102.5	**103**	**103**	108	**107.5**	**103**	**103**	**105**	**105**	**101**	102
P3S2	**84**	**84**	**87**	**87**	**83**	**83**	**83**	**83**	**84**	**84**	**84**	**84**	**84**	**84**	**83**	**83**
P4S1	**118**	**118**	**123.5**	125	**116**	117	**118**	119	122.5	**122**	119	**118**	**122**	**122**	**116**	**116**
P4S2	**104**	**104**	**104**	105	**102**	**102**	**102**	**102**	**106**	**106**	**103**	**103**	**105**	**105**	**102**	**102**
P5S1	**164**	165	**171**	173	**160**	161	**164**	**164**	**174.5**	176	164	**163**	171	**170.5**	**160**	161
P5S2	**144**	**144**	**145**	146	**140**	**140**	**141**	**141**	**146.5**	147	**142**	**142**	144	**143.5**	**140**	**140**

Problems	MVO		SHO		SHTS		SOS		SSA		sTLBO		YYPO	
	WoH	WH	WoH	WH	WoH	WH	WoH	WH	WoH	WH	WoH	WH	WoH	WH
P1S1	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**
P1S2	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**
P2S1	**60**	95	**133.5**	134	**60**	**60**	**60**	**60**	**60**	**60**	**60**	**60**	**60**	**60**
P2S2	**44**	**44**	**45**	**45**	**44**	**44**	**44**	**44**	**44**	**44**	**44**	**44**	**44**	**44**
P3S1	**103**	**103**	**241**	241.5	**102**	103	102	**101**	105.5	**104**	**101**	102	**103**	**103**
P3S2	84	**83**	87.5	**87**	**83**	**83**	**83**	**83**	**84**	**84**	**83**	**83**	**84**	**84**
P4S1	120	**119**	**133.5**	**133.5**	**118**	120	**115**	**115**	**120.5**	121	**115**	115.5	119	**118.5**
P4S2	**105**	**105**	**112.5**	**112.5**	**103**	**103**	**102**	**102**	**104**	**104**	**102**	**102**	**104**	**104**
P5S1	**165**	166	**396**	**396**	**165**	167.5	**160**	161	**168**	168.5	**160**	**160**	**166**	167
P5S2	143.5	**143**	**155**	**155**	**141.5**	142	**141**	**141**	**144**	**144**	141	**140**	**144**	**144**

Table E5. Standard deviation of the fitness value of all algorithms with the horizon (WH) and without the horizon constraints (WoH)

Problems	ABC		AWDO		COA		CSO		DNLPSO		GWO		MFO		MPEDE	
	WoH	WH	WoH	WH	WoH	WH	WoH	WH	WoH	WH	WoH	WH	WoH	WH	WoH	WH
P1S1	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**5.84**	**5.84**	**0.00**	**0.00**	5.84	**0.00**	**0.00**	5.84
P1S2	**0.00**	**0.00**	**0.76**	**0.76**	**0.00**	**0.00**	**0.00**	**0.00**	**0.55**	**0.55**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
P2S1	**0.00**	**0.00**	**2.19**	2.77	**24.20**	28.55	**0.00**	13.15	29.10	**28.86**	26.69	**24.20**	33.59	**30.31**	30.11	**21.36**
P2S2	**0.00**	**0.00**	**1.02**	**1.02**	**0.00**	**0.00**	**0.31**	**0.31**	**0.46**	**0.46**	**0.31**	**0.31**	**0.25**	**0.25**	**0.00**	**0.00**
P3S1	**0.82**	1.16	**3.45**	4.06	1.43	**0.68**	1.53	**1.19**	61.17	**55.41**	**0.94**	1.82	43.43	**43.29**	1.01	**1.00**
P3S2	0.52	**0.43**	1.15	**1.10**	0.38	**0.00**	**0.50**	0.51	**1.19**	1.20	0.45	**0.43**	0.68	**0.58**	**0.31**	**0.31**
P4S1	1.04	**0.82**	2.96	**2.79**	**1.04**	1.35	**1.91**	1.94	**1.82**	1.89	1.41	**1.29**	**1.63**	1.89	**0.92**	0.92
P4S2	**0.76**	0.76	1.41	**1.30**	**0.00**	0.18	**0.55**	0.56	2.19	**2.16**	**0.61**	0.66	**1.02**	1.06	0.48	**0.38**
P5S1	**1.26**	1.36	80.94	**80.22**	**1.16**	1.74	**2.56**	2.64	**39.12**	97.51	**1.92**	2.23	38.15	**3.76**	**1.23**	1.33
P5S2	**0.97**	1.30	2.49	**2.27**	0.45	**0.38**	0.61	**0.45**	2.45	**2.40**	**1.09**	1.14	1.88	**1.60**	**0.45**	0.50

Problems	MVO		SHO		SHTS		SOS		SSA		sTLBO		YYPO	
	WoH	WH	WoH	WH	WoH	WH	WoH	WH	WoH	WH	WoH	WH	WoH	WH
P1S1	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
P1S2	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
P2S1	**35.28**	36.08	**22.64**	25.95	**24.47**	26.69	**0.00**	**0.00**	32.63	**26.61**	24.20	**18.39**	**0.00**	**0.00**
P2S2	**0.25**	**0.25**	**0.94**	**0.94**	**0.00**	**0.00**	**0.00**	**0.00**	**0.35**	**0.35**	**0.31**	**0.31**	**0.00**	**0.00**
P3S1	23.53	**1.68**	**2.60**	3.58	**0.90**	0.98	**0.67**	0.84	47.85	**2.53**	**0.76**	1.07	**1.53**	22.87
P3S2	**0.41**	0.50	1.53	**1.49**	**0.50**	0.51	0.25	**0.18**	**0.61**	0.63	**0.48**	0.49	0.51	**0.50**
P4S1	2.22	**1.79**	74.47	**69.54**	**1.40**	1.58	0.73	**0.63**	2.02	**1.88**	0.95	**0.63**	1.30	**1.20**
P4S2	**1.30**	1.55	**1.92**	**1.92**	0.72	**0.66**	0.57	**0.56**	1.34	**1.31**	0.55	**0.45**	**0.76**	**0.76**
P5S1	**2.22**	2.91	**3.94**	4.36	**3.03**	3.53	**1.28**	1.30	38.43	**38.42**	1.41	**1.36**	2.65	**2.57**
P5S2	2.07	**1.84**	**2.07**	**2.07**	**1.12**	1.14	0.56	**0.50**	**1.39**	1.47	**0.47**	0.51	**1.04**	1.10

Table E6. Worst objective function values determined by algorithms with repair (WR) and without repair (WoR) using the heuristic technique

Problems	ABC		AWDO		COA		CSO		DNLPSO		GWO		MFO		MPEDE	
	WoR	WR	WoR	WR	WoR	WR	WoR	WR	WoR	WR	WoR	WR	WoR	WR	WoR	WR
P1S1	**26**	**26**	60	**26**	58	**26**	**26**	**26**	59	**58**	58	**26**	**58**	**58**	58	**26**
P1S2	**18**	**18**	25	**21**	21	**18**	21	**18**	**21**	**21**	**18**	**18**	22	**18**	22	**18**
P2S1	131	**60**	143	**138**	133	**130**	135	**60**	141	**133**	136	**132**	134	**131**	133	**130**
P2S2	46	**44**	53	**48**	46	**44**	49	**45**	53	**46**	48	**45**	52	**45**	46	**44**
P3S1	236	**104**	255	**247**	235	**108**	239	**107**	257	**238**	235	**105**	241	**232**	235	**105**
P3S2	88	**85**	100	**89**	88	**84**	93	**84**	104	**87**	93	**84**	215	**86**	87	**84**
P4S1	125	**120**	295	**131**	125	**118**	286	**123**	299	**127**	278	**120**	289	**125**	122	**118**
P4S2	110	**105**	132	**108**	109	**102**	113	**104**	123	**113**	115	**104**	121	**106**	108	**104**
P5S1	382	**166**	414	**387**	378	**164**	393	**170**	424	**388**	389	**169**	396	**379**	379	**164**
P5S2	151	**145**	374	**151**	148	**141**	357	**142**	373	**155**	171	**145**	360	**150**	146	**141**

Problems	MVO		SHO		SHTS		SOS		SSA		sTLBO		YYPO	
	WoR	WR	WoR	WR	WoR	WR	WoR	WR	WoR	WR	WoR	WR	WoR	WR
P1S1	58	**26**	58	**26**	58	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**
P1S2	21	**18**	22	**18**	**18**	**18**	**18**	**18**	21	**18**	21	**18**	**18**	**18**
P2S1	137	**130**	149	**139**	134	**133**	133	**60**	136	**130**	132	**130**	133	**60**
P2S2	48	**45**	60	**47**	48	**44**	46	**44**	51	**45**	47	**45**	47	**44**
P3S1	243	**232**	268	**246**	240	**104**	236	**103**	244	**235**	235	**103**	244	**107**
P3S2	95	**85**	226	**90**	96	**84**	87	**84**	101	**85**	89	**84**	91	**84**
P4S1	283	**123**	317	**290**	292	**121**	124	**117**	290	**124**	276	**118**	284	**121**
P4S2	118	**108**	279	**115**	117	**105**	109	**104**	122	**108**	112	**104**	114	**105**
P5S1	393	**171**	436	**403**	396	**173**	377	**163**	401	**378**	382	**166**	398	**172**
P5S2	165	**150**	381	**159**	168	**144**	151	**142**	176	**148**	154	**141**	164	**145**

Table E7. Mean objective function values determined by algorithms with repair (WR) and without repair (WoR) using the heuristic technique

Problems	ABC		AWDO		COA		CSO		DNLPSO		GWO		MFO		MPEDE	
	WoR	WR	WoR	WR	WoR	WR	WoR	WR	WoR	WR	WoR	WR	WoR	WR	WoR	WR
P1S1	**26.00**	**26.00**	38.87	**26.00**	32.40	**26.00**	**26.00**	**26.00**	38.70	**27.07**	27.07	**26.00**	29.20	**27.07**	28.13	**26.00**
P1S2	**18.00**	**18.00**	20.33	**18.20**	18.40	**18.00**	18.40	**18.00**	19.00	**18.10**	**18.00**	**18.00**	20.83	**18.00**	20.53	**18.00**
P2S1	88.30	**60.00**	137.10	**133.97**	116.63	**69.33**	130.87	**60.00**	132.53	**74.47**	131.27	**71.73**	129.17	**83.40**	128.17	**76.33**
P2S2	45.17	**44.00**	48.37	**45.70**	45.37	**44.00**	46.40	**44.10**	47.40	**44.17**	45.60	**44.10**	48.77	**44.07**	45.43	**44.00**
P3S1	228.57	**102.53**	247.27	**241.27**	223.57	**102.50**	234.53	**103.13**	248.00	**148.03**	224.03	**102.87**	236.20	**121.80**	188.07	**101.73**
P3S2	86.40	**84.07**	94.47	**87.17**	86.23	**83.17**	88.63	**83.43**	92.70	**84.20**	90.20	**83.73**	98.50	**84.13**	85.10	**83.10**
P4S1	123.03	**118.13**	248.73	**123.80**	122.00	**116.43**	201.47	**118.47**	292.90	**122.70**	148.47	**118.27**	257.43	**121.97**	119.87	**115.80**
P4S2	108.73	**104.10**	120.80	**104.57**	105.50	**102.00**	109.73	**102.33**	117.07	**106.40**	112.40	**103.20**	116.70	**104.70**	105.10	**102.20**
P5S1	337.50	**164.07**	394.80	**206.23**	257.70	**160.37**	375.60	**164.47**	410.83	**181.57**	379.67	**163.97**	388.43	**177.87**	264.40	**159.93**
P5S2	148.87	**143.53**	213.40	**145.53**	144.57	**140.27**	169.80	**141.03**	178.67	**147.23**	158.47	**141.90**	188.03	**144.17**	143.83	**140.27**

Problems	MVO		SHO		SHTS		SOS		SSA		sTLBO		YYPO	
	WoR	WR	WoR	WR	WoR	WR	WoR	WR	WoR	WR	WoR	WR	WoR	WR
P1S1	27.07	**26.00**	35.43	**26.00**	27.07	**26.00**	**26.00**	**26.00**	**26.00**	**26.00**	**26.00**	**26.00**	**26.00**	**26.00**
P1S2	18.10	**18.00**	21.07	**18.00**	**18.00**	**18.00**	**18.00**	**18.00**	18.80	**18.00**	20.80	**18.00**	**18.00**	**18.00**
P2S1	129.53	**90.33**	143.77	**126.60**	129.00	**69.43**	127.83	**60.00**	131.87	**81.00**	120.90	**69.33**	130.63	**60.00**
P2S2	46.37	**44.07**	53.67	**45.13**	46.00	**44.00**	45.10	**44.00**	47.93	**44.13**	45.60	**44.10**	46.03	**44.00**
P3S1	236.00	**107.87**	262.83	**240.53**	235.83	**102.13**	222.87	**101.63**	237.73	**126.13**	223.07	**101.63**	237.93	**102.87**
P3S2	91.50	**83.97**	196.17	**87.30**	90.87	**83.40**	85.60	**83.07**	95.97	**84.20**	85.97	**83.33**	89.17	**83.53**
P4S1	197.17	**119.67**	310.83	**182.47**	234.83	**118.33**	120.93	**115.53**	220.97	**120.33**	126.90	**115.70**	224.20	**118.60**
P4S2	113.53	**104.77**	207.80	**112.40**	113.10	**103.03**	105.97	**102.43**	116.73	**104.27**	107.00	**102.33**	111.23	**103.90**
P5S1	376.60	**164.77**	429.43	**395.70**	387.10	**165.63**	242.63	**160.13**	390.67	**175.27**	293.97	**160.43**	385.03	**166.23**
P5S2	159.60	**144.00**	377.87	**155.30**	157.43	**141.67**	147.80	**140.63**	166.70	**144.17**	144.63	**140.70**	154.43	**143.53**

Table E8. Median of objective function values determined by algorithms with repair (WR) and without repair (WoR) using heuristic technique

Problems	ABC		AWDO		COA		CSO		DNLPSO		GWO		MFO		MPEDE	
	WoR	WR	WoR	WR	WoR	WR	WoR	WR	WoR	WR	WoR	WR	WoR	WR	WoR	WR
P1S1	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**
P1S2	**18**	**18**	21	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	**18**	21	**18**	21	**18**
P2S1	61	**60**	137	**134**	130	**60**	131	**60**	134	**60**	131	**60**	131	**60**	130	**60**
P2S2	45	**44**	48	**46**	45.5	**44**	46	**44**	47	**44**	46	**44**	49	**44**	46	**44**
P3S1	233	**103**	248	**242**	232	**102**	234	**103**	248	**108**	233	**103**	236	**105**	229	**101**
P3S2	86.5	**84**	94	**87**	86	**83**	89	**83**	92	**84**	91	**84**	94	**84**	85	**83**
P4S1	123	**118**	284.5	**123.5**	122	**116**	202	**118**	293	**122.5**	123	**119**	282	**122**	120	**116**
P4S2	109	**104**	120	**104**	106	**102**	109.5	**102**	117	**106**	113	**103**	117	**105**	105	**102**
P5S1	378	**164**	394	**171**	171.5	**160**	382	**164**	410	**174.5**	379	**164**	389	**171**	173	**160**
P5S2	149	**144**	167.5	**145**	144	**140**	149	**141**	166.5	**146.5**	158.5	**142**	163	**144**	144	**140**

Problems	MVO		SHO		SHTS		SOS		SSA		sTLBO		YYPO	
	WoR	WR	WoR	WR	WoR	WR	WoR	WR	WoR	WR	WoR	WR	WoR	WR
P1S1	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**26**
P1S2	**18**	**18**	21	**18**	**18**	**18**	**18**	**18**	**18**	**18**	21	**18**	**18**	**18**
P2S1	132	**60**	143.5	**133.5**	131	**60**	130	**60**	131	**60**	130	**60**	130	**60**
P2S2	46	**44**	53	**45**	46	**44**	45	**44**	48	**44**	46	**44**	46	**44**
P3S1	235.5	**103**	263	**241**	235.5	**102**	231	**102**	238	**105.5**	232	**101**	238	**103**
P3S2	91	**84**	222	**87.5**	91	**83**	85.5	**83**	95	**84**	85.5	**83**	89	**84**
P4S1	130.5	**120**	311	**133.5**	279	**118**	121	**115**	281.5	**120.5**	122	**115**	279	**119**
P4S2	113	**105**	273.5	**112.5**	113	**103**	106	**102**	117	**104**	107	**102**	111.5	**104**
P5S1	383	**165**	429	**396**	386	**165**	168	**160**	390.5	**168**	375	**160**	384	**166**
P5S2	159.5	**143.5**	378	**155**	158	**141.5**	148	**141**	166.5	**144**	144	**141**	154	**144**

Gantt chart of the optimal schedule

Gantt chart helps to visualize the distribution of orders across the available machines. The number of horizontal bars present in the Gantt chart would be equal to the total number of orders to be completed in the problem. SOS is the only algorithm that has determined multiple realizations of the optimal cost in all problems. Hence, in this section, the Gantt chart corresponding to one of the optimal schedules determined by SOS on each problem is presented in 1.

The first row of plots in 1 indicates the Gantt chart of the scheduling problems with data set one, and the second row includes the Gantt chart of problems with data set two. The major difference between these two datasets is the requirement of a longer processing time of orders in the first dataset than the second dataset that is visible in the Gantt chart. Moreover, due to the larger processing time, not all the orders need to be assigned to the machine with the least processing cost. Hence, the problem with dataset one has utilized all the available machines to complete the processing of all the orders. In contrast, at least one machine has remained unutilized in the case of problems with the second dataset. As one machine is not used in problems with the second dataset, an interested user can identify any other feasible solution (may not be optima) by reallotting some of the orders to the unutilized machines. At the same time, it is not possible in those problems with the first dataset. It is evident from the Gantt chart that none of the orders are allotted to machine one in the problems with the second dataset. Such a scenario has arisen because of the higher processing cost of machine one and the lesser processing time requirement of orders in the second dataset. It is possible to determine the best schedule identified in all problems with the second dataset, even without considering machine one. However, if any of the assigned machines become unusable, the inclusion of machine one might provide a better suboptimal schedule.

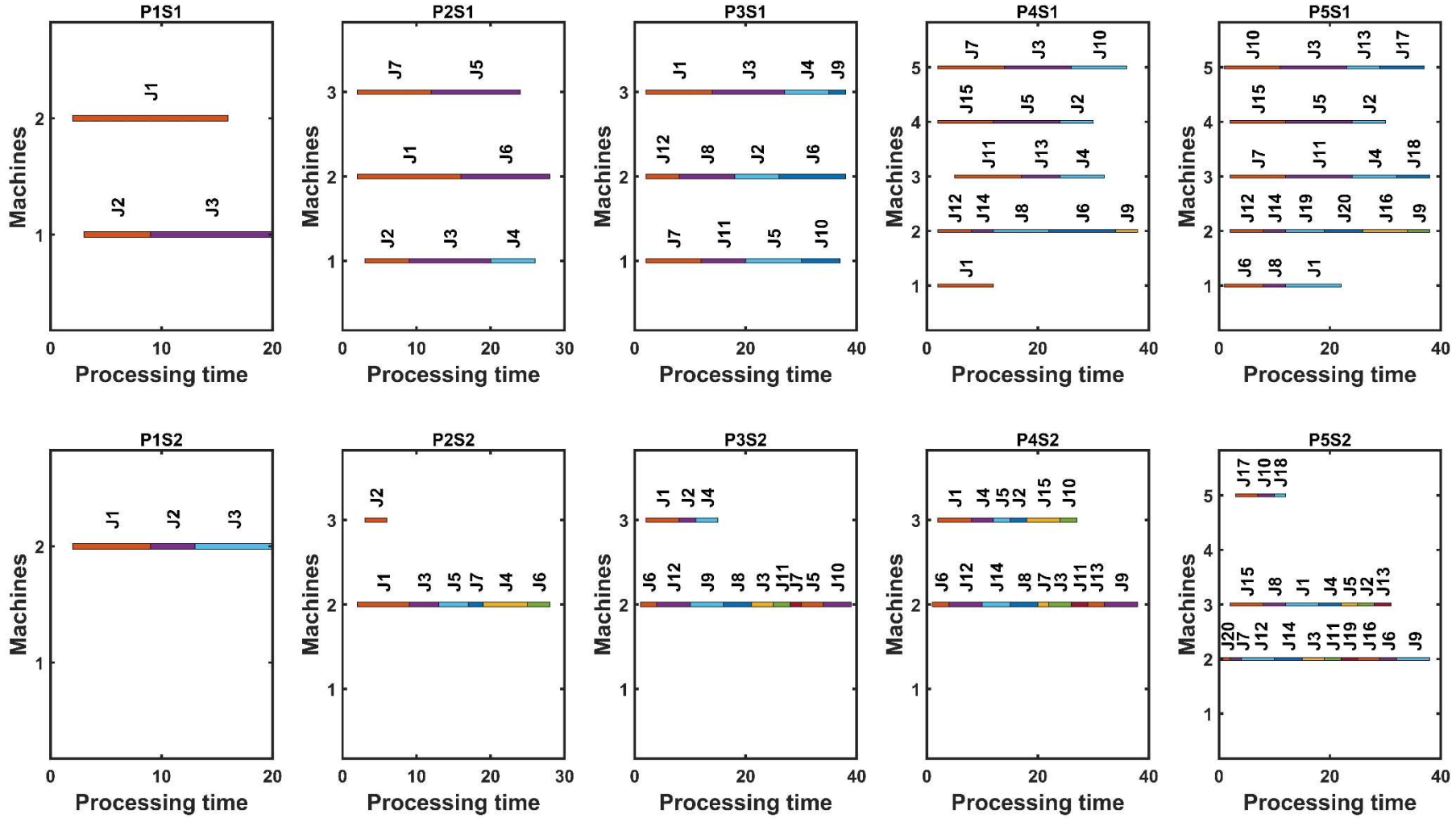


Fig. E1. Gantt chart of the optimal solution of all problem

Table E9. Number of realizations corresponding to each problem

Problem	ABC		AWDO		COA		CSO		DNLPSO	
	WoR	WR	WoR	WR	WoR	WR	WoR	WR	WoR	WR
P1S1	**3**	1	**3**	1	**3**	1	**3**	1	**3**	1
P1S2	**3**	2	**3**	2	**3**	2	**3**	2	**3**	2
P2S1	**8**	2	0	0	**6**	2	0	**2**	1	**2**
P2S2	7	**16**	0	**4**	4	**14**	1	**20**	0	**14**
P3S1	0	**3**	0	0	0	**6**	0	**2**	0	**1**
P3S2	0	**3**	0	0	0	**25**	0	**17**	0	**8**
P4S1	0	0	0	0	0	**6**	0	**2**	0	0
P4S2	0	0	0	0	0	**30**	0	**21**	0	0
P5S1	0	0	0	0	0	0	0	0	0	0
P5S2	0	0	0	0	0	**22**	0	**5**	0	0
Problem	**GWO**		**MFO**		**MPEDE**		**MVO**		**SHO**	
	WoR	WR	WoR	WR	WoR	WR	WoR	WR	WoR	WR
P1S1	**3**	1	**3**	1	**3**	1	**3**	1	**3**	1
P1S2	**3**	2	**2**	**2**	**3**	2	**3**	2	0	**1**
P2S1	0	**2**	1	**2**	1	**2**	1	**2**	0	**1**
P2S2	3	**16**	0	**16**	3	**14**	0	**18**	0	**8**
P3S1	0	**1**	0	**1**	0	**9**	0	**3**	0	0
P3S2	0	**8**	0	**4**	0	**26**	0	**3**	0	0
P4S1	0	**1**	0	0	0	**14**	0	0	0	0
P4S2	0	**3**	0	0	0	**25**	0	0	0	0
P5S1	0	0	0	0	0	0	0	0	0	0
P5S2	0	**2**	0	0	0	**22**	0	0	0	0
Problem	**SHTS**		**SOS**		**SSA**		**sTLBO**		**YYPO**	
	WoR	WR	WoR	WR	WoR	WR	WoR	WR	WoR	WR
P1S1	**3**	1	**3**	1	**2**	1	**3**	1	**3**	1
P1S2	**3**	2	**3**	2	**3**	2	**2**	**2**	**3**	2
P2S1	1	**2**	1	**2**	0	**2**	**4**	2	0	**2**
P2S2	1	**20**	8	**19**	0	**16**	2	**11**	0	**20**
P3S1	0	**6**	0	**8**	0	**1**	0	**8**	0	**7**
P3S2	0	**18**	1	**28**	0	**3**	0	**18**	0	**14**
P4S1	0	0	0	**18**	0	0	0	**17**	0	0
P4S2	0	**6**	0	**18**	0	**1**	0	**21**	0	0
P5S1	0	0	0	**2**	0	0	0	0	0	0
P5S2	0	**4**	0	**12**	0	0	0	**9**	0	0

The fitness of the final population obtained for solving P5S1 by SOS with and without the help of the heuristic mechanism is provided in Fig. E2. It is observed that all the solutions in the

final population of SOS with the heuristic mechanism are feasible solutions with the same fitness value. However, without using the heuristic mechanism, the final population obtained by SOS contains infeasible solutions. The fitness value of the best solution determined using SOS with the help of the heuristic mechanism is better than the value determined by SOS.

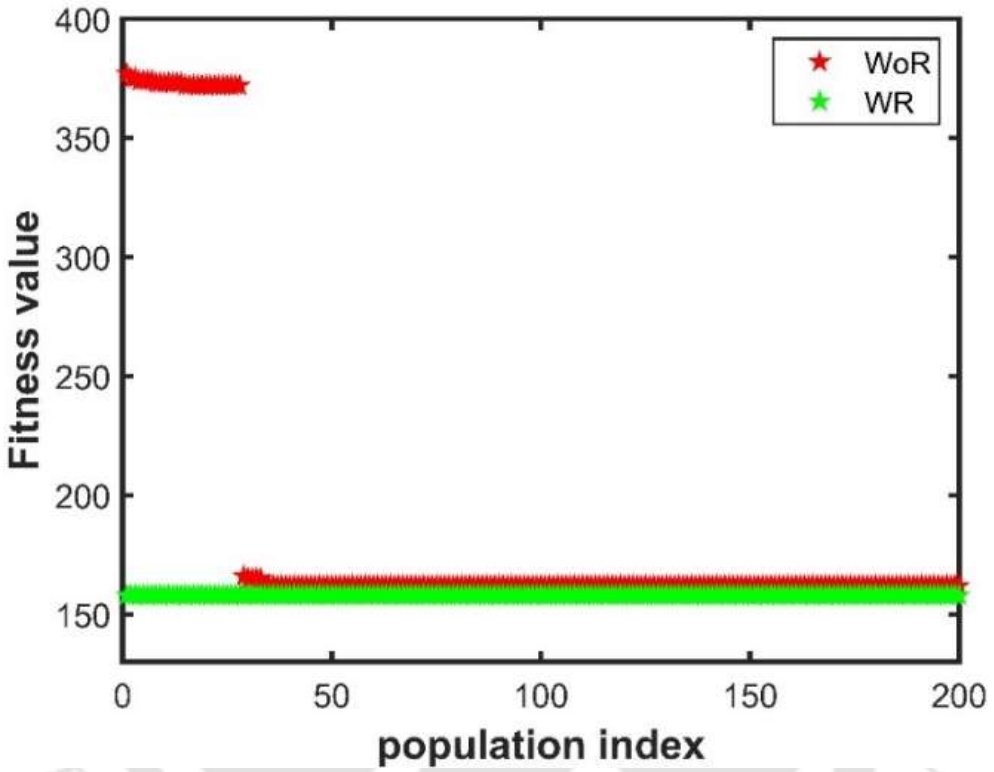


Fig. E2 Fitness value of the final population of SOS for solving P5S1

The performance of the proposed no-wait time heuristic mechanism is analyzed by solving a high-dimensional problem with 100 jobs and 20 machines. Two problem instances (P6S1 and P6S2) vary in the processing time required on the machines for each job is solved using 15 metaheuristic techniques with and without incorporating the proposed heuristic mechanism. The statistical analysis of the results obtained by all the techniques for solving P6S1 and P6S2 is provided in Table E10.

Performance analysis of proposed mechanism on the high-dimensional instances

The effectiveness of the proposed mechanism in solving high-dimensional instances is analyzed by solving two problem instances (P6S1 and P6S2) with 100 jobs and 20 machines. All the metaheuristic techniques were able to solve the high-dimension problem instances, where YYPO determined the best schedule in P6S1 and COA in P6S2. The convergence of the best fitness value with the function evaluation of the best five techniques with the proposed mechanism for P6S1 and P6S2 is given in Fig. E3.

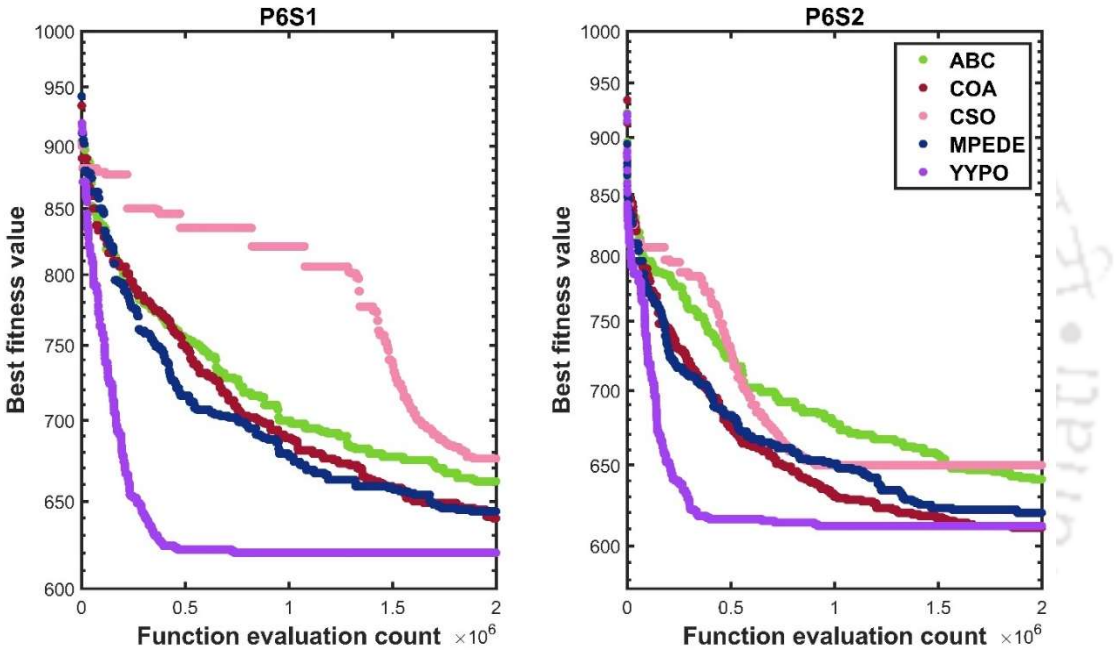


Fig. E3 Convergence of best fitness value for high-dimensional problems

YYPO has provided a quick convergence in both the problem instances and determined the best fitness value in P6S1. In P6S1, all techniques except YYPO are not able to converge even after utilizing the maximum allowed functional evaluations. YYPO and CSO, in P6S2, have converged to their final value by using approximately the same number of function evaluations. However, the best fitness value determined by YYPO is better than CSO. The statistical results and the Gantt charts corresponding to the best schedule obtained by solving the high dimensional problems (P6S1 and P6S2) are provided in Table E10, Fig E4, and Fig. E5.

Table E10. Statistical analysis of results obtained for solving P6S1 and P6S2

Techniques	P6S1									
	Best		Worst		Mean		Median		SD	
	WoR	WR	WoR	WR	WoR	WR	WoR	WR	WoR	WR
ABC	761.00	**662.00**	2035.00	**674.00**	1318.87	**668.77**	797.00	**669.00**	631.15	**2.93***
AWDO	2309.00	**863.00**	2357.00	**940.00**	2339.37	**900.27**	2341.50	**902.00**	**11.01**	16.69
COA	686.00	**640.00**	2007.00	**650.00**	1439.53	**645.90**	1985.00	**646.50**	647.78	**3.22**
CSO	679.00	**676.00**	2068.00	**835.00**	999.43	**738.37**	734.50	**705.00**	539.19	**61.48**
DNLPSO	**816.00**	**816.00**	2302.00	**879.00**	1142.23	**855.13**	862.50	**858.50**	585.15	**14.84**
GWO	**727.00**	**727.00**	2127.00	**889.00**	1031.17	**766.20**	761.00	**756.00**	547.67	**35.76**
MFO	2110.00	**732.00**	2201.00	**815.00**	2150.13	**771.47**	2144.00	**770.50**	22.97	**19.18**
MPEDE	694.00	**644.00**	2032.00	**668.00**	1882.17	**652.50**	2012.50	**652.00**	394.87	**4.85**
MVO	757.00	**696.00**	2115.00	**746.00**	1859.93	**715.87**	2071.50	**713.50**	490.61	**13.56**
SHO	2326.00	**886.00**	2359.00	**2186.00**	2344.07	**1176.70**	2344.50	**925.00**	**9.20**	510.22
SHTS	2126.00	**822.00**	2308.00	**883.00**	2217.97	**862.67**	2201.50	**863.50**	62.46	**11.08**
SOS	2138.00	**759.00**	2181.00	**780.00**	2163.43	**770.20**	2165.00	**771.00**	10.86	**5.52**
SSA	2071.00	**726.00**	2137.00	**790.00**	2112.27	**754.70**	2117.50	**754.00**	18.47	**13.86**
sTLBO	**688.00**	771.00	2247.00	**854.00**	1651.53	**820.90**	2004.50	**824.50**	634.85	**17.14**
YYPO	660.00	**620.00***	2123.00	**635.00***	722.57	**626.53***	672.00	**626.00***	264.64	**3.43**
Techniques	**P6S2**									
	Best		**Worst**		**Mean**		**Median**		**SD**	
	WoR	WR	WoR	WR	WoR	WR	WoR	WR	WoR	WR
ABC	657.00	**641.00**	678.00	**651.00**	671.40	**645.97**	672.00	**646.00**	4.44	**2.58**
AWDO	769.00	**711.00**	2215.00	**844.00**	2152.67	**818.87**	2200.00	**823.50**	261.46	**23.80**
COA	649.00	**611.00***	664.00	**612.00***	656.97	**611.30***	657.00	**611.00***	4.20	**0.47***
CSO	**650.00**	**650.00**	1994.00	**676.00**	715.50	**665.73**	667.00	**665.50**	241.86	**5.91**
DNLPSO	**786.00**	**786.00**	2171.00	**823.00**	1078.37	**807.03**	812.00	**807.00**	550.13	**9.08**
GWO	**674.00**	**674.00**	2043.00	**723.00**	960.17	**693.37**	695.00	**692.50**	543.32	**10.01**
MFO	2041.00	**690.00**	2095.00	**731.00**	2069.07	**710.73**	2067.00	**711.50**	12.68	**11.61**
MPEDE	652.00	**620.00**	680.00	**633.00**	664.10	**625.37**	664.00	**625.00**	7.03	**3.02**
MVO	699.00	**668.00**	2048.00	**715.00**	1155.47	**689.33**	727.00	**689.50**	626.66	**11.55**
SHO	2187.00	**828.00**	2222.00	**858.00**	2206.97	**844.87**	2209.00	**846.50**	8.93	**7.32**
SHTS	2051.00	**703.00**	2174.00	**809.00**	2103.43	**748.33**	2091.00	**745.00**	40.42	**25.37**
SOS	2043.00	**696.00**	2076.00	**726.00**	2064.90	**714.93**	2065.50	**716.00**	7.80	**6.38**
SSA	722.00	**697.00**	2086.00	**745.00**	1365.03	**720.30**	777.50	**720.00**	662.47	**12.18**
sTLBO	659.00	**642.00**	2131.00	**757.00**	814.43	**658.30**	669.50	**650.00**	439.55	**28.11**
YYPO	624.00	**612.00**	643.00	**618.00**	632.47	**615.73**	633.00	**616.00**	4.29	**1.34**

*Minimum value in each statistic measure.

In P6S1, it is observed that AWDO, MFO, SHO, SHTS, SOS, and SSA are unable to determine even one feasible solution in any run, while with the help of the heuristic mechanism, all these techniques determined feasible schedules in all the 30 runs. Moreover, the feasible mean value of all the metaheuristic techniques in P6S1 indicates that these techniques have determined

feasible schedules in all runs with the help of heuristic mechaism. As the number of feasible schedules in the search space is more in P6S2 than in P6S1, most of the techniques except MFO, SHO, SHTS, and SOS have determined at least one feasible schedule without the proposed heuristic mechanism. However, with the help of the heuristic mechanism, these techniques along with other techniques, have identified a schedule with better fitness value. The better mean value of metaheuristic techniques with the heuristic mechanism than those without the heuristic mechanism specifies their improved performance across all the runs. In P6S1 and P6S2, YYPO with the help of the heuristic mechanism has outperformed all other techniques by determining a schedule with a better fitness value.

The Gantt chart depicting the best schedule obtained by solving P6S1 and P6S2 is depicted in Fig. A1 and Fig. A2, respectively. It is observed that the no-wait time heuristic mechanism has aided YYPO (best technique) in determining a schedule without any idle time between the processing of two jobs in P6S1, whereas the best schedule determined by COA possesses idle time in P6S2, which is primarily due to the delayed arrival time of the succeeding job.

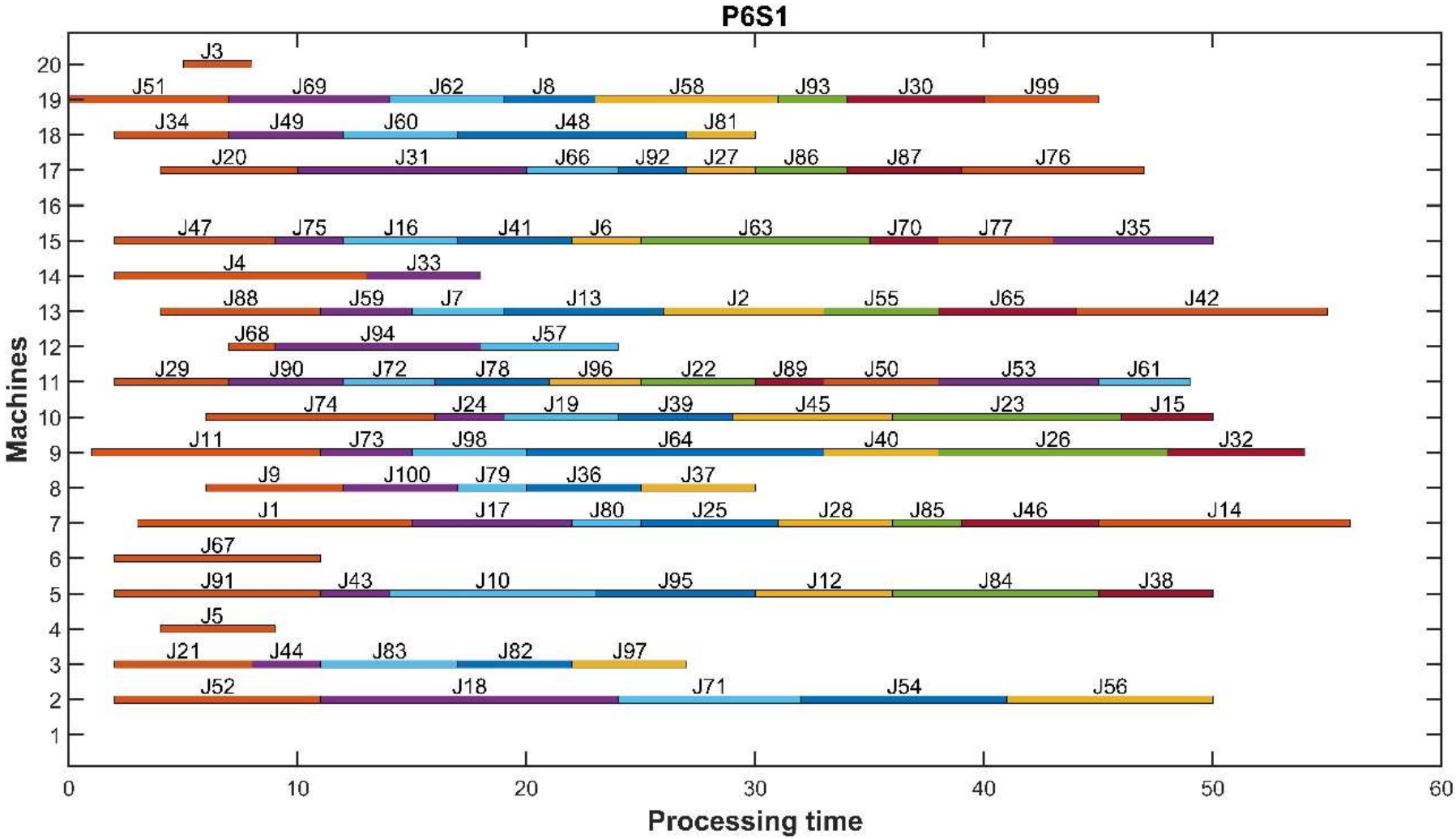


Fig E4. Gantt chart of best schedule of P6S1

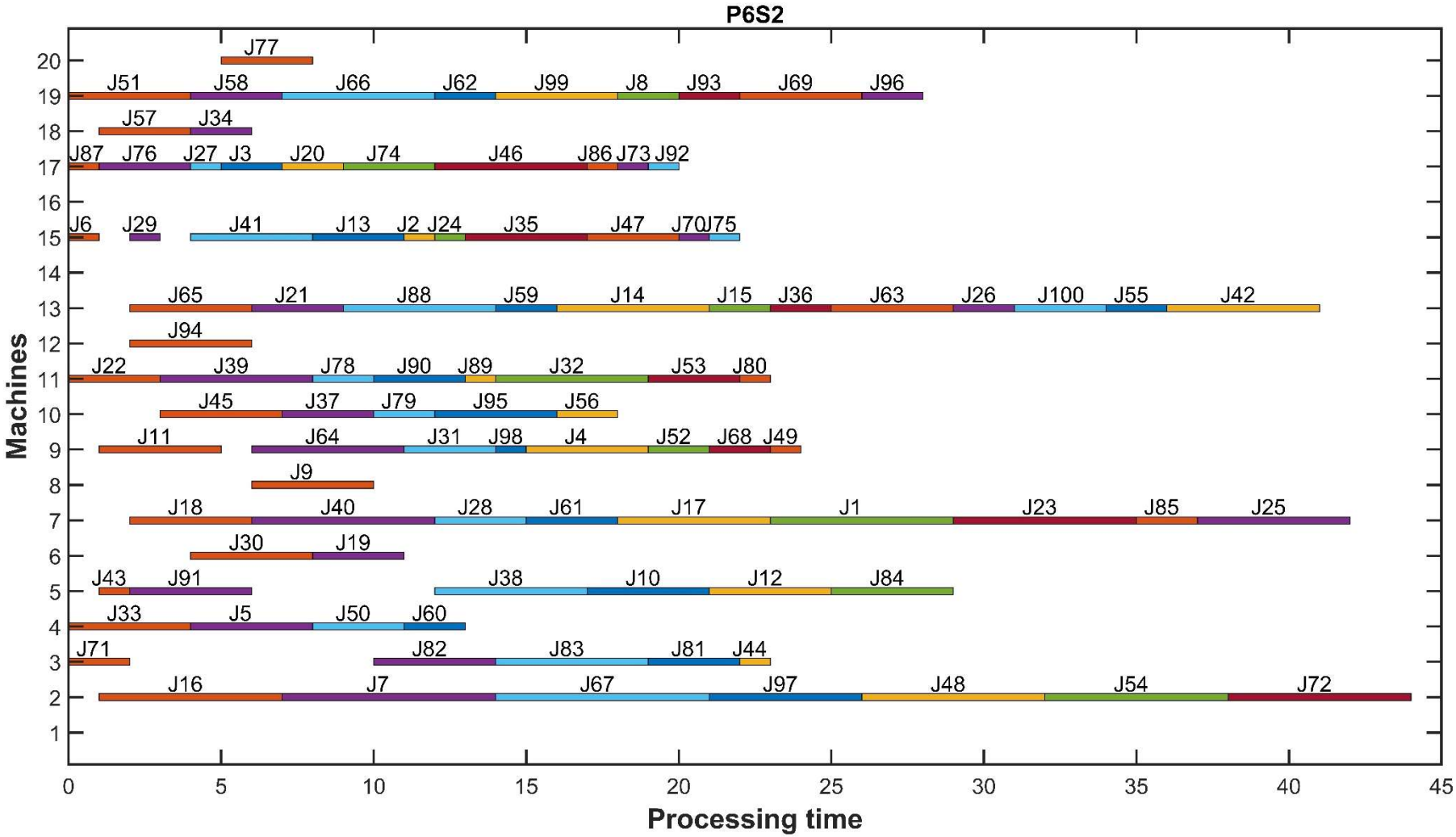


Fig E5. Gantt chart of best schedule of P6S2

Appendix F: Multi-objective scheduling in the vegetable processing and packaging facility using metaheuristic based framework

The use of sequence-dependent scheduling is important in vegetable processing and packaging facility as it requires the packaging of multiple types of products, which vary in terms of shelf life, shape, size, and texture. A sequence of similar vegetables requires small to negligible setup time, whereas a change in the type of vegetables will cause significant setup requirements. In such a scenario, scheduling vegetable consignments by considering only the processing time or the cost and completely neglecting the setup time required for the machines leads to inappropriate sequences in real-life cases. The scheduling problem considers real-life complexities, such as the arrival time and the discharge time of each vegetable consignment, apart from other conventional challenges. Multiple consignments must be processed using dissimilar parallel machines, which vary in processing duration and cost. The objective is to identify a schedule with the minimum total cost and minimum makespan (time span for completing the processing of all consignments) by satisfying the associated constraints.

As per the optimization framework proposed in Chapter 6, a metaheuristic technique assigns multiple vegetable consignments among the dissimilar parallel processing and packaging machines and schedules the processing of consignments on the assigned machine. The new wait time heuristic mechanism is applied to reschedule the processing of vegetable consignments on the assigned machine, thereby assisting metaheuristic techniques in identifying feasible solutions quickly. The studies performed in this work to analyze the efficacy of the proposed framework can be enumerated as (i) solved a case study of a single objective scheduling problem using the proposed framework with minimization of total cost and minimization of makespan as two individual objectives and compared the results with the solutions obtained without employing heuristic mechanism and (ii) solved the same case study

Kommadath, R., D. Maharana, R. Anandalakshmi, and P. Kotecha, *Multi-objective scheduling in the vegetable processing and packaging facility using metaheuristic based framework.* Food and Bioproducts Processing, 2023, **137**, p. 1-19. DOI: https://doi.org/10.1016/j.fbp.2022.10.005

of multi-objective scheduling problem using the proposed framework with minimization of total cost and minimization of makespan as conflicting objectives.

Appendix F is organized as follows: a brief description of the scheduling problem in the vegetable processing and packaging facility is provided in Section F.1. Section F.2 provides a detailed description of the proposed framework. The experimental settings utilized in this study are mentioned in Section F.3. A detailed analysis of the results obtained using the proposed framework and the metaheuristic techniques is performed in Section F.4. Finally, this work is concluded by consolidating the major findings of the study in Section F.5.

F.1. Problem description

A vegetable processing and packaging facility collects the vegetables from multiple farms and packs them in the desired manner. A single farm can send one or more types of vegetables to the packaging facility, whereas different farms can deliver the same vegetable. Each consignment from a particular farm varies in the type and quantity of vegetables. The processing and packaging facility constitutes multiple parallel machines that can pack various vegetables. Some of the available machines are fully automatic, while a few need additional requirements. Based on the extent of additional requirements and automation of the machines, the processing time and the cost needed to pack each vegetable consignment are different among machines. A machine can pack more than one vegetable, and the transition in packaging from one type of vegetable to another necessitates a setup time. A setup time is required due to the necessary pre-processing requirement of the machines to proceed with the next consignment. It varies with the next vegetable in the queue that has to be packed. A cost is associated with the setup time between the processing of vegetables, and it is directly proportional to the duration of the setup time. Different vegetable consignments from the farms arrive at the packaging facility simultaneously. Each vegetable consignment has an arrival time and a discharge time. The time at which the vegetable consignment is available for processing

and packaging in the facility is considered as the arrival time. A consignment with a zero arrival time specifies that it can be processed immediately. Any consignment with an arrival time greater than zero indicates lag time based on the initial pre-processing requirements. The discharge time is the maximum time by which these vegetables have to be packed and shipped from the facility to avoid any spoilage before reaching the market for consumption by customers. Makespan is the maximum time that elapses from the beginning to the completion of processing and packaging the consignments by considering all the machines. The objective of the problem is to schedule each vegetable consignment to the available machines based on optimizing the performance criteria of minimizing the total cost and minimizing the makespan while satisfying all the underlying constraints that are often found in standard scheduling problems (Jain & Grossmann, 2001).

(i) All the vegetable consignments must be packed by utilizing any available machines.

(ii) Once the packaging of a vegetable is initiated in a particular machine, it should be completed in the same machine.

(iii) The packaging of any vegetable consignment should not start before its arrival time as it is available for processing only at the arrival time (arrival time constraint).

(iv) A machine can process one vegetable consignment at a time (overlap constraint).

(v) The packaging of any vegetable cannot exceed its discharge time to avoid any spoilage (discharge time constraint).

(vi) Packaging of succeeding consignments assigned to a particular machine is not allowed without assuring the required setup time (setup time constraint).

Decision Variables: The allotted machine for each vegetable consignment and the start time of the consignment processing are the two types of decision variables. The values of these decision variables provide information such as (i) the choice of the machine to process each

consignment, (ii) the sequence of processing the consignments assigned to a particular machine, and (iii) the start time of the consignment processing on the assigned machine.

F.2. Proposed framework

The proposed optimization framework employed in Chapter 6 is used to solve the scheduling of vegetable processing and packaging facilities, which includes the metaheuristic technique, no-wait time heuristic technique, and the objective function.

F.2.1. Mathematical formulation of the fitness function

The constraints considered in the model given in Chapter 6 are also present in the scheduling model of the vegetable processing and packaging facility. However, the model in Chapter 6 did not consider the setup time between each job. However, it cannot be neglected for the vegetable consignments as it requires pre-processing. Hence, setup time constraints are additionally included in the model.

There should be a setup time ($W_{i,i+1}^{m}$) between two consecutive consignments of different types assigned to the same machine, and the duration of setup time depends on the succeeding vegetable consignment. The delay ($delay_{i,i+1}^{m}$) in processing the succeeding consignment using the m^{th} machine can be determined as given in Eq. (F.1), and the respective violation (Vw_{i}^{m}) is calculated using Eq. (F.2).

$$delay_{i,i+1}^{m} = t_{i+1} - end_{i}\ ,\ \forall i \in I_{m}, m \in 1,2,...,M : t_{i} < t_{i+1} \tag{F.1}$$

$$Vw_{i}^{m} = W_{i,i+1}^{m} - delay_{i,i+1}^{m} \tag{F.2}$$

The penalty (P_{i}^{m}) incurred in violating the setup time constraints for each consecutive consignment on m^{th} machines can be calculated using Eq.(F.3).

$$Pw_{i}^{m} = \max\left(0, Vw_{i}^{m}\right) \quad \forall i \in I_{m}, m \in 1,2,...,M \tag{F.3}$$

The setup cost is determined by multiplying a cost coefficient with the required setup time, which is calculated using the equation given in Eq. (F.4) , where α is the fixed cost coefficient and $W_{i,i+1}^{m}$ is the setup time required between i^{th} and $(i+1)^{\text{th}}$ order on the m^{th} machine.

$$Wc_{i,i+1}^{m} = \alpha W_{i,i+1}^{m}, \quad \forall i \in I_m \tag{F.4}$$

A binary variable (λ) is used to represent whether a potential schedule is feasible or not, and it is indicated as Eq. (F.5).

$$\lambda = \begin{cases} 0 & if \ P^{Total} = 0 \\ 1 & otherwise \end{cases} \tag{F.5}$$

Fitness function: The two objective functions considered in this study are the minimization of the total cost (f_{TC}) and the minimization of makespan (f_{MS}), which are calculated using Eq. (F.6) and Eq. (F.7). The total cost of a feasible schedule is the summation of the processing cost and the cost corresponding to the setup time between two consecutive consignment processing on all the machines. The fitness value of any infeasible schedule additionally includes the sum of all penalties corresponding to each constraint violation (P^{Total}) and the sum of the maximum cost for processing each consignment among all machines.

$$Min.\ f_{TC} = \sum_{i=1}^{I} c_{im} + \sum_{m=1}^{M} \sum_{i=1}^{|I_m|-1} Wc_{i,i+1}^{m} + P^{Total} + \lambda \sum_{i \in I} \max_{m \in M} (c_{im}) \tag{F.6}$$

In the above equation, c_{im} is the processing cost of i^{th} consignment on m^{th} machine and $Wc_{i,i+1}^{m}$ is the setup cost corresponding to the processing of i^{th} and $(i+1)^{th}$ consignments on m^{th} machine. The setup cost is determined by multiplying a cost coefficient with the required setup time. The maximum processing cost among all the machines for each consignment is added to the fitness value of an infeasible schedule to distinguish it from the worst feasible solution.

$$Min.\ f_{MS} = \left(\max_{i \in \{1,2,..,I\}} (end_i) - \min_{i \in \{1,2,..,I\}} (t_i) \right) + P^{Total} + \lambda \max_{i \in \{1,2,..,I\}} (d_i) \tag{F.7}$$

Adding maximum discharge time to the makespan of an infeasible schedule helps distinguish a feasible solution from an infeasible one. The two objectives given in Eq. (F.6) and Eq. (F.7) leads to a multi-objective optimization problem.

F.2.2. Heuristic mechanism

The heuristic mechanism follows a deterministic procedure to reschedule the consignments assigned to a particular machine. As per the potential schedule, when a machine is assigned with only one consignment, then its processing is rescheduled to its arrival time. In the case of multiple consignments assigned to a machine, sorting the consignments based on their start time is performed. The processing of the first consignment begins at its arrival time. Subsequent vegetable consignments are allowed to start processing immediately after the setup time required by the machine, which is dependent on the type of the succeeding consignment. The start time of all consignments except the first is determined using Eq. (F.8), where I_m is the set of consignments assigned to the m^{th} machine.

$$t_i = t_{i-1} + p_{(i-1)m_i} + W_{i-1,i} \quad \forall i \in 2,...,I_m \tag{F.8}$$

If the start time of a consignment violates any of the bounds, then the start time is shifted to the respective violated bound. The modified start time of the consignment that violates the lower bound is determined using Eq.(F.9), and the start time that violates the upper bound is modified using Eq. (F.10).

$$t_i = \max\left(t_i, r_i\right) \quad \forall i \in 1,2,...,I_m \tag{F.9}$$

$$t_i = \min\left(t_i, ub_i\right) \quad \forall i \in 1,2,...,I_m \tag{F.10}$$

F.2.3. Working of the proposed framework with metaheuristic techniques

In the proposed framework, the metaheuristic technique generates potential solutions within the bounds of decision variables to solve the scheduling problem. Prior to the fitness evaluation of each potential solution, it undergoes the heuristic mechanism. The heuristic mechanism

reschedules the processing of consignments assigned to each machine at the earliest possible time and does not modify the assignment of the consignment to the machines provided by the metaheuristic techniques. After the necessary modifications, the fitness value of the potential solution is evaluated, and then the modified solution, along with its fitness value, is returned to the metaheuristic technique. The solution strategy followed in the proposed framework is depicted in Fig. F1.

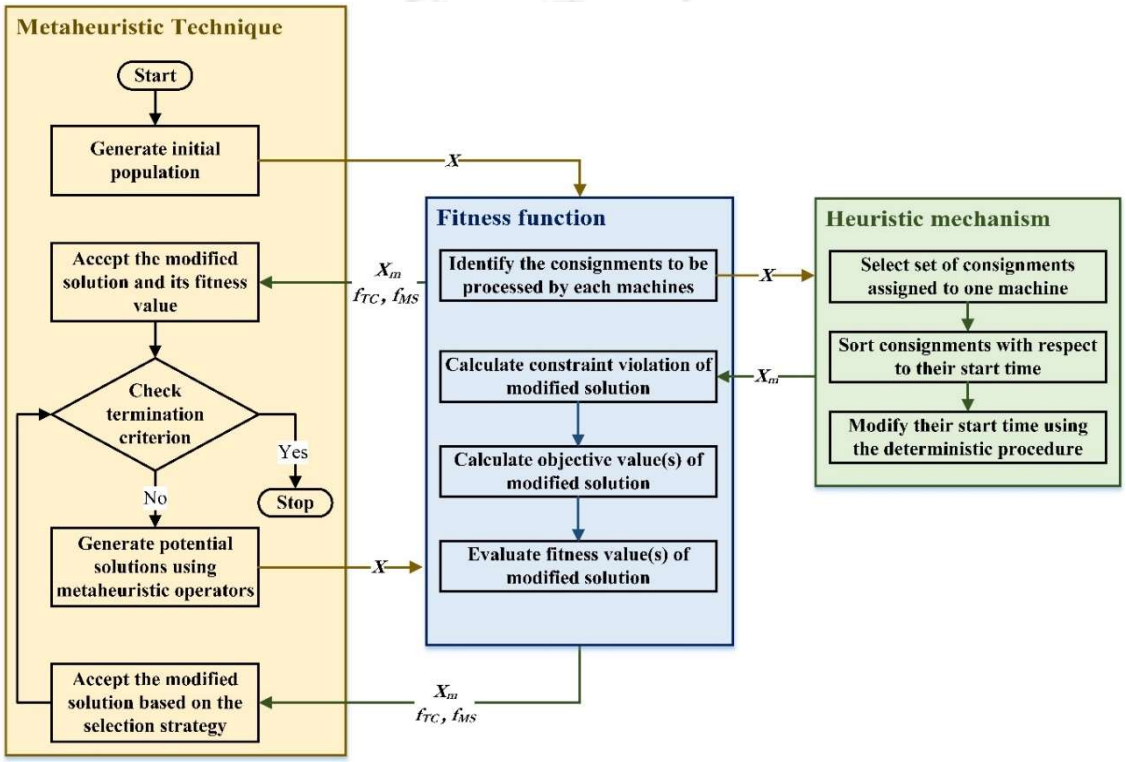


Fig. F1. Solution strategy employing the proposed framework

F.3. Experimental settings

The case study consists of twenty consignments of five different vegetables procured from nine farms. The arrival time of all the vegetables received at the processing and packaging facility from a specific farm is the same. In contrast, the discharge time is different based on the quality and shelf life of the vegetables. Five dissimilar parallel machines are available to process and pack each vegetable consignment which varies in the processing duration and cost. The setup time between the processing of two different consignments varies with the type of vegetable processing in a machine. The information regarding the arrival time and discharge time of each

vegetable consignment, the processing cost and duration required by each machine to process the consignment, and the setup time required between the processing of two different consignments are taken from the literature (Jain & Grossmann, 2001). The consignment number corresponding to vegetables from various farms is provided in Table F1.

Table F1. Consignment number corresponding to each vegetable consignment

Vegetables from various farms	Consignment number
$V_1^1, V_1^2, V_1^3, V_1^4, V_1^5, V_1^8$	C1, C2, C4, C9, C11, C18
$V_2^2, V_2^3, V_2^5, V_2^6, V_2^9$	C3, C5, C12, C14, C19
$V_3^3, V_3^5, V_3^7, V_3^9$	C6, C13, C16, C20
V_4^3, V_4^6, V_4^7	C7, C15, C17
V_5^3, V_5^4	C8, C10

In data, a consignment with a zero arrival time specifies that it can be processed immediately. Any consignment with an arrival time greater than zero indicates lag time based on the initial pre-processing requirements. The current work attempted to use a realistic scenario presented in (Fellows & Ouaouich, 2004), to demonstrate the efficacy of the proposed framework. It would encourage researchers to use the framework to test different scheduling datasets of various food and bioprocessing industries.

The efficacy of the proposed framework in solving the single objective model of the scheduling problem is analyzed using five metaheuristic techniques. The results obtained using the proposed framework are compared with those obtained using the metaheuristic techniques without incorporating the heuristic mechanism. The term naïve metaheuristic technique is used in this work to represent the metaheuristic technique without the heuristic mechanism. The current study also analyzed the efficiency of the proposed framework in solving the multi-objective model using three different multi-objective metaheuristic techniques.

The five metaheuristic techniques, namely Artificial Bee Colony (ABC) (Karaboga & Basturk, 2007), Coyote Optimization Algorithm (COA) (Pierezan & Coelho, 2018), Grey Wolf Optimizer (GWO) (Mirjalili et al., 2014), Multi-Population Ensemble Differential Evolution

(MPEDE) (Wu et al., 2016), and Sanitized Teaching Learning Based Optimization (sTLBO) are incorporated in this study. The multi-objective metaheuristic techniques employed to determine the Pareto solutions on minimizing the total cost and makespan are Multi-objective Cuckoo Search (MOCS) (Yang & Deb, 2013), Fast and Elitist Multi-objective Genetic Algorithm (NSGA II) (Deb et al., 2002) and Non-dominated Sorting Teaching Learning Based Optimization (NSTLBO). On account of the stochastic nature, each metaheuristic technique is executed for thirty independent runs. The population size considered for each algorithm is 200. The maximum allowed function evaluation (maxFE) is the termination criterion considered for all the techniques and is set to 4,00,000. The simulations are executed using MATLAB 2019a on a machine with an Intel i7 processor (3.6 GHz) and 32GB RAM.

F.4. Results and discussion

This section analyses the results obtained on solving the case study as a single objective scheduling problem with (a) minimization of total cost and (b) minimization of makespan as the objectives using the proposed framework and naïve metaheuristic techniques. Additionally, the analysis of the multi-objective optimization of the scheduling problem using the proposed framework is also provided in this section.

F.4.1. Single objective analysis: Minimization of total cost

This section analyzes the results obtained on solving the scheduling problem with minimization of the total cost using the proposed framework and the naïve metaheuristic techniques. The statistical analysis of the fitness value obtained by the proposed framework (PF) over the naïve metaheuristic technique is shown in Table F2. The minimum value among each statistic measure is highlighted using the **boldface** font. The analysis is a statistical summary based on the fitness value of each run.

Table F2. Statistical analysis of the total cost ($) by the metaheuristic techniques

Metaheuristic techniques	Best		Mean		Median		Standard Deviation	
	Naïve	PF	Naïve	PF	Naïve	PF	Naïve	PF
ABC	402.00	182.00	409.53	189.00	409.00	189.00	3.56	**3.27**
COA	184.00	178.00	383.10	199.53	403.00	185.50	64.76	52.84
GWO	396.00	181.00	411.23	283.90	410.50	194.00	8.95	104.88
MPEDE	187.00	**177.00**	382.03	**184.60**	403.00	**185.00**	64.95	3.89
sTLBO	393.00	180.00	410.10	198.40	411.00	**185.00**	6.79	52.55

The benefits of the heuristic mechanism are evident from the better statistic measures like best, mean, and median of the fitness values of the proposed framework than the naïve metaheuristic technique. A comparative analysis of the results obtained using the proposed framework and the naïve metaheuristic techniques suggests the superiority of MPEDE in determining the best fitness value followed by COA. The high standard deviation value of GWO with the heuristic mechanism and MPEDE without the heuristic mechanism shows their inconsistency, whereas COA is accompanied by a high standard deviation in both cases. The performance of the naïve COA is competitive with other naïve techniques by determining the lowest value in best and median statistics. The best, mean, and median of the fitness value determined by MPEDE within the proposed framework is better than other metaheuristic techniques within the proposed framework or as a naïve technique. The benefits of incorporating the heuristic mechanism are emphasized by the significantly better statistic measures obtained by the proposed framework than the naïve metaheuristic techniques. Moreover, the results also ensure the approach followed by the heuristic mechanism in handling the infeasible solutions aided the metaheuristic techniques in efficiently exploring the search space.

The Gantt charts provided in

Fig. F2 show the best schedule determined by the techniques with the proposed framework (COA) and naïve technique (MPEDE). On account of the idle time of machines, the best schedule obtained using the proposed framework (

Fig. F2(b)) has lesser idle time than the naïve metaheuristic technique (

Fig. F2(a)). The benefit of having less idle time in the best solution determined using the proposed framework is mainly due to scheduling all the jobs assigned to a particular machine at its earliest possible time. It is also observed from the schedule obtained using the proposed framework (

Fig. F2(b)) that there is no idle time as well as the setup time in machines M1 and M2. It is primarily because these machines are processing vegetable consignments of the same type. In all other machines, the delay in the processing of consignment is due to the requirement of the setup time.

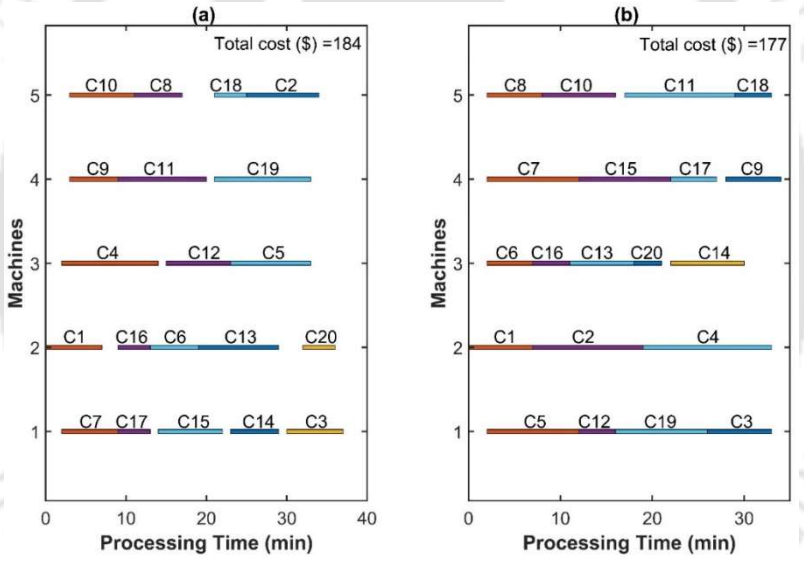


Fig. F2. Gantt chart of the best solution for the objective: minimization of total cost (a) naïve metaheuristic technique and (b) proposed framework

In the case of the solution determined using the naïve technique (

Fig. F2(a)), the setup time and idle time are observed between processing some of the consignments. For example, an idle time is observed on machine M1 while processing the consignments C17 and C15, even though they are of the same type. Such a scenario is also found in the case of consignments C13 and C20 (machine M2). After processing consignment

C8 in machine M5, the processing of consignment C18 is not started immediately after the necessary setup time, implying the inactivity of machine M5. Another evident advantage of the heuristic mechanism is that the makespan of the best schedule obtained with the assistance of the heuristic mechanism is lesser than the naive metaheuristic technique (

Fig. F2(a-b)).

The time and cost distribution on each machine as per the best schedule determined using the naïve metaheuristic technique and proposed framework are provided in Fig. F3. From Fig. F3 (a) and Fig. F3 (b), it can be observed that the time taken by each machine to process all the allotted consignments is approximately equal as per the best schedules obtained with the naïve technique and the proposed framework.

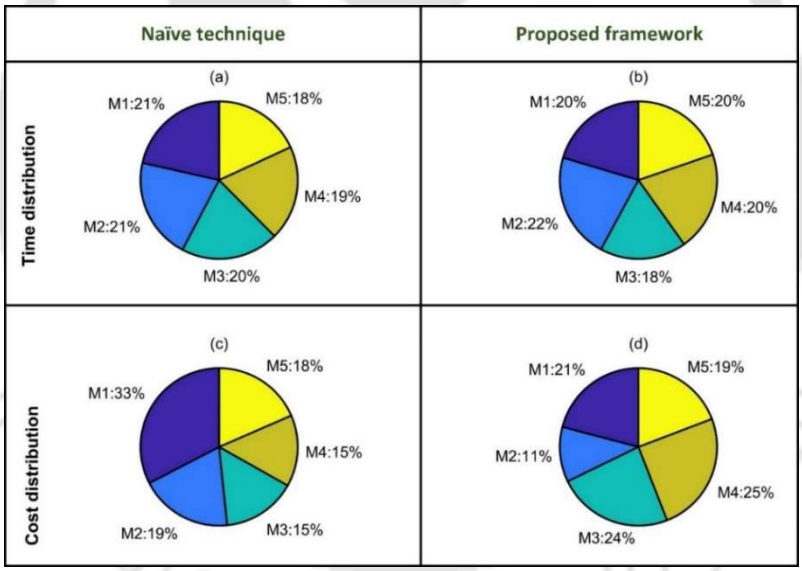


Fig. F3. Time and cost distribution on machines as per the schedule with minimum total cost obtained using naïve technique ((a) and (c)) and proposed framework ((b) and (d))

The cost distribution in all the machines is not equal as per the best solution determined by the naïve technique and proposed framework, which is evident from Fig. F3 (c) and Fig. F3 (d), respectively. However, it is observed from Fig. F3 (d) that the solution obtained using the proposed framework is able to reduce the number of consignments assigned to the costliest machine M1, and thus helps to reduce the total cost. As machine M1 is the most expensive

machine to process the consignments, its cost contribution is found to be the maximum in the solution determined using the naïve technique, which resulted in a solution with a higher cost than the solution of the proposed framework.

The convergence of median fitness value with respect to the function evaluations obtained with the naïve technique and proposed framework is as depicted in Fig. F4(a) and Fig. F4(b), respectively. The convergence of the median value by all the techniques using the proposed framework is better than the naïve metaheuristic techniques that emphasize the proposed framework helped the techniques to identify better solutions in most runs (Fig. F4(a-b)). The benefit of the heuristic mechanism on all the metaheuristic techniques is evident by discovering better schedules with respect to the total cost, even from the initial function evaluations. It can also be observed that the employment of the heuristic approach helps in efficient convergence of solutions by utilizing fewer function evaluations than the naïve technique (Fig. F4(b)).

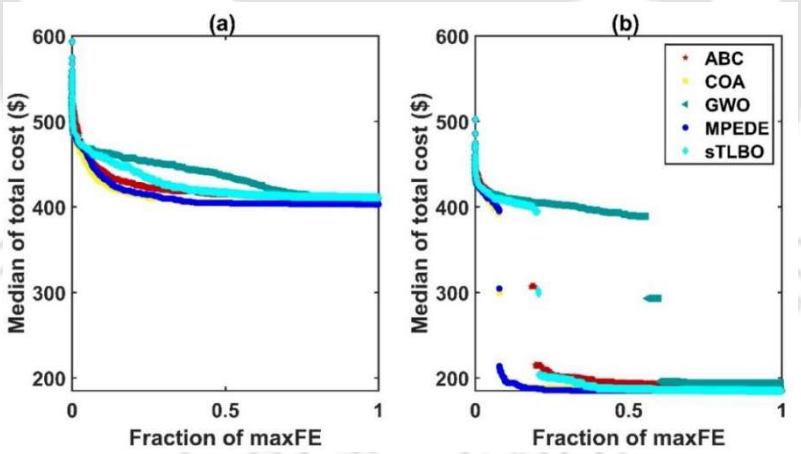


Fig. F4. Convergence of median total cost corresponding to the fraction of maximum function evaluations (maxFE) (a) naïve metaheuristic technique and (b) proposed framework

F.4.2. Single objective analysis: Minimization of makespan

The statistical analysis of the makespan value provided by using the proposed framework and naïve metaheuristic technique is given in Table F3, with the minimum value highlighted using

the **bold-face** font. In all the statistical measures, the benefits of the heuristic mechanism on the metaheuristic techniques in determining better solutions are evident. All the naïve metaheuristic techniques, except MPEDE, are not able to determine a feasible schedule in at least one run. Even though MPEDE is able to provide a feasible best solution, its infeasible mean and median value indicates that MPEDE is unable to determine a feasible solution in all the runs. On analyzing the best statistics, ABC, GWO, and MPEDE using the proposed framework have identified the schedule with minimum makespan. The least standard deviation value and the mean and median value closer to the best value of MPEDE with the proposed framework show its consistency in determining better solutions in all the runs

Table F3. Statistical analysis of makespan (min) determined by metaheuristic techniques

Metaheuristic techniques	Best		Mean		Median		Standard deviation	
	Naïve	PF	Naïve	PF	Naïve	PF	Naïve	PF
ABC	79	**32**	81.37	36.60	81	35	1.30	6.95
COA	75	33	78.93	35.93	79	35	2.63	7.07
GWO	76	**32**	88.27	39.17	87	36	6.83	11.89
MPEDE	34	**32**	73.30	**33.93**	77	**34**	12.80	**0.91**
sTLBO	75	33	86.87	37.53	84.5	35	8.22	10.13

The Gantt charts provided in Fig. F8 depict the best schedule (determined by MPEDE) obtained on minimizing the makespan using the naïve technique and proposed framework. It is observed from Fig. F8(a) that the maximum number of consignments are assigned to machine M1 in the best solution of the naïve metaheuristic technique, as it is the least time taking machine. Similarly, the minimum number of consignments is processed on machine M2, which is the most time-consuming machine.

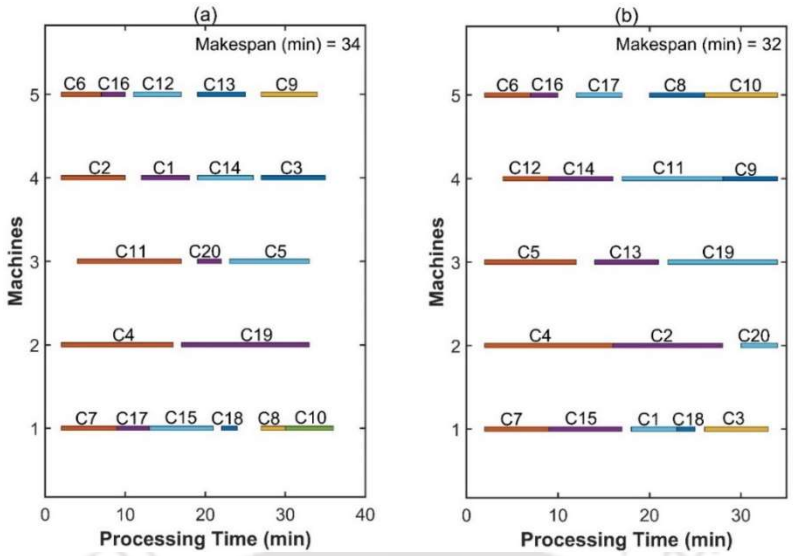


Fig. F5. Gantt chart of the best solution for objective: minimization of makespan (a) naïve metaheuristic technique and (b) proposed framework

The best schedule determined using the naïve technique that the distribution of consignments on machines is not equal neither in terms of the number nor in the total duration. In the case of the best schedule determined using the proposed framework (Fig. F8(b)), the total duration taken for processing all consignments assigned to each machine is approximately equal except for machine M4, indicating the good distribution of consignments based on their processing time. The efficient time distribution in all the machines using the proposed framework led to discovering a better solution.

The time and cost distribution per machine based on the best schedule obtained by the naïve metaheuristic technique and the proposed framework for minimizing the makespan are given in Fig. F6. In both the schedules, the time distribution observed in Fig. F6 (a) and Fig. F6 (b) specifies that all the machines are equally occupied. The best solution obtained using the naïve metaheuristic technique has attempted to assign maximum consignments to Machine M1, as it needs the least duration to complete all the consignments. However, as per the best schedule of the proposed framework, all the machines are equally distributed based on time and without overloading the least cost machines (Fig. F6 (b) and (d)).

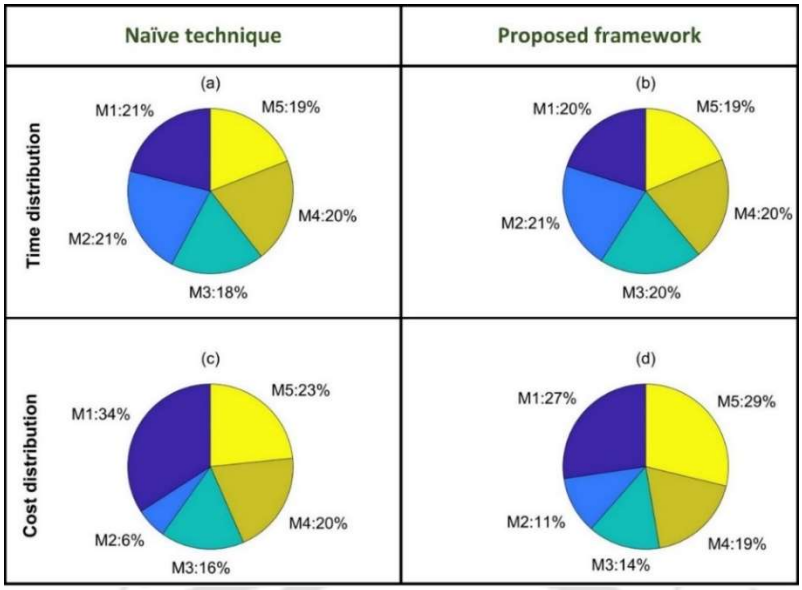


Fig. F6. Time and cost distribution on machines as per the schedule with minimum makespan obtained with ((a) and (c)) naïve technique and ((b) and (d)) proposed framework

The convergence of the median makespan value with the function evaluations of both the naïve metaheuristic technique and the proposed framework is given in Fig. F7. It is observed that all the naïve metaheuristic techniques are converged to a higher makespan value, while in all runs, the metaheuristic techniques with the proposed framework are able to discover better solutions by utilizing fewer numbers of function evaluations. The median makespan of MPEDE with heuristic mechanism is able to converge quickly to the final value.

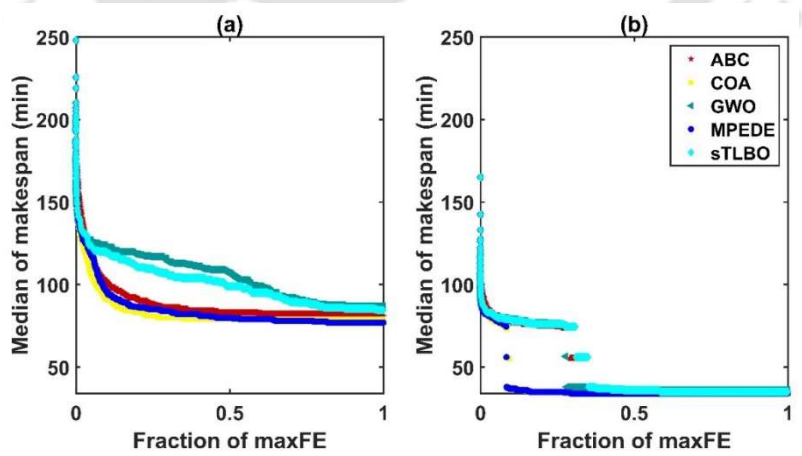


Fig. F7. Convergence of median makespan corresponding to the fraction of maximum function evaluations (maxFE) (a) naïve metaheuristic technique and (b) proposed framework

F.4.3. Multi-objective analysis

The result analysis of the proposed framework in solving the single objective model with minimization of total cost and minimization of makespan evidently reported the benefits of the proposed framework in determining better solutions over solving the case study using the naïve metaheuristic techniques. Hence the proposed framework is employed to solve the multi-objective model of the scheduling problem in vegetable processing and packaging facility.

The global Pareto solutions are identified from the set of non-dominated solutions determined by all the techniques. The non-dominated solutions identified by the metaheuristic techniques and the global Pareto front are provided in Fig. F8. It is evident from Fig. F8 (a) that the non-dominated solutions identified by NSTLBO strictly dominate all the non-dominated solutions of MOCS and NSGA II. The global Pareto front determined from the non-dominated solutions of all the techniques is provided in Fig. F8 (b). The makespan of the corner point representing the minimum makespan and the maximum total cost matches with the minimum makespan obtained on solving the single objective model. However, the total cost of the corner point with maximum makespan and minimum total cost is worse than the minimum total cost determined in the single objective model, indicating the scope of improvement in the Pareto point.

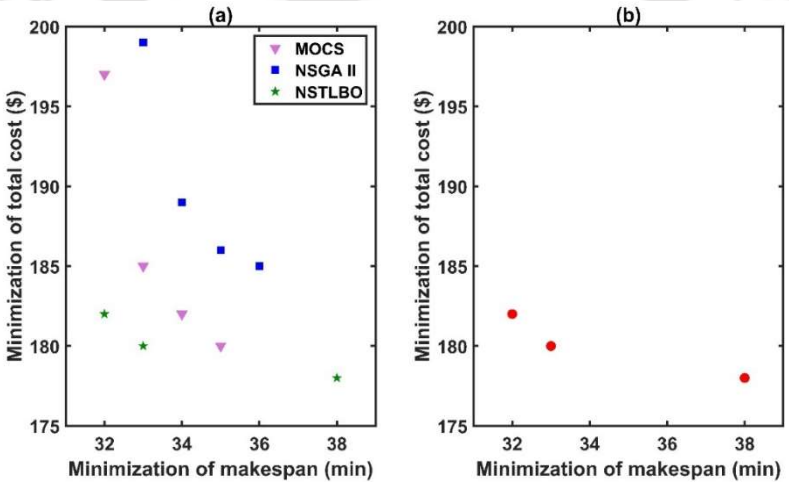


Fig. F8. Non-dominated solutions (a) determined by the metaheuristic techniques with the proposed framework and (b) present in the global Pareto front

The Gantt charts depicting all the three global Pareto points are provided in Fig. F9. It is observed that the makespan of the first solution (Fig. F9 (a)) is one unit lesser than the second solution (Fig. F9 (b)), while the second solution is better with a reduction of 2 units than the first solution regarding the total cost. Hence if a user provides priority for minimum total cost can proceed with the second solution compared to the first solution. A similar observation can be made by analyzing the second solution (Fig. F9 (b)) and the third solution (Fig. F9 (c)). Among the second and third solutions, the total cost of the second solution is two units higher than the third solution; however, it possesses a benefit of 5 units lesser makespan than the third solution. The second solution would be a better choice for a user whose priority is minimum makespan with an increase of two units in the total cost compared to the third solution. In comparison to the third solution, the consignments are distributed almost equally in all machines (except machine M2) for the first and second solutions. In the case of the third solution, the consignments are distributed based on the cost, and hence the third solution has the least total cost.

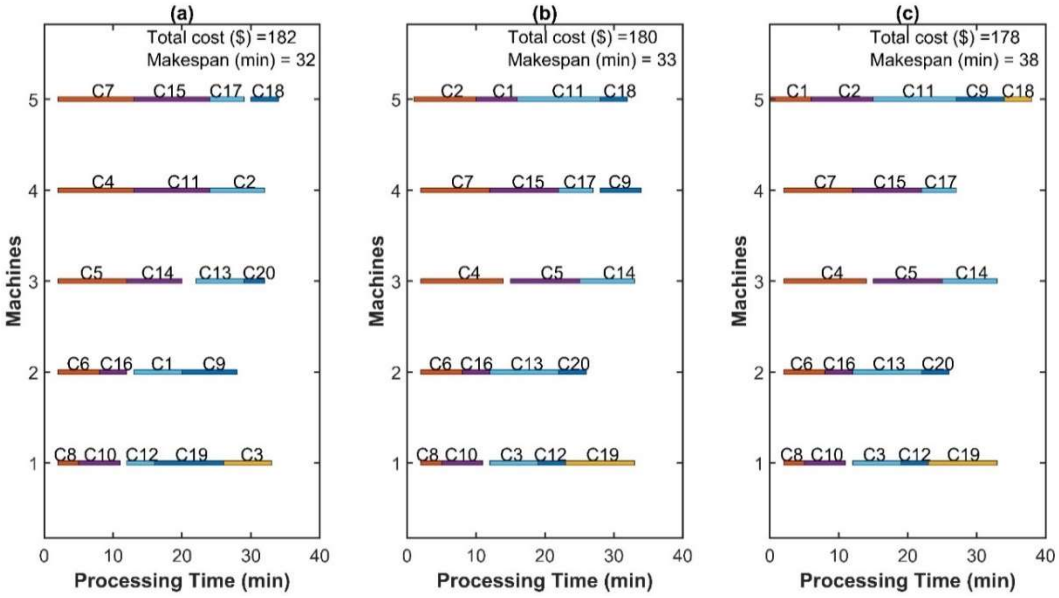


Fig. F9. Gantt chart of solutions in the global Pareto front (a) solution with minimum and makespan maximum total cost, (b) solution with moderate makespan and total cost, (c) solution with maximum makespan and minimum total cost

F.5. Conclusion

This study proposed an optimization framework utilizing a metaheuristic technique and a heuristic mechanism to solve the scheduling of a vegetable processing and packaging facility, delivering the vegetables to the end-users with extended shelf-life. The heuristic mechanism reschedules the consignment processing on each machine to reduce the violation of constraints, thereby assisting the metaheuristic technique in quickly searching for feasible solutions. The proposed framework was able to determine feasible schedules in the majority of runs for both objectives using all optimization techniques, which was not observed in the case of naïve metaheuristic techniques. The analysis revealed that the heuristic mechanism assisted the metaheuristic techniques to quickly obtain feasible schedules with a significant improvement in the final fitness value over naïve technique in both objectives. On optimizing the single objective model, MPEDE using the proposed framework outperforms all other techniques and has provided an improvement of 5% in both fitness values when compared to its naïve technique. The multi-objective framework of the scheduling problem is solved using three multi-objective metaheuristic techniques. NSTLBO, with the proposed framework, has identified non-dominated solutions that dominate the solutions determined by the other two techniques.

Data used for scheduling vegetable processing and packaging facility

Table F4. Data of the case study involving 20 consignments and 5 machines

Farms	Vegetable Consignment	Arrival time (min)	Discharge time (min)	Processing cost ($)					Processing time (min)				
				C1	C2	C3	C4	C5	D1	D2	D3	D4	D5
F1	C1	0*	30	13	7	10	12	11	5	7	7	6	6
F2	C2	1	34	12	7	10	11	10	7	12	10	8	9
	C3	1	37	8	5	6	7	6	7	14	11	8	10
	C4	2	33	10	6	8	9	9	10	14	12	11	13
	C5	2	33	12	10	11	12	11	10	13	10	11	12
F3	C6	2	20	13	7	10	12	11	3	6	5	4	5
	C7	2	25	8	5	6	7	7	7	14	13	10	11
	C8	2	34	9	5	7	9	8	3	8	7	5	6
F4	C9	3	34	8	5	6	7	7	6	8	7	6	7
	C10	3	37	10	6	8	9	8	6	12	10	7	8
	C11	4	31	12	7	10	11	10	11	16	13	11	12
F5	C12	4	25	9	5	7	9	8	4	10	8	5	6
	C13	4	32	9	5	6	8	7	4	10	7	5	6
F6	C14	5	33	10	6	8	9	8	6	12	8	7	8
	C15	5	30	15	9	12	14	13	8	16	12	10	11
F7	C16	6	20	10	6	8	10	9	2	4	4	3	3
	C17	6	32	15	9	12	14	13	4	7	6	5	5
F8	C18	7	38	8	5	6	7	6	2	8	6	13	4
F9	C19	10	34	8	5	6	7	7	10	16	12	12	13
	C20	10	38	10	6	8	9	8	2	4	3	2	3

Table F5. Setup (in min) time between two consecutive consignments

	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16	C17	C18	C19	C20
C1	0	0	1	0	1	2	2	3	0	3	0	1	2	1	2	2	2	0	1	2
C2	0	0	1	0	1	2	2	3	0	3	0	1	2	1	2	2	2	0	1	2
C3	1	1	0	1	0	2	2	3	1	3	1	0	2	0	2	2	2	1	0	2
C4	0	0	1	0	1	2	2	3	0	3	0	1	2	1	2	2	2	0	1	2
C5	1	1	0	1	0	2	2	3	1	3	1	0	2	0	2	2	2	1	0	2
C6	1	1	1	1	1	0	2	3	1	3	1	1	0	1	2	0	2	1	1	0
C7	1	1	1	1	1	2	0	3	1	3	1	1	2	1	0	2	0	1	1	2
C8	1	1	1	1	1	2	2	0	1	0	1	1	2	1	2	2	2	1	1	2
C9	0	0	1	0	1	2	2	3	0	3	0	1	2	1	2	2	2	0	1	2
C10	1	1	1	1	1	2	2	0	1	0	1	1	2	1	2	2	2	1	1	2
C11	0	0	1	0	1	2	2	3	0	3	0	1	2	1	2	2	2	0	1	2
C12	1	1	0	1	0	2	2	3	1	3	1	0	2	0	2	2	2	1	0	2
C13	1	1	1	1	1	0	2	3	1	3	1	1	0	1	2	0	2	1	1	0
C14	1	1	1	1	0	2	2	3	1	3	1	0	2	0	2	2	2	1	0	2
C15	1	1	1	1	1	2	0	3	1	3	1	1	2	1	0	2	0	1	1	2
C16	1	1	1	1	1	0	2	3	1	3	1	1	0	1	2	0	2	1	1	0
C17	1	1	1	1	1	2	0	3	1	3	1	1	2	1	0	2	0	1	1	2
C18	0	0	1	0	1	2	2	3	0	3	0	1	2	1	2	2	2	0	1	2
C19	1	1	0	1	0	2	2	3	1	3	1	0	2	0	2	2	2	1	0	2
C20	1	1	1	1	1	0	2	3	1	3	1	1	0	1	2	0	2	1	1	0

Appendix G: Thermoeconomic model of CACRS and additional results

Nomenclature

b	Specific exergy, kJ/ kg	T	Temperature, ˚C/K
$\dot{B}$	Exergy flow, kW	T_m	Logarithmic mean temperature difference
$\dot{B}_D$	Exergy destruction, kW	T_C	Crystallization temperature, ˚C
C	Cost, US$	$T_{overlap}$	Temperature difference in the cascade-condenser, ˚C
CCU	Cost of cold utility, US$/year	$\dot{W}$	Work, kW
COP	Coefficient of performance	X,w	Concentration of absorbent solution, %
C_{CO_2}	Cost for CO_2 emission, US$/year	Z	Total system capital cost, US$
C_P	Specific heat capacity, kJ/ kg. K	1,2,3…	State points
CRF	Capital recovery factor		**Geek letters**
$\dot{E}$	Energy, kW	ε	Effectiveness of solution heat exchanger
h	Specific enthalpy, kJ/kg	η	Efficiency
IR	Interest rate, %	λ	Emission factor, kg/ kW. h
$\dot{m}$	Mass flow rate, kg/s	ρ	Density, kg/m^3
MF	Maintenance factor		**Subscripts**
m_{CO_2}	Mass of CO_2 emission, kg	0	Reference value
PR	Period of repayment, years	T	Total
P	Pressure, kPa/bar	r	Real
$\dot{Q}$	Heat load, kW		
S	Specific entropy, kJ/ kg. K		
OHP	Period of operation, hours		

Thermoeconomic and environmental model

Steady-state energy and exergy analysis is carried out for a refrigeration load of 80.7 kW by excluding kinetic and potential energy. Atmospheric temperature and pressure (298 K and 101.3 kPa) are considered as references. Heat transfer to the surroundings is neglected except where otherwise stated. Pressure and heat losses are neglected throughout the CACRS. The energy given to any other additional components of CACRS, whichever is not mentioned, is considered negligible. Adiabatic expansion is considered at expansion valves. Concentration and all other thermodynamic properties of the absorbent solution are equilibrium values at their corresponding temperatures and pressures. All components of CACRS are maintained at a

constant temperature. The refrigerant leaves the condenser and evaporator, and the absorption solution leaves the absorber and the generator at their saturated state.

The objective function of the CACRS model for minimizing the total annual cost is given below by relating system cost with its exergy.

$$\begin{aligned} min\ C_T(x) = OHP\left[C_{exergy}\dot{B}_{input\,exergy} + C_{electricity}\ \dot{W}_{total}\right] + \\ CRF\ Z_{total} + CUC\left[\dot{Q}_{condenser} + \dot{Q}_{absorber}\right] + C_{environment} \end{aligned} \tag{K.1}$$

In Eq. (K.1), the first two terms depict the exergy cost and electricity consumption cost for the compressor and pump, the third term is the initial investment cost for all components of CACRS, the fourth term depicts the cold utility cost and the fifth term represents the penalty cost due to CO_2 emission.

Exergy cost

The second law of thermodynamics based exergy analysis is applied to CACRS as follows:

$$\dot{B} = \dot{Q}\left(1 - \frac{T_0}{T}\right) - \dot{W} + \sum_{input} \dot{m}b - \sum_{output} \dot{m}b - \dot{B}_D \quad \text{where} \quad b = (h - h_0) - T_0(s - s_0) \tag{K.2}$$

Mass and energy balances for CACRS are given as follows:

$$\dot{m} = \sum_{input} \dot{m} - \sum_{output} \dot{m} \tag{K.3}$$

$$\dot{E} = \dot{Q} - \dot{W} + \sum_{input} \dot{m}h - \sum_{output} \dot{m}h \tag{K.4}$$

Exergy destruction in every individual component of CACRS, total exergy destruction of CACRS, and input exergy to CACRS are given in Eq. (K.5) − (K.7), considering only physical exergy.

$$\dot{B}_D = \sum \dot{B}_{input} - \sum \dot{B}_{output} \tag{K.5}$$

$$\dot{B}_{D,total} = \sum_{\substack{all\ heat \\ exchangers}} \dot{B}_D + \dot{W}_{total} \tag{K.6}$$

$$\dot{B}_{input\ exergy} = \dot{B}_{D,total} + \left(\dot{B}_{21} - \dot{B}_{22}\right) \tag{K.7}$$

Electricity consumption cost for compressor and pump

The total electrical cost corresponding to the compressor and pump is calculated using work done by the compressor and the pump of CACRS and is given below

$$\dot{W}_{compressor} = \dot{m}_{VCRS\,refrigerant}\left(h_{4r} - h_3\right) \text{ where } h_{4r} = h_3 + \frac{\left(h_{4isentropic} - h_3\right)}{\eta_{compressor}} \tag{K.8}$$

$$\dot{W}_{pump} = \frac{\dot{m}_6\left(P_{13} - P_5\right)}{\rho_{mixture}\eta_{pump}} \text{ where } \rho_{mixture} = X_6\rho_{salt\,solution} + \left(1 - X_6\right)\rho_{water} \tag{K.9}$$

$$\dot{W}_{total} = \dot{W}_{compressor} + \dot{W}_{pump} \tag{K.10}$$

Investment cost of all components of CACRS

The investment cost depends on the size of all components present in CACRS, except for absorbent solution, refrigerants, valves, pumps, and pipes, as shown in Eq. (K.11).

$$Z = 516.6211\ A + 268.45 \tag{K.11}$$

The size of the components (A) is calculated based overall heat transfer coefficient (U) according to various components of CACRS, which is given as,

$$A = \frac{\dot{Q}}{U\,\Delta T_{LMTD}} \tag{K.12}$$

$$\eta_{isentropic} = 0.85 - 0.046667\left(\frac{P_1}{P_3}\right) \tag{K.13}$$

$$\eta_{isentropic} = 0.85 - 0.046667\left(\frac{P_1}{P_3}\right) \tag{K.14}$$

$$Z_{total} = \left(\sum_{\substack{all\,heat\\ Exchangers}} Z + Z_{compressor}\right) MF \tag{K.15}$$

The capital recovery factor (CRF) is represented as,

$$CRF = \left(\frac{RI\left(1 + RI\right)^{PR}}{\left(1 + RI\right)^{PR} - 1}\right) \tag{K.16}$$

Heat balance equations for all the components of CACRS are given in Table G.1.

Table G.1 Heat balance equations for all CACRS components

Heat balance equations	*LMTD*
$\dot{Q}_{evaporator}=\dot{m}_{VCRS\,refrigerant}\left(h_3-h_2\right)=\dot{m}_{21}\left(h_{21}-h_{22}\right)$	$T_{m,evaporator}=\frac{\left(T_{21}-T_2\right)-\left(T_{22}-T_3\right)}{log\left(\frac{T_{21}-T_2}{T_{22}-T_3}\right)}$
$\dot{Q}_{cascade\ condenser}=\dot{m}_{VARS\,refrigerant}\left(h_5-h_{14}\right)=\dot{m}_{VCRS\,refrigerant}\left(h_4-h_1\right)$	$T_{m,cascade\ condenser}=\left(T_1-T_5\right)$
$\dot{Q}_{absorber}=\dot{m}_{VARS\,refrigerant}h_5+\dot{m}_{11}h_{11}-\dot{m}_6h_6=\dot{m}_{20}\left(h_{20}-h_{19}\right)$	$T_{m,absorber}=\frac{\left(T_{11}-T_{20}\right)-\left(T_6-T_{19}\right)}{log\left(\frac{T_{11}-T_{20}}{T_6-T_{19}}\right)}$
$\dot{Q}_{generator}=\dot{m}_{VARC\,refrigerant}h_{12}+\dot{m}_{11}h_9-\dot{m}_6h_8=\dot{m}_{17}\left(h_{17}-h_{18}\right)$	
$\dot{Q}_{solution\,heat\,exchanger}=\dot{m}_{11}\left(h_9-h_{10}\right)=\dot{m}_6\left(h_8-h_7\right)$	$T_{m,solution\,heat\,exchanger}=\frac{\left(T_9-T_8\right)-\left(T_{10}-T_7\right)}{log\left(\frac{T_9-T_8}{T_{10}-T_7}\right)}$
$\dot{Q}_{condenser}=\dot{m}_5\left(h_{12}-h_{13}\right)=\dot{m}_{15}\left(h_{16}-h_{15}\right)$	
$\dot{m}_6\left(X_9-X_6\right)=\dot{m}_{VARS\,refrigerant}X_9$	$T_{m,generator}=\frac{\left(T_{17}-T_9\right)-\left(T_{18}-T_{12}\right)}{log\left(\frac{T_{17}-T_9}{T_{18}-T_{12}}\right)}$
$\dot{m}_9\left(X_9-X_6\right)=\dot{m}_{VARS\,refrigerant}X_6$	
$\varepsilon=\frac{\left(\dot{m}C_p\right)_9}{\left(\dot{m}C_p\right)_{minimum}}\frac{T_{12}-T_{10}}{T_{12}-T_6}=\frac{\left(\dot{m}C_p\right)_6}{\left(\dot{m}C_p\right)_{minimum}}\frac{T_8-T_6}{T_{12}-T_6}$	$T_{m,condenser}=\frac{\left(T_{13}-T_{15}\right)-\left(T_{13}-T_{16}\right)}{log\left(\frac{T_{13}-T_{15}}{T_{13}-T_{16}}\right)}$

Penalty cost

Generation of steam and electricity using fossil fuels is accompanied by the emission of CO_2 into the atmosphere, and the penalty cost is charged for the emission of CO_2 as,

$$C_{penalty\,cost}=OHP\left[\lambda\dot{W}_{total}C_{CO_2}\right] \tag{K.17}$$

The COP of the VCRS, VARS, and CACRS are given as below

$$COP_{VCRS}=\frac{\dot{Q}_{evaporator}}{\dot{W}_{compressor}} \tag{K.18}$$

$$COP_{VARS}=\frac{\dot{Q}_{cascade\ condenser}}{\dot{Q}_{generator}+\dot{W}_{pump}} \tag{K.19}$$

$$COP_{CACRS} = \frac{\dot{Q}_{evporator}}{\dot{Q}_{generator} + \dot{W}_{total}} \quad \text{(K.20)}$$

The optimization model follows a solution approach that accounts for the equations and constraints present in the CACRS model. A pseudocode demonstrating the steps of the solution approach is given in Fig. G.1.

Step 1	Acquire values of decision variables R, S, $T1$, $T3$, $T6$, $T12$, $T13$, $Toverlap$, and ε from the metaheuristic technique
Step 2	Use REFPROP to determine the appropriate state properties for the refrigerants, coolants, and hot streams
Step 3	***if*** $X_9 - X_6 > 0$ | Assign TAC = 10^7 Go to Step 15
Step 4	Calculate $\dot{m}_{ref1}$, $\dot{m}_{ref2}$, $\dot{m}_6$ and $\dot{m}_9$
Step 5	Calculate C_{p6} and C_{p9}
Step 6	***if*** $\dot{m}_6 C_{p6} < \dot{m}_9 C_{p9}$ | Assign $(\dot{m}C)_{min} = \dot{m}_6 C_{p6}$ **else** | Assign $(\dot{m}C)_{min} = \dot{m}_9 C_{p9}$
Step 7	Calculate T_8 and T_{10}
Step 8	***if*** $T_{10} - T_C < 5$ | Assign TAC = 10^7 Go to step 15
Step 9	Calculate H for absorbent solution
Step 10	Calculate $\dot{Q}$ and $\dot{m}_{15}$ to $\dot{m}_{22}$ for coolant and hot streams
Step 11	Calculate $\dot{W}_C$ and $\dot{W}_p$
Step 12	Calculate T_m, A, Z for all CACRS components
Step 13	Calculate b, B_{in}, B_{DT}
Step 14	Calculate penalty cost and TAC
Step 15	Return the decision variables and the TAC to the algorithm

Fig. G.1 Pseudocode of the solution procedure

Additional thermodynamic parameters

Table G.2 gives the details about COP, heat load, and area of the heat exchanger corresponding to the minimum total annual cost calculated using optimal operating parameters as selected by

all algorithms for the best absorbent solution-refrigerant-refrigerant combinations. The energy efficiency of the system is related to COP, whereas exergy destruction corresponds to the amount of unused available energy by the system. The investment cost of the refrigeration system is calculated based on the area of heat exchangers. Hence, the effect of optimum operational parameters on these parameters is discussed.

As an example, the thermodynamic analysis of ($CaCl_2$-LiBr-$LiNO_3$)-H_2O-R152a and ($CaCl_2$-LiBr-$LiNO_3$)-H_2O-R123, which gives the lower and higher minimum total annual cost, is discussed. There is no remarkable difference in COP_{VCRS} between ($CaCl_2$-LiBr-$LiNO_3$)-H_2O-R152a and ($CaCl_2$-LiBr-$LiNO_3$)-H_2O-R123 due to identical evaporator temperature (T_3) and compressor work (W_c). The trade-off between heat load on the generator and cascade condenser as well as the identical exergy destruction, leaves no significant difference in COP_{VARS} and COP_{CACRS} between ($CaCl_2$-LiBr-$LiNO_3$)-H_2O-R123 and ($CaCl_2$-LiBr-$LiNO_3$)-H_2O-R152a as shown in Table G.2. It is noted that a 2˚C increase in T_{13} resulted in around 1.5 m^2 reduction in condenser area for the identical heat load and given coolant conditions for ($CaCl_2$-LiBr-$LiNO_3$)-H_2O-R123 when compared to ($CaCl_2$-LiBr-$LiNO_3$)-H_2O-R152a. This is due to the reduction in the strong solution concentration, i.e., absorption of refrigerant is more, and hence, the mass flow rate of a weak and strong solution decreases. The size of the solution exchanger is lesser for ($CaCl_2$-LiBr-$LiNO_3$)-H_2O-R123 than ($CaCl_2$-LiBr-$LiNO_3$)-H_2O-R152a which is due to the lesser thermal compressor heat load as reported in Table G.2. The reduction of T_{over} increases the temperature and evaporator area in VARS (HTL) for ($CaCl_2$-LiBr-$LiNO_3$)-H_2O-R123, and, therefore, the cascade condenser area is increased by around 9 m^2. Hence, the size of all components of CACRS and the total minimum annual cost is increased for ($CaCl_2$-LiBr-$LiNO_3$)-H_2O-R123 when compared to ($CaCl_2$-LiBr-$LiNO_3$)-H_2O-R152a. Lower condenser temperature in VARS is always necessary for the better performance of CACRS.

Table G.2 Additional thermodynamic parameters for CACRS

Algorithm	Absorbent solution	Refrigerant	COP (VCRS)	COP (VARS)	COP (CACRS)	BDT (kW)	Compressor work load (kW)	Absorber heat load (kW)	Generator heat load (kW)	Condenser heat load (kW)	Cascade condenser heat load (kW)	Area of cascade condenser (m^2)	Area of evaporator (m^2)	Area of absorber (m^2)	Area of heat exchanger (m^2)	Area of generator (m^2)	Area of condenser (m^2)
ABC	$CaCl_2$-LiBr-$LiNO_3$-H_2O	R152a	18.78	0.93	0.84	13.35	4.30	85.70	91.67	90.12	85.00	15.00	9.54	9.42	9.80	5.70	5.13
CSA		R152a	18.78	0.93	0.84	13.31	4.30	85.53	91.44	90.09	85.00	14.90	9.54	9.46	9.79	5.43	5.50
DNLPSO		R152a	18.78	0.93	0.84	13.33	4.30	85.59	91.53	90.11	85.00	14.94	9.54	9.44	9.77	5.56	5.33
FP		R290	18.66	0.92	0.83	13.50	4.32	86.26	92.32	90.22	85.02	14.91	9.33	9.34	10.11	6.21	4.59
GA		R152a	18.78	0.93	0.85	13.25	4.30	85.24	91.08	90.03	85.00	15.24	9.54	9.45	9.70	5.12	5.94
MFO		R290	18.48	0.93	0.84	13.41	4.37	85.65	91.60	90.18	85.07	15.01	9.54	9.43	9.74	5.61	5.27
MPEDE		R152a	18.78	0.93	0.84	13.33	4.30	85.58	91.52	90.11	85.00	15.00	9.54	9.43	9.73	5.59	5.29
SHO		R123	18.88	0.94	0.85	13.11	4.27	84.71	90.43	89.90	84.97	24.28	9.54	9.30	8.50	5.68	4.08
SOS		R152a	18.76	0.93	0.84	13.30	4.30	85.41	91.32	90.09	85.00	15.15	9.49	9.44	9.63	5.45	5.47
SPMGTLO		R152a	18.78	0.93	0.84	13.33	4.30	85.58	91.52	90.11	85.00	14.98	9.54	9.43	9.74	5.57	5.31

Appendix H: Additional details of sTLBO and its parallel variants

Psuedocode H I: ***Sanitized Teaching Learning based Optimization***

Step 1: Define the bounds of the decision variables, the dimension of the function (N_d) and the objective function (or fitness function for constrained optimization problems)

Step 2: Initialize the algorithmic parameters, such as the class size (N_P) and the number of iterations (N_G).

Step 3: Generate a class of learners randomly such that the marks (or decision variables) of each learner in every subject is within its specified domain. Evaluate the fitness of each learner.

Step 4**:** Initialize the iteration counter, *g* as zero

Step 5: Increment the *g* by 1

Step 6: Initialize class counter to zero

Step 7: Increment the class counter by 1

Step 8: *Beginning of the teacher phase:* Determine the best learner *Xbest* in the class based on the value of the fitness function. This learner will act as the teacher for this learner.

Step 9: Generate a random number (*rand*) between 0 and 1. Determine the teaching factor T_F using $T_F = round(1+rand)$

Step 10: Determine the average marks of the class in each subject (*Xmean*)

Step 11: Determine the marks of the potential new solution of the class using the following expression, $T_j^{new} = S_j^i + r_j\left(S_j^{best} - T_F S_j^{mean}\right) \quad \forall j = 1,..,N_d$. In this expression, T_j^{new} is the mark of the potential new solution in the subject *j*, S_j^i is the mark in the subject *j* of the i^{th} learner, r_j is a random number between 0 and 1, S_j^{best} is the mark in the subject *j* of S^{best}, S_j^{mean} is the average obtained in subject *j* of all solutions in the class.

Step 12: Ensure that each of the decision variables of the potential learner $\left(T_j^{new}\right)$ is within its bound using $T_j^{new} = \begin{cases} \max\left(T_j^{new}, lb_j\right) \\ \min\left(T_j^{new}, ub_j\right) \end{cases} \quad \forall j = 1,..,N_d$ where lb_j and ub_j are the lower and upper bounds of the decision variable j.

Step 13: Evaluate the fitness of T^{new}. If the fitness of T^{new} is better than the current learner undergoing the teacher phase, replace the current learner with T^{new} or else discard T^{new}. *End of teacher Phase*

Step 14: *Beginning of learner Phase* Select a random partner (S^{rand}) from the class.

Step 15: Based on the fitness of (S^{rand}) and the fitness of (S^i), a potential learner is determined using the relation $S_j^{new} = \begin{cases} S_j^i + r_j\left(S_j^i - S_j^{rand}\right); & f\left(S^i\right) \le f\left(S^{rand}\right) \\ S_j^i + r_j\left(S_j^{rand} - S_j^i\right); & f\left(S^i\right) > f\left(S^{rand}\right) \end{cases} \quad \forall j = 1,..,N_d$

Step 16: Ensure that each of the decision variables of the potential learner is within its bound using $S_j^{new} = \begin{cases} \max\left(S_j^{new}, lb_j\right) \\ \min\left(S_j^{new}, ub_j\right) \end{cases} \quad \forall j = 1,..,N_d$

Step 17: Evaluate the fitness of S^{new}. If the fitness of S^{new} is better than S^i, replace S^i with S^{new}, else discard S^{new}. *End of learner Phase*

Step 18: Go to Step 7 until all solutions complete one iteration of the teaching-learning process.

Step 19: Go to Step 5 until the required number of iterations is completed.

Step 20: Select the best solution in the entire class as the optimal solution.

Psuedocode H II: ***Member Parallelized Teaching Learning based Optimization***

Step 1: Define the bounds of the decision variables, the number of decision variables (N_d) and the objective function (or fitness function for constrained optimization problems).

Step 2: Initialize the algorithmic parameters, such as class size (N_P), and the number of iterations (N_G)

Step 3: Generate a class of learners randomly such that the marks (or decision variables) of each learner in every subject is within its specified domain. Evaluate the fitness of each learner.

***Step 4*:** Initialize the iteration counter as zero

Step 5: Increment the iteration counter by 1

Step 6: Initialize class counter to zero

Step 7: Determine the learning partner $S^{partner}$ for each learner by randomly permuting the numbers (1,2,…,N_P). Let $S^{partner,i}$ indicate the learner who is partnering with the learner i

Step 8: Increment the class counter by 1

Step 9: Determine the best learner S^{best} in the class based on the value of the objective function

Step 10: Determine the teaching factor T_F for each decision variable as a value of either 1 or 2.

Step 11: Determine the average mark of the class in each subject (S^{mean})

Step 12: Determine a potential solution of the class using the following expression $T_j^{new} = S_j^i + r_j\left(S_j^{best} - T_{F,j}S_j^{mean}\right) \quad \forall j = 1,..,N_d$. In this expression, T_j^{new} is the mark of the potential solution in the subject j , S_j^i is the mark in the subject j of the current learner, r_j is a random number between 0 and 1, S_j^{best} is the mark scored by the learner with the best objective

function value in the subject j, S_j^{mean} is the average of all the solutions of the population in subject j.

Step 13: Generate two potential learners using the following relation

$$S_j^{1new} = S_j^i + r_j\left(S_j^i - S_j^{partner,i}\right) \qquad \forall j = 1,..,N_d$$
$$S_j^{2new} = S_j^i + r_j\left(S_j^{partner,i} - S_j^i\right) \qquad \forall j = 1,..,N_d$$

where $S_j^{partner,i}$ indicates the marks in the subject j of the learner who is partnering with the current solution (S^i) undergoing the learner phase.

Step 14: Ensure that all the decision variables of the three potential learners T^{new}, S^{1new} and S^{2new} are within their bounds. If lb_j and ub_j are the lower and upper bounds of the decision variable j, a solution (X) can be bounded as shown in the following equation

$$X_j = \begin{cases} \max\left(X_j, lb_j\right) \\ \min\left(X_j, ub_j\right) \end{cases} \qquad \forall j = 1,..,N_d$$

Step 15: Evaluate the fitness of T^{new}, S^{1new} and S^{2new}.

Step 16: Combine these three new potential learners with the original class and determine the best N_P solutions from the $N_P + 3$ solutions using the fitness function values.

Step 17: Increase the class counter by one. Go to Step 8 until all the solutions complete one iteration of the teaching-learning process.

Step 18: Go to Step 5 until the required number of iterations is completed.

Step 19: Select the best solution in the entire class as the optimal solution.

Psuedocode H III: ***Class Parallelized Teaching Learning based Optimization***

Step 1: Define the bounds of the decision variables, the number of decision variables (N_d) and the objective function (or fitness function for constrained optimization problems).

Step 2: Initialize the algorithmic parameters, such as class size N_P, and the number of iterations (N_G)

Step 3: Generate a class of learners randomly such that the marks (or decision variables) of each learner in every subject is within its specified domain. Evaluate the fitness of each learner in the class.

Step 4**:** Initialize the iteration counter as zero

Step 5: Increment the iteration counter by 1

Step 6: Determine the learning partner $S^{partner}$ for each l by randomly permuting the numbers $(1,2,..,N_P)$. Let $S^{partner,i}$ indicate the learner who is partnering with learner i

Step 7: Determine the best learner S^{best} in the class based on the value of the fitness function

Step 8: Generate the teaching factor T_F as a value of 1 or 2 for the whole class of size $N_P \times N_d$.

Step 9: Determine the average mark of the class in each subject $\left(S^{mean}\right)$

Step 10: Determine $3N_P$ potential learners using the following relation

$$T_j^{new,i} = S_j^i + r_j^i \left(S_j^{best} - T_{F,j}^i S_j^{mean}\right) \quad \forall j = 1,2,..,N_d, i = 1,2,..N_P$$
$$S_j^{1new,i} = S_j^i + r_j^i \left(S_j^i - S_j^{partner,i}\right) \quad \forall j = 1,2,..,N_d, i = 1,2,..N_P$$
$$S_j^{2new,i} = S_j^i + r_j^i \left(S_j^{partner,i} - S_j^i\right) \quad \forall j = 1,2,..,N_d, i = 1,2,..N_P$$

In this expression, $T_j^{new,i}$ is the mark of the potential solution i in the subject j , S_j^i is the mark in the subject j of the learner i, r_j^i is a random number between 0 and 1, S_j^{best} is the

mark in the subject j of S^{best} , S_j^{mean} is the average of the population in the subject j and $S_j^{partner,i}$ is the mark in the subject j of the learner partnering the current solution.

Step 11: Ensure that all the decision variables of the $3N_P$ potential learners are within their bounds. If lb_j and ub_j are the lower and upper bounds of the decision variable j, a solution (X) can be bounded as shown in the following equation

$$X_j = \begin{cases} \max\left(X_j, lb_j\right) \\ \min\left(X_j, ub_j\right) \end{cases} \quad \forall j = 1,..,N_d$$

Step 12: Evaluate the fitness of all the $3N_P$ solutions.

Step 13: Combine these $3N_P$ potential learners with the original class and determine the best N_P solutions from the $4N_P$ solutions using the fitness function values.

Step 14: Go to Step 5 until the required number of iterations is completed.

Step 15: Select the best solution in the entire class as the optimal solution.

Psuedocode H IV*: Phase Parallelized Teaching Learning based Optimization*

Step 1: Define the bounds of the decision variables, number of decision variables (N_d) and the objective function (or fitness function for constrained optimization problems).

Step 2: Initialize the algorithmic parameters, such as class size and the number of iterations

Step 3: Generate a class of learners randomly such that the marks (or decision variables) of each learner in every subject is within its specified domain. Evaluate the fitness of each learner in the class.

***Step 4*:** Initialize the iteration counter as zero

Step 5: Increment the iteration counter by 1

Step 6: Determine the best learner S^{best} in the class based on the value of the fitness function

Step 7: Generate the teaching factor T_F as a value of 1 or 2 for the whole class of size $N_P \times N_d$.

Step 8: Determine the average mark of the class in each subject $\left(S^{mean}\right)$

Step 9: Determine N_P potential new solutions of the class using the following expression,

$$T_j^{new,i} = S_j^i + r_j^i \left(S_j^{best} - T_{F,j}^i S_j^{mean}\right) \quad \forall j = 1,2,..,N_d, i = 1,2,..N_P .$$

In this expression, $T_j^{new,i}$ is the mark of the potential new solution i in the subject j, S_j^i is the mark in the subject j of learner i, r_j^i is a random number between 0 and 1, S_j^{best} is the mark in the subject j of S^{best}, S_j^{mean} is the average of the population in the subject j.

Step 10: Ensure that all the decision variables of the N_P potential learnres are within their bounds.

If B_j^l and B_j^u are the lower and upper bounds of the decision variable j, a solution (X) can be bounded as shown in the following equation

$$X_j = \begin{cases} \max\left(X_j, B_j^l\right) \\ \min\left(X_j, B_j^u\right) \end{cases} \qquad \forall j = 1,..,N_d$$

Step 11: Evaluate the fitness of all the N_P bounded solutions.

Step 12: Combine these N_P potential learners with the original class and determine the best N_P solutions from the $2N_P$ solutions using the fitness function values.

Step 13: Determine the learning partner $S^{partner}$ for each learner by randomly permuting the numbers $(1,2,..,N_P)$. Let $S^{partner,i}$ indicate the learner who is partnering with the learner i

Step 14: Generate $2N_P$ potential learners using the following relation, where $S_j^{partner,i}$ indicates the marks in the subject j of the learner who is partnering with the learner i.

$$S_j^{1new,i} = S_j^i + r_j^i\left(S_j^i - S_j^{partner,i}\right) \qquad \forall j = 1,2,..,N_d, i = 1,2,..N_P$$
$$S_j^{2new,i} = S_j^i + r_j^i\left(S_j^{partner,i} - S_j^i\right) \qquad \forall j = 1,2,..,N_d, i = 1,2,..N_P$$

Step 15: Ensure that all the decision variables of the $2N_P$ potential learners are in the bounds.

Step 16: Evaluate the fitness of the bounded $2N_P$ solutions.

Step 17: Combine these $2N_P$ potential learners with the original class and determine the best N_P solutions from the $3N_P$ solutions using the fitness function values.

Step 18: Go to Step 5 until the required number of iterations is completed.

Step 19: Select the best solution in the entire class as the optimal solution.

Table H1 Comparative analysis of the various parallel schemes

	sTLBO	**MTLBO**	**CTLBO**	**PTLBO**
Updating of class in every iteration	Twice for every member of the class	Once for every solution in the class (after completion of learner phase)	Once for every generation (after completion of learner phase)	Twice for every generation (once after teacher phase and after the learner phase)
No. of times updating	$2N_GN_P$	N_GN_P	N_G	$2N_G$
Sorting	Sorting of the objective value is not required	Once after the whole class completes both the teacher and learner phase	Once for every generation	Twice for every generation; One after the teacher phase and the other after the learner phase
No. of times sorting required	0	$N_GN_P - 1$ with a vector of size $N_P + 3$	$N_G - 1$ with a vector of size $4N_P$	N_G times with a vector of size $2N_P$ and N_G times with a vector of size $3N_P$
Calculation of population mean	N_GN_P	N_GN_P	N_G	N_G
No. of function evaluations	$N_P(2N_G+1)$	$N_P(3N_G+1)$	$N_P(3N_G+1)$	$N_P(3N_G+1)$
No. of learners generated per member in the learner phase	One	Two	Two	Two
Time required for function evaluation	$\left\lceil \frac{N_P}{N_R} \right\rceil + 2N_PN_G$	$\left\lceil \frac{N_P}{N_R} \right\rceil + \left\lceil \frac{3}{N_R} \right\rceil N_PN_G$	$\left\lceil \frac{N_P}{N_R} \right\rceil + \left\lceil \frac{3N_P}{N_R} \right\rceil N_G$	$\left\lceil \frac{N_P}{N_R} \right\rceil + \left(\left\lceil \frac{N_P}{N_R} \right\rceil + \left\lceil \frac{2N_P}{N_R} \right\rceil \right) N_G$

Fig. H1 shows the amount of computational time required to evaluate the objective function in sTLBO and all three parallelized strategies with respect to the population size. The number of generations and the number of resources is maintained constant at 100 and 6, respectively, with the computational time required for evaluating the objective function of a single member as 1 unit of time. As expected, the amount of computational time required increases with the increase in the class size, but it can also be noticed that the amount of computational time required for sTLBO is much higher than other three parallelized schemes. The computational time necessary for evaluating the objective function in MTLBO is lower than sTLBO and greater than CTLBO and PTLBO. Among CTLBO and PTLBO, the amount of computational time required by the phase parallelized scheme will be greater than or equal to the class parallelized scheme but can never be less than it.

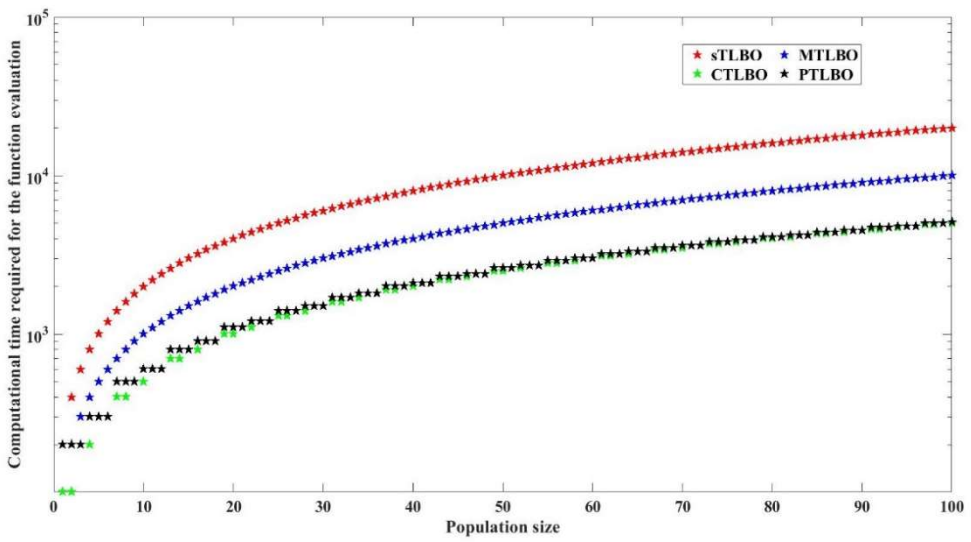


Fig. H1. Computational time required to evaluate the objective function with respect to N_P

Fig. H2 shows the variation of computational time versus the time required for a one-time evaluation of the objective function. It is found that this profile is similar toFig. 9.6, and similar observations can be inferred.

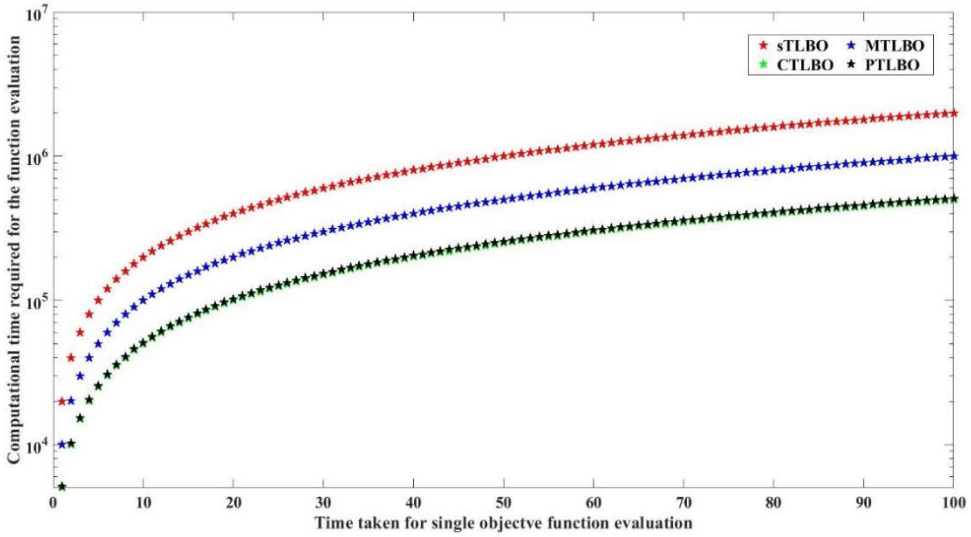


Fig. H2 Computational time required to evaluate the objective function with respect to NT

Appendix J: Additional results of parallel variants in classical benchmark functions

Table J1. Details of the classical benchmark function

ID	Name	N_d	Domain	Function
BF1	De Jong function	2	[-2.048, 2.048]	$-\left[3905.93-100\left(x_1^2-x_2\right)^2-\left(1-x_1\right)^2\right]$
BF2	Goldstein and Price function	2	[-2, 2]	$\left(\begin{array}{l}\left[1+\left(x_1+x_2+1\right)^2\left(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2\right)\right]\\ \left[30+\left(2x_1-3x_2\right)^2\left(18-32x_1+12x_1^2+48x_2-36x_1x_2+27x_2^2\right)\right]\end{array}\right)$
BF3	Martin and Gaddy function	2	[0, 10]	$\left(x_1-x_2\right)^2+\left[\left(x_1+x_2-10\right)/3\right]^2$
BF4	Rosenbrock function	30	[-30,30]	$\sum_{i=1}^{D}\left[100(x_i^2-x_{i+1})^2+\left(1-x_i\right)^2\right]$
BF5	Sphere function	30	[-100, 100]	$\sum_{i=1}^{D}x_i^2$
BF6	Powell badly scaled function	2	[-50, 50]	$-\left[\left(10x_1x_2-1\right)^2+\left[exp\left(-x_1\right)+exp\left(-x_2\right)-1.0001\right]^2\right]$
BF7	B2 function	2	[-50, 50]	$x_1^2+2x_2^2-0.3cos\left(3\pi x_1\right)-0.4cos\left(4\pi x_2\right)+0.7$
BF8	Booth function	2	[-50, 50]	$\left(x_1+2x_2-7\right)^2+\left(2x_1+x_2-5\right)^2$
BF9	Griewank function	30	[-50, 50]	$\frac{1}{4000}\sum_{i=1}^{D}x_i^2-\prod_{i=1}^{D}cos\left(\frac{x_i}{\sqrt{i}}\right)+1$
BF10	Rastrigin function	30	[-5.12, 5.12]	$\sum_{i=1}^{D}\left[x_i^2-10\,cos\left(2\pi x_i\right)+10\right]$
BF11	Ackley function	30	[-32,32]	$-20\,exp\left(-0.2\sqrt{\frac{1}{30}\sum_{i=1}^{D}x_i^2}\right)-exp\left(\frac{1}{30}\sum_{i=1}^{D}cos\,2\pi x_i\right)$
BF12	Step function	30	[-100,100]	$\sum_{i=1}^{D}\left[x_i-0.5\right]^2$
BF13	Schwefel function	30	[-500,500]	$-\sum_{i=1}^{D}\left(x_i\,sin\left(\sqrt{\lvert x_i\rvert}\right)\right)$
BF14	Penalised function	30	[-50, 50]	$f\left(x\right)=\sum_{i=1}^{D}u\left(x_i,10,100,4\right)+\frac{\pi}{D}\left[\begin{array}{l}10\,sin^2\left(\pi y_1\right)+\\ \sum_{i=1}^{D-1}\left(y_i-1\right)^2\left\{1+10\,sin^2\left(\pi y_{i+1}\right)\right\}+\\ \left(y_D-1\right)^2\end{array}\right]$ $y_i=1+\frac{1}{4}\left(x_i+1\right);$ $u\left(x_i,a,k,m\right)=\begin{cases}k\left(x_i-a\right)^m, & x_i>a,\\ 0, & -a\le x_i\le a,\\ k\left(-x_i-a\right)^m, & x_i<-a\end{cases}$

For all the profiles depicted in the second row of Fig. J1, the member TLBO is the fastest in determining a better solution in the initial few functional evaluations, but in many cases (Schwefel, Step, Penalised, and Rastrigin), stagnates to a suboptimal solution. The MTLBO determines the optimal solution for the Ackley function in less than 21000 functional evaluations, whereas the rest of the three algorithms require additional function evaluations.

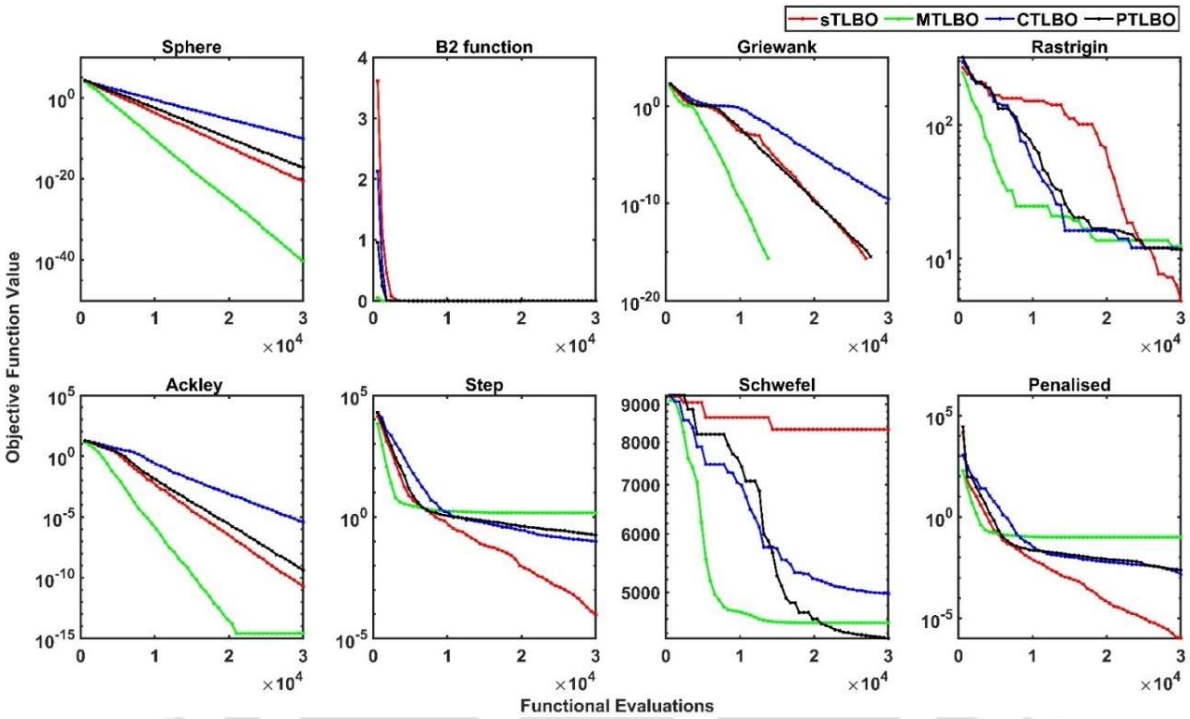


Fig. J1 Performance of algorithms on benchmark functions

Appendix K: Statistical analysis of parallel variants on CEC test suites and single level production planning problem

Table K1. Best error values obtained in each problem of CEC14 testsuite

	AVOA	MA	MFO	SCA	SHO	TGA	sTLBO	CTLBO	MTLBO	PTLBO
F1	1.22E+01	**9.59E-01**	2.95E+04	1.71E+06	3.87E+07	2.94E+06	3.60E+03	4.21E+02	6.20E+03	2.16E+02
F2	1.86E-02	**5.68E-14**	8.28E-03	1.76E+08	2.52E+09	2.54E+08	3.43E-01	6.61E-02	1.78E+05	1.07E+00
F3	5.68E+01	**6.82E-13**	6.86E+01	9.78E+02	1.48E+04	5.21E+03	7.65E+01	1.51E+00	5.04E+01	2.70E+00
F4	5.04E-02	**0.00E+00**	1.73E-02	2.04E+01	3.29E+02	4.49E+01	1.58E-02	1.07E-02	9.25E-01	1.89E-02
F5	2.00E+01	2.00E+01	2.00E+01	2.02E+01	2.02E+01	2.00E+01	1.91E+01	**1.65E+00**	1.98E+01	2.00E+01
F6	1.38E+00	1.14E+00	1.25E-01	4.00E+00	8.80E+00	5.52E+00	**2.64E-04**	5.59E-04	5.71E-01	6.34E-03
F7	3.69E-02	2.71E-02	3.94E-02	3.38E+00	5.36E+01	4.97E+00	**5.68E-13**	7.40E-03	1.72E-01	7.40E-03
F8	1.99E+00	**0.00E+00**	2.98E+00	1.66E+01	7.78E+01	1.53E+01	9.95E-01	9.95E-01	9.97E-01	**0.00E+00**
F9	8.95E+00	3.98E+00	9.95E+00	1.57E+01	6.16E+01	2.10E+01	2.23E+00	**1.99E+00**	2.51E+00	2.98E+00
F10	1.35E+01	3.76E-01	4.02E+01	3.98E+02	1.41E+03	2.71E+02	3.69E+00	6.89E+00	7.40E+00	**2.50E-01**
F11	2.58E+02	**6.89E+00**	1.76E+02	6.54E+02	1.56E+03	8.05E+02	1.64E+01	2.53E+01	2.40E+01	3.85E+01
F12	**7.79E-03**	2.26E-02	3.04E-02	5.81E-01	8.34E-01	5.33E-01	3.80E-01	6.67E-02	3.53E-02	6.55E-02
F13	1.17E-01	1.04E-01	1.26E-01	3.01E-01	1.82E+00	2.70E-01	7.72E-02	2.60E-02	4.61E-02	**1.51E-02**
F14	**1.08E-01**	1.96E-01	1.64E-01	3.58E-01	1.68E+01	3.85E-01	1.20E-01	2.09E-01	2.05E-01	1.74E-01
F15	1.26E+00	4.94E-01	4.74E-01	4.31E+00	1.07E+02	7.84E+00	8.19E-01	**3.26E-01**	6.26E-01	4.26E-01
F16	1.58E+00	1.84E+00	2.31E+00	2.53E+00	3.42E+00	2.28E+00	1.31E+00	9.67E-01	1.07E+00	**7.82E-01**
F17	5.87E+01	**3.48E+01**	8.24E+02	5.36E+03	1.95E+05	9.38E+03	6.82E+02	6.74E+01	1.11E+02	5.12E+01
F18	4.23E+01	1.76E+01	8.01E+01	5.91E+02	1.54E+04	2.72E+02	8.08E+01	2.50E+01	1.45E+02	**7.17E+00**
F19	9.07E-01	9.10E-01	1.06E+00	3.60E+00	9.23E+00	4.62E+00	1.53E-01	6.81E-02	1.73E-01	**4.51E-02**
F20	8.39E+01	4.85E+00	4.19E+00	1.32E+02	5.15E+03	4.22E+01	3.47E+01	**2.11E+00**	4.34E+00	8.32E+00
F21	3.20E+02	1.00E+00	8.99E+01	1.43E+03	7.54E+03	1.03E+03	4.58E+01	9.96E-01	4.90E+01	**9.43E-01**
F22	6.34E+00	1.99E+01	**6.70E-03**	3.06E+01	1.53E+02	2.58E+01	9.13E-01	2.67E-01	2.53E-01	2.52E-02
F23	**2.00E+02**	3.29E+02	3.29E+02	3.34E+02	**2.00E+02**	**2.00E+02**	3.29E+02	3.29E+02	3.29E+02	3.29E+02
F24	1.23E+02	1.16E+02	1.14E+02	1.34E+02	2.00E+02	1.49E+02	1.00E+02	**1.00E+02**	1.09E+02	1.00E+02
F25	1.41E+02	1.46E+02	1.35E+02	1.56E+02	2.00E+02	1.68E+02	**1.07E+02**	1.12E+02	1.22E+02	1.14E+02
F26	1.00E+02	1.00E+02	1.00E+02	1.00E+02	1.01E+02	1.00E+02	1.00E+02	1.00E+02	1.00E+02	**1.00E+02**
F27	4.28E+00	3.68E+00	3.57E+00	7.70E+00	2.00E+02	6.13E+00	1.77E+00	1.77E+00	3.06E+00	**1.18E+00**
F28	2.00E+02	**1.00E+02**	3.69E+02	4.04E+02	2.00E+02	4.11E+02	3.69E+02	3.60E+02	3.81E+02	3.69E+02
F29	**2.00E+02**	2.21E+02	3.44E+02	6.45E+02	**2.00E+02**	3.98E+02	3.21E+02	3.01E+02	2.73E+02	3.02E+02
F30	5.95E+02	6.33E+02	4.59E+02	7.58E+02	**2.00E+02**	1.09E+03	4.60E+02	4.60E+02	4.73E+02	4.60E+02

Table K2. Worst error values obtained in each problem of CEC14 testsuite

	AVOA	MA	MFO	SCA	SHO	TGA	sTLBO	CTLBO	MTLBO	PTLBO
F1	2.35E+05	**8.24E+03**	2.32E+07	1.23E+07	4.61E+08	2.67E+07	1.74E+05	1.27E+05	1.19E+07	1.45E+05
F2	1.18E+04	**8.31E+02**	1.18E+04	1.42E+09	1.51E+10	1.23E+09	2.00E+03	4.07E+03	1.03E+09	5.29E+03
F3	2.03E+03	1.40E+04	2.88E+04	9.76E+03	1.75E+06	1.53E+04	2.57E+03	**1.50E+03**	4.79E+03	1.52E+03
F4	6.76E+01	**3.48E+01**	1.26E+02	7.22E+01	3.83E+03	1.61E+02	3.48E+01	3.48E+01	8.71E+01	3.48E+01
F5	**2.03E+01**	2.04E+01	2.06E+01	2.05E+01	2.05E+01	2.04E+01	2.05E+01	2.03E+01	2.05E+01	2.04E+01
F6	8.56E+00	5.27E+00	5.93E+00	9.03E+00	1.25E+01	8.86E+00	**2.76E+00**	3.46E+00	4.41E+00	4.57E+00
F7	1.37E+00	9.06E-01	4.84E+00	1.80E+01	2.24E+02	3.26E+01	**1.24E-01**	3.15E-01	1.42E+01	1.97E-01
F8	2.09E+01	**9.95E-01**	3.79E+01	5.19E+01	1.31E+02	4.37E+01	1.19E+01	1.19E+01	2.21E+01	1.19E+01
F9	6.47E+01	1.89E+01	4.54E+01	5.67E+01	1.22E+02	5.08E+01	1.51E+01	1.69E+01	2.69E+01	**1.39E+01**
F10	3.87E+02	**3.59E+02**	7.90E+02	1.22E+03	2.30E+03	1.05E+03	8.60E+02	5.48E+02	3.77E+02	4.26E+02
F11	1.53E+03	1.29E+03	1.71E+03	1.58E+03	2.75E+03	1.56E+03	1.30E+03	**9.59E+02**	9.86E+02	1.10E+03
F12	7.72E-01	1.25E+00	**7.14E-01**	1.61E+00	3.78E+00	1.36E+00	1.57E+00	7.96E-01	1.21E+00	1.23E+00
F13	8.69E-01	4.29E-01	9.10E-01	8.46E-01	4.78E+00	1.02E+00	**2.62E-01**	3.33E-01	1.81E+00	2.94E-01
F14	8.91E-01	8.24E-01	1.03E+00	1.81E+00	6.54E+01	8.21E+00	**4.23E-01**	4.86E-01	3.58E+00	4.73E-01
F15	8.04E+00	2.69E+00	4.70E+00	9.32E+00	6.71E+04	7.87E+01	2.30E+00	2.18E+00	1.41E+01	**2.05E+00**
F16	3.68E+00	3.46E+00	4.02E+00	3.70E+00	4.34E+00	3.39E+00	3.13E+00	2.94E+00	3.16E+00	**2.85E+00**
F17	1.48E+04	2.62E+04	2.94E+05	4.67E+04	1.15E+07	4.08E+05	**4.09E+03**	1.79E+04	4.24E+04	1.80E+04
F18	3.56E+04	2.01E+04	3.73E+04	6.71E+04	8.74E+07	3.01E+05	**7.67E+03**	2.01E+04	3.53E+04	2.56E+04
F19	4.42E+00	3.27E+00	4.41E+00	6.59E+00	1.16E+02	8.17E+00	1.89E+00	**1.30E+00**	3.40E+00	1.87E+00
F20	1.03E+04	1.13E+04	6.27E+04	9.57E+03	6.98E+06	3.84E+03	**4.64E+02**	4.97E+03	8.86E+03	6.61E+03
F21	1.87E+04	1.02E+04	3.02E+04	2.06E+04	4.30E+06	1.20E+04	**4.68E+02**	1.25E+03	6.02E+03	1.31E+03
F22	1.65E+02	1.78E+02	2.42E+02	1.96E+02	8.08E+02	9.99E+01	**2.88E+01**	3.72E+01	1.63E+02	5.87E+01
F23	**2.00E+02**	3.29E+02	3.42E+02	3.51E+02	**2.00E+02**	**2.00E+02**	3.29E+02	3.29E+02	3.35E+02	3.29E+02
F24	2.00E+02	2.05E+02	1.65E+02	1.61E+02	2.00E+02	1.85E+02	**1.20E+02**	1.27E+02	2.03E+02	1.27E+02
F25	2.00E+02	2.02E+02	2.04E+02	2.04E+02	2.00E+02	2.00E+02	**1.97E+02**	2.02E+02	2.04E+02	2.03E+02
F26	1.01E+02	1.00E+02	1.01E+02	1.01E+02	1.06E+02	1.01E+02	**1.00E+02**	1.00E+02	1.01E+02	1.00E+02
F27	**2.00E+02**	4.33E+02	4.47E+02	4.11E+02	**2.00E+02**	4.24E+02	4.00E+02	4.01E+02	5.27E+02	4.01E+02
F28	**2.00E+02**	9.53E+02	4.14E+02	5.84E+02	**2.00E+02**	5.51E+02	4.95E+02	4.91E+02	8.12E+02	5.05E+02
F29	2.12E+03	2.10E+06	1.60E+03	9.58E+03	**2.00E+02**	4.11E+03	1.25E+03	1.51E+03	3.12E+06	1.51E+03
F30	2.62E+03	2.18E+03	1.31E+03	3.47E+03	**2.00E+02**	3.18E+03	7.50E+02	1.12E+03	2.01E+03	8.42E+02

Table K3. Mean error values obtained in each problem of CEC14 testsuite

	AVOA	MA	MFO	SCA	SHO	TGA	sTLBO	CTLBO	MTLBO	PTLBO
F1	1.11E+05	**7.48E+02**	1.37E+06	5.83E+06	1.59E+08	1.03E+07	3.75E+04	2.73E+04	1.31E+06	4.36E+04
F2	2.79E+03	**4.21E+01**	7.17E+03	5.75E+08	9.21E+09	5.79E+08	4.70E+02	7.03E+02	1.24E+08	9.32E+02
F3	6.36E+02	3.05E+03	1.15E+04	2.87E+03	1.66E+05	1.17E+04	9.06E+02	**4.58E+02**	8.61E+02	5.06E+02
F4	2.13E+01	2.41E+01	3.07E+01	4.88E+01	1.64E+03	8.82E+01	1.73E+01	2.12E+01	4.55E+01	**1.64E+01**
F5	2.00E+01	2.02E+01	2.01E+01	2.03E+01	2.04E+01	2.02E+01	2.03E+01	**1.98E+01**	2.02E+01	2.02E+01
F6	5.33E+00	3.61E+00	3.18E+00	6.41E+00	1.03E+01	7.57E+00	**4.26E-01**	5.10E-01	2.40E+00	7.14E-01
F7	3.68E-01	3.24E-01	5.10E-01	8.44E+00	1.39E+02	1.60E+01	**5.06E-02**	9.59E-02	3.52E+00	7.68E-02
F8	8.29E+00	**1.55E-01**	1.66E+01	3.48E+01	1.05E+02	3.19E+01	5.33E+00	4.76E+00	9.47E+00	5.64E+00
F9	2.68E+01	9.42E+00	2.52E+01	3.94E+01	8.86E+01	3.90E+01	**6.70E+00**	6.80E+00	1.09E+01	6.85E+00
F10	**1.07E+02**	1.92E+02	3.15E+02	8.86E+02	1.94E+03	7.43E+02	3.08E+02	1.49E+02	1.57E+02	1.22E+02
F11	6.93E+02	6.08E+02	7.94E+02	1.18E+03	2.15E+03	1.26E+03	8.63E+02	4.06E+02	4.53E+02	**3.95E+02**
F12	3.15E-01	2.58E-01	**2.51E-01**	1.14E+00	1.52E+00	9.89E-01	1.04E+00	2.73E-01	3.84E-01	3.27E-01
F13	4.20E-01	2.46E-01	3.01E-01	5.44E-01	3.12E+00	4.99E-01	1.42E-01	**1.39E-01**	3.24E-01	1.46E-01
F14	3.40E-01	3.60E-01	3.61E-01	8.35E-01	3.85E+01	3.77E+00	**2.51E-01**	3.34E-01	5.51E-01	3.49E-01
F15	3.44E+00	1.23E+00	1.51E+00	6.87E+00	1.15E+04	2.16E+01	1.41E+00	**9.55E-01**	2.06E+00	9.99E-01
F16	2.90E+00	2.74E+00	3.11E+00	3.27E+00	3.83E+00	3.09E+00	2.42E+00	2.14E+00	2.33E+00	**2.04E+00**
F17	5.14E+03	1.91E+03	2.26E+04	2.01E+04	2.83E+06	1.11E+05	**1.57E+03**	5.08E+03	7.26E+03	5.98E+03
F18	9.09E+03	**7.91E+02**	1.45E+04	1.42E+04	1.60E+07	3.60E+04	1.16E+03	4.48E+03	9.53E+03	5.85E+03
F19	2.84E+00	1.82E+00	2.19E+00	4.70E+00	3.95E+01	6.41E+00	1.07E+00	**7.43E-01**	1.58E+00	8.25E-01
F20	2.85E+03	2.69E+03	8.57E+03	1.42E+03	7.79E+05	7.66E+02	**1.54E+02**	6.22E+02	2.46E+03	1.16E+03
F21	6.58E+03	1.29E+03	6.15E+03	6.06E+03	1.31E+06	3.62E+03	**2.32E+02**	3.26E+02	9.04E+02	2.84E+02
F22	3.54E+01	1.26E+02	4.11E+01	5.50E+01	4.27E+02	4.48E+01	1.87E+01	1.15E+01	2.78E+01	**8.91E+00**
F23	**2.00E+02**	3.29E+02	3.31E+02	3.40E+02	**2.00E+02**	**2.00E+02**	3.29E+02	3.29E+02	3.31E+02	3.29E+02
F24	1.67E+02	1.48E+02	1.34E+02	1.48E+02	2.00E+02	1.65E+02	**1.11E+02**	1.11E+02	1.30E+02	1.13E+02
F25	1.93E+02	1.96E+02	1.93E+02	1.94E+02	2.00E+02	1.95E+02	**1.31E+02**	1.64E+02	1.83E+02	1.67E+02
F26	1.00E+02	1.00E+02	1.00E+02	1.01E+02	1.03E+02	1.00E+02	1.00E+02	1.00E+02	1.00E+02	**1.00E+02**
F27	1.77E+02	3.32E+02	2.57E+02	3.12E+02	2.00E+02	**9.44E+01**	1.40E+02	1.52E+02	3.17E+02	1.78E+02
F28	**2.00E+02**	5.71E+02	3.80E+02	4.56E+02	**2.00E+02**	4.31E+02	3.99E+02	3.96E+02	4.81E+02	4.14E+02
F29	7.16E+02	4.15E+04	9.08E+02	3.35E+03	**2.00E+02**	1.46E+03	6.35E+02	5.95E+02	3.10E+05	6.14E+02
F30	1.21E+03	1.40E+03	6.07E+02	1.38E+03	**2.00E+02**	1.78E+03	5.29E+02	5.84E+02	8.56E+02	5.67E+02

Table K4. Median error values obtained in each problem of CEC14 testsuite

	AVOA	MA	MFO	SCA	SHO	TGA	sTLBO	CTLBO	MTLBO	PTLBO
F1	1.25E+05	**1.30E+02**	2.51E+05	5.13E+06	1.36E+08	9.75E+06	2.36E+04	1.68E+04	5.19E+05	2.98E+04
F2	7.40E+02	**6.65E-01**	9.67E+03	5.47E+08	8.81E+09	5.24E+08	1.94E+02	4.42E+02	4.20E+07	4.59E+02
F3	6.00E+02	2.44E+03	1.17E+04	2.30E+03	7.60E+04	1.20E+04	7.73E+02	**2.98E+02**	6.94E+02	4.03E+02
F4	3.48E+01	3.48E+01	3.48E+01	4.90E+01	1.34E+03	8.86E+01	**4.34E+00**	3.48E+01	4.32E+01	4.53E+00
F5	**2.00E+01**	2.03E+01	2.01E+01	2.03E+01	2.04E+01	2.02E+01	2.04E+01	2.01E+01	2.02E+01	2.01E+01
F6	5.55E+00	3.59E+00	3.53E+00	6.32E+00	1.03E+01	7.53E+00	**1.22E-01**	2.32E-01	2.20E+00	2.80E-01
F7	2.88E-01	2.24E-01	1.45E-01	7.98E+00	1.36E+02	1.59E+01	**4.67E-02**	8.11E-02	2.26E+00	7.14E-02
F8	7.96E+00	**0.00E+00**	1.52E+01	3.51E+01	1.05E+02	3.22E+01	4.97E+00	3.98E+00	8.97E+00	4.97E+00
F9	2.69E+01	8.95E+00	2.42E+01	3.98E+01	8.79E+01	3.90E+01	6.88E+00	**5.97E+00**	1.01E+01	5.97E+00
F10	**7.90E+01**	2.41E+02	2.80E+02	8.88E+02	1.96E+03	7.48E+02	2.60E+02	1.59E+02	1.60E+02	1.29E+02
F11	6.84E+02	5.80E+02	7.75E+02	1.18E+03	2.14E+03	1.29E+03	8.94E+02	4.22E+02	4.67E+02	**3.55E+02**
F12	2.82E-01	**1.52E-01**	2.01E-01	1.14E+00	1.28E+00	1.01E+00	1.08E+00	2.41E-01	3.17E-01	2.52E-01
F13	4.01E-01	2.42E-01	2.91E-01	5.54E-01	3.10E+00	4.85E-01	1.36E-01	**1.30E-01**	2.88E-01	1.44E-01
F14	3.14E-01	3.42E-01	3.01E-01	8.21E-01	3.90E+01	3.56E+00	**2.50E-01**	3.43E-01	3.48E-01	3.47E-01
F15	3.17E+00	1.14E+00	1.41E+00	6.89E+00	7.50E+03	1.51E+01	1.36E+00	**8.99E-01**	1.40E+00	9.19E-01
F16	2.99E+00	2.73E+00	3.13E+00	3.28E+00	3.80E+00	3.12E+00	2.45E+00	2.18E+00	2.33E+00	**2.09E+00**
F17	4.92E+03	**5.32E+02**	5.17E+03	1.56E+04	2.09E+06	6.27E+04	1.29E+03	2.46E+03	4.02E+03	2.90E+03
F18	6.42E+03	**1.34E+02**	6.47E+03	1.17E+04	8.76E+06	1.40E+04	7.80E+02	2.59E+03	6.46E+03	2.88E+03
F19	3.06E+00	1.69E+00	2.24E+00	4.67E+00	3.57E+01	6.46E+00	1.19E+00	1.04E+00	1.54E+00	**1.02E+00**
F20	1.79E+03	4.11E+02	4.80E+03	4.22E+02	1.97E+05	3.47E+02	**1.27E+02**	2.23E+02	1.71E+03	5.37E+02
F21	4.92E+03	1.70E+02	2.04E+03	5.20E+03	8.03E+05	3.39E+03	2.31E+02	2.43E+02	5.72E+02	**1.44E+02**
F22	2.35E+01	1.41E+02	3.71E+01	5.47E+01	4.12E+02	4.07E+01	2.19E+01	2.28E+00	2.01E+01	**8.07E-01**
F23	**2.00E+02**	3.29E+02	3.29E+02	3.40E+02	**2.00E+02**	**2.00E+02**	3.29E+02	3.29E+02	3.31E+02	3.29E+02
F24	1.54E+02	1.45E+02	1.33E+02	1.48E+02	2.00E+02	1.65E+02	**1.11E+02**	1.12E+02	1.23E+02	1.13E+02
F25	2.00E+02	2.00E+02	2.02E+02	2.02E+02	2.00E+02	2.00E+02	**1.26E+02**	1.51E+02	2.00E+02	1.87E+02
F26	1.00E+02	1.00E+02	1.00E+02	1.01E+02	1.03E+02	1.00E+02	1.00E+02	1.00E+02	1.00E+02	**1.00E+02**
F27	2.00E+02	3.94E+02	4.00E+02	4.03E+02	2.00E+02	6.14E+01	4.36E+00	**3.99E+00**	3.65E+02	4.72E+00
F28	**2.00E+02**	5.84E+02	3.79E+02	4.39E+02	**2.00E+02**	4.30E+02	3.81E+02	3.81E+02	4.81E+02	3.81E+02
F29	6.23E+02	2.71E+02	8.88E+02	2.69E+03	**2.00E+02**	1.36E+03	6.04E+02	5.83E+02	6.55E+02	5.55E+02
F30	1.06E+03	1.45E+03	5.42E+02	1.25E+03	**2.00E+02**	1.60E+03	5.03E+02	5.77E+02	7.41E+02	5.70E+02

Table J5. Results of Wilcoxon signed-rank test for the CEC 2020 test suite

Algorithm	CTLBO			MTLBO			PTLBO		
	R+	R−	p-value	R+	R−	p-value	R+	R−	p-value
AVOA	312.0	**429.0**	**2.5E-02**	**576.0**	165.0	**2.9E-03**	254.0	**487.0**	**4.1E-02**
MA	308.0	433.0	1.2E-01	**586.0**	155.0	**3.0E-03**	320.0	421.0	1.7E-01
MFO	70.0	**671.0**	**1.4E-06**	339.5	401.5	4.5E-01	122.0	**619.0**	**6.4E-07**
SHO	38.0	**703.0**	**1.1E-07**	11.0	**730.0**	**1.9E-07**	38.0	**703.0**	**1.1E-07**
TGA	51.0	**690.0**	**3.3E-07**	148.0	**593.0**	**1.3E-03**	51.0	**690.0**	**3.3E-07**
IMODE	**739.0**	2.0	**1.3E-07**	**741.0**	0.0	**7.7E-08**	**738.0**	3.0	**9.8E-08**
APGSK	**740.0**	1.0	**8.4E-08**	**741.0**	0.0	**7.7E-08**	**740.0**	1.0	**8.4E-08**
j2020	**738.0**	3.0	**9.8E-08**	**740.0**	1.0	**8.4E-08**	**738.0**	3.0	**9.8E-08**

Table K5. Statistical results obtained on solving single level production planning problem (Chauhan & Kotecha, 2020)

		AVOA (2021)	MA (2020)	MFO (2015)	SHO (2017)	TGA (2018)	IMODE (2020)	AGSK (2020)	Proposed variants		
									CTLBO	MTLBO	PTLBO
Case 1	B	367.55	–	561.82	0	–	250.22	–	**653.55**	573.16	616.71
	M	175.45	–	446.72	0	–	–	–	481.06	351.29	**500.41**
	Md	167.20	–	450.14	0	–	62.27	–	477.92	351.30	**514.14**
Case 2	B	493.55	–	661.13	0	–	246.39	–	667.05	559.34	**692.65**
	M	182.66	–	491.74	0	–	–	–	**541.96**	388.38	524.75
	Md	167.20	–	506.85	0	–	98.17	–	**542.35**	396.34	535.80
Case 3	B	431.05	–	776.25	0	–	343.49	–	782.83	659.04	**813.80**
	M	246.69	–	625.94	0	–	–	–	605.40	423.82	**642.38**
	Md	275.45	–	650.76	0	–	–	–	614.27	423.29	**677.07**
Case 4	B	778.65	–	**1006.59**	0	–	417.20	–	992.90	804.98	936.69
	M	321.92	–	**800.13**	0	–	–	–	768.10	599.50	772.98
	Md	322.00	–	**809.92**	0	–	–	–	769.40	582.50	757.85
Case 5	B	531.80	–	587.00	0	–	347.63	–	601.15	**699.31**	661.55
	M	196.65	–	438.90	0	–	254.92	–	478.26	380.31	**508.97**
	Md	183.60	–	476.94	0	–	246.46	–	477.35	373.94	**523.62**
Case 6	B	578.95	–	**763.28**	0	–	405.26	–	749.75	624.32	745.35
	M	215.04	–	**588.90**	0	–	–	–	556.33	418.73	576.79
	Md	208.00	–	**597.40**	0	–	238.22	–	548.83	440.20	594.55
Case 7	B	635.30	–	861.85	0	–	530.42	–	856.60	882.15	**943.21**
	M	275.15	–	**711.38**	0	–	418.58	–	656.45	541.44	667.27
	Md	248.80	–	**714.60**	0	–	420.20	–	645.70	540.61	672.75
Case 8	B	778.65	–	1118.10	0	–	790.12	–	1057.36	1067.78	**1156.76**
	M	406.15	–	910.34	0	–	505.90	–	906.37	770.70	**940.53**
	Md	426.50	–	916.67	0	–	504.45	–	903.55	740.09	**960.95**

'B' indicates Best value, 'M' indicates mean value, 'Md' indicates median value and '-' represents an infeasible value corresponding to the statistic measure

Appendix L: Additional results of MOITLBO variants

The non-dominated solutions determined by variant I in all runs for UF1-UF6 are provided in Fig L1 and the same for UF7-UF10 in Fig. L2.

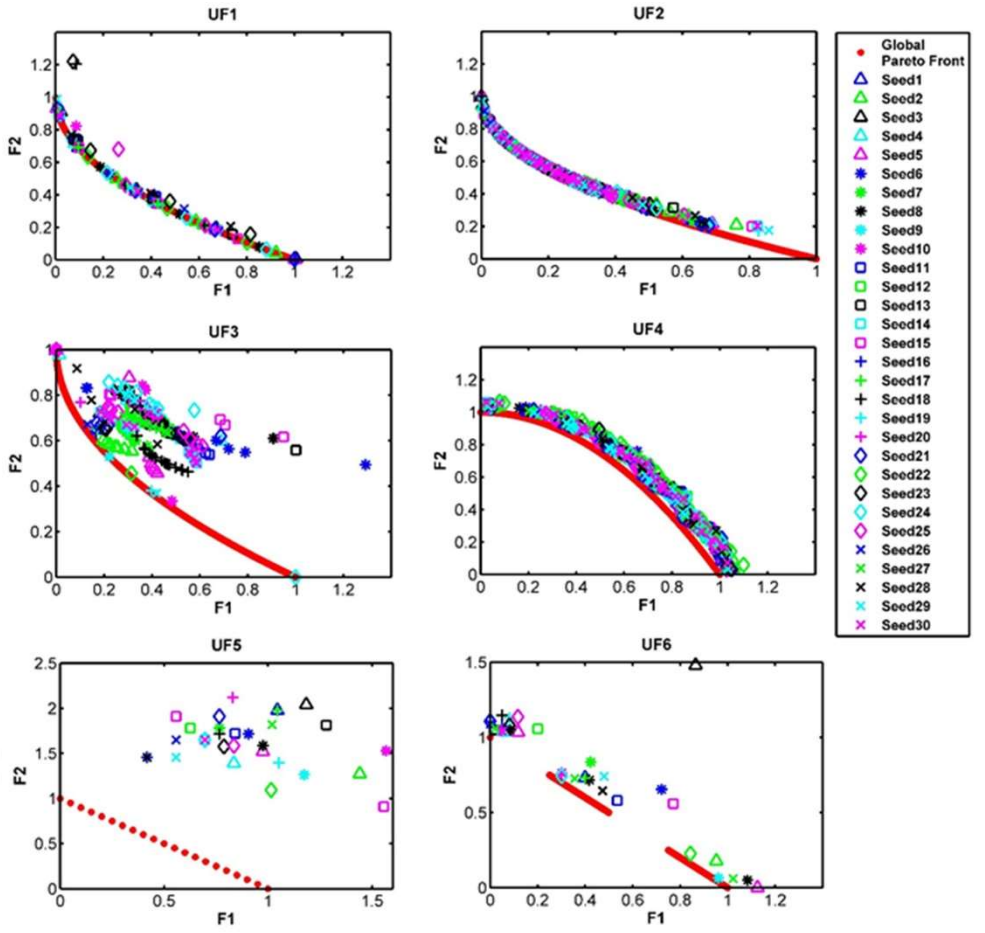


Fig. L1 Pareto fronts for UF1 – UF6 obtained in all 30 runs using Variant I

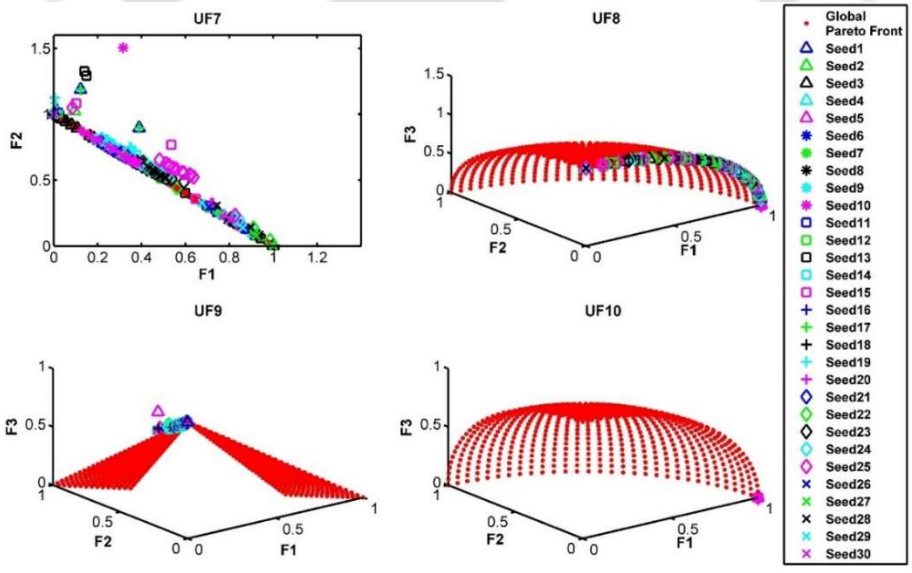


Fig. L2 Pareto fronts for UF7 – UF10 obtained in all 30 runs using Variant I

The non-dominated solutions determined by variant II in all runs for UF1-UF6 are provided in Fig L3 and the same for UF7-UF10 in Fig. L4.

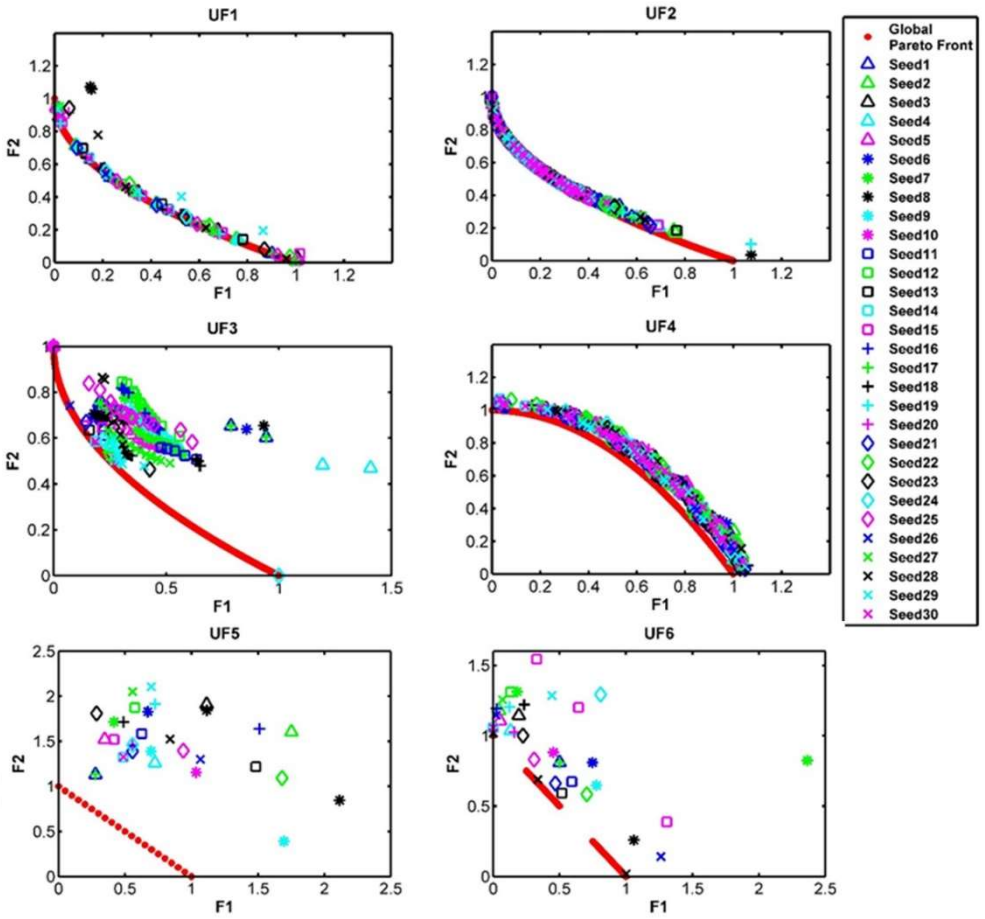


Fig. L3 Pareto fronts for UF1 – UF6 obtained in all 30 runs using Variant II

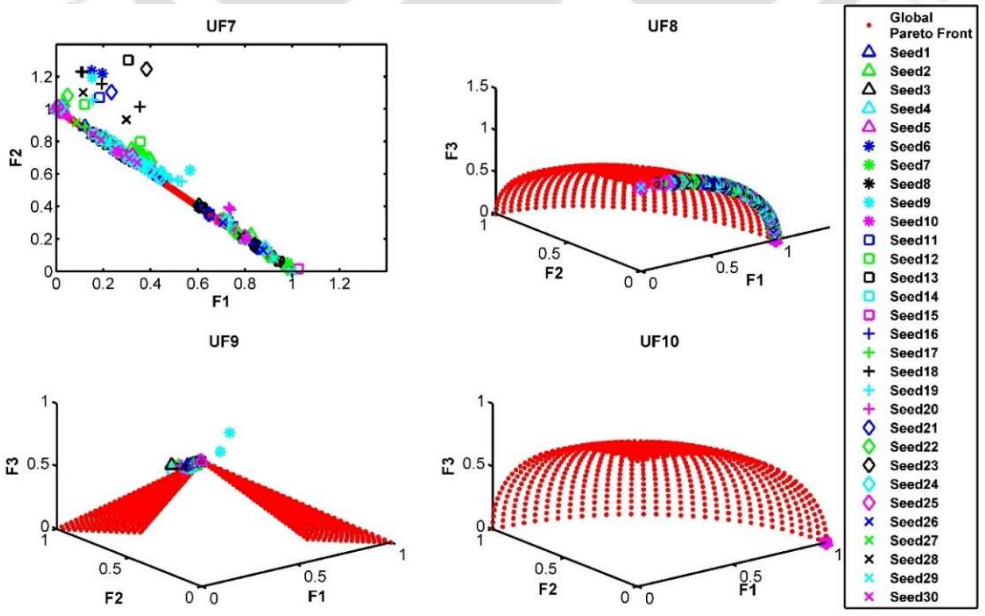


Fig. L4 Pareto fronts for UF7 – UF10 obtained in all 30 runs using Variant II

Publications

Peer reviewed journal articles from thesis

[1] **Kommadath, R.,** D. Maharana, and P. Kotecha, *An effective strategy for solving single and multi-unit production planning models with unique process constraints using metaheuristic techniques.* Expert Systems with Applications, 2023. **224**: p. 119813. DOI: https://doi.org/10.1016/j.eswa.2023.119813

[2] **Kommadath, R.,** D. Maharana, and P. Kotecha, *A metaheuristic-based efficient strategy for multi-unit production planning with unique process constraints.* Applied Soft Computing, 2023. **134**: p. 109871. DOI: https://doi.org/10.1016/j.asoc.2022.109871

[3] **Kommadath, R.,** D. Maharana, and P. Kotecha, *Efficient scheduling of jobs on dissimilar parallel machines using heuristic assisted metaheuristic techniques.* Chemical Engineering Research and Design, 2022. **188**: p. 916-934. DOI: https://doi.org/10.1016/j.cherd.2022.10.011.

[4] **Kommadath, R**., D. Maharana, R. Anandalakshmi, and P. Kotecha, *Multi-objective scheduling in the vegetable processing and packaging facility using metaheuristic based framework.* Food and Bioproducts Processing, 2023. **137**: p. 1-19. DOI: https://doi.org/10.1016/j.fbp.2022.10.005

[5] **Kommadath, R**., D. Maharana, C. Sivadurgaprasad, and P. Kotecha, *Parallel computing strategies for Sanitized Teaching Learning Based Optimization.* Journal of Computational Science, 2022. **63**: p. 101766. DOI: https://doi.org/10.1016/j.jocs.2022.101766

[6] Chinta, S., R. **Kommadath,** and P. Kotecha, *A note on multi-objective improved teaching–learning based optimization algorithm (MO-ITLBO).* Information Sciences, 2016. **373**: p. 337-350. DOI: https://doi.org/10.1016/j.ins.2016.08.061

[7] **Kommadath, R**., D. Maharana, P. Kotecha, and R. Anandalakshmi, *An efficient optimization strategy for vapor compression–absorption-based cascaded refrigeration system using various evolutionary techniques.*, International Journal of Green Energy, 2023 p. 1-16. DOI: https://doi.org/10.1080/15435075.2023.2244049

[8] **Kommadath, R.,** D. Maharana, and P. Kotecha, *Multiobjective Phase-wise Teaching Learning Based Optimization with no wait time heuristic for Job Shop Scheduling Problem*, Concurrency and Computation: Practice and Experience (under review)

[9] **Kommadath, R.,** D. Maharana, and P. Kotecha, *Multi-objective production planning model facilitating multiple processing units solved using metaheuristic techniques,* Chemical Engineering Research and Design (under review)

Peer reviewed book chapter from thesis (Scopus indexed)

[1] **Kommadath, R**. and P. Kotecha, *Scheduling of Jobs on Dissimilar Parallel Machine Using Computational Intelligence Algorithms*, in Nature-Inspired Methods for Metaheuristics Optimization: Algorithms and Applications in Science and Engineering, F. Bennis and R.K. Bhattacharjya, Editors. 2020, Springer International Publishing: Cham. p. 441-464. DOI: https://doi.org/10.1007/978-3-030-26458-1_24

Peer reviewed conference proceedings from thesis

[1] **Kommadath, R**., A.K. Saini, and P. Kotecha. *Production Scheduling on Multiple Parallel Machines Using Recent Metaheuristic Techniques*. in *Smart Trends in Computing and Communications*. 2023. Singapore: Springer Nature Singapore. https://doi.org/10.1007/978-981-99-0769-4_63

[2] **Kommadath, R**., M.S. Nagarag, D. Maharana, and P. Kotecha, *Efficient Bounding Strategy for CEC 2020 Winner Algorithms in Solving Production Planning Problems*. in *Smart Trends in Computing and Communications*. 2023. Singapore: Springer Nature Singapore. https://doi.org/10.1007/978-981-99-0838-7_16

[3] **Kommadath, R**., A.K. Saini, and P. Kotecha. *Production Planning in Process Industries using CEC 2021 Winning Algorithms*. in *2022 International Conference on Automation, Computing and Renewable Systems (ICACRS)*. 2022. https://doi.org/10.1109/ICACRS55517.2022.10029143

[4] **Kommadath, R.,** B. Ramchandani, M.S. Nagaraj, and P. Kotecha, *Performance Evaluation of Recently Proposed Metaheuristics Algorithms on Solving Job Shop Scheduling Problem*. in *2022 International Conference on Automation, Computing and Renewable Systems (ICACRS)*. 2022. https://doi.org/10.1109/ICACRS55517.2022.10029066

[5] **Kommadath, R**. and P. Kotecha. *Evaluation of Teaching Learning Based Optimization with Focused Learning on Expensive Optimization Problems (CEC2017)*. in *Smart Innovations in Communication and Computational Sciences*. 2019. Singapore: Springer Singapore. https://doi.org/10.1007/978-981-10-8968-8_37

[6] **Kommadath, R**. and P. Kotecha. *Teaching Learning Based Optimization with focused learning and its performance on CEC2017 functions*. in *2017 IEEE Congress on Evolutionary Computation (CEC)*. 2017. https://doi.org/10.1109/CEC.2017.7969595

[7] **Kommadath, R.,** J. Dondeti, and P. Kotecha, *Benchmarking JAYA and Sine Cosine Algorithm on Real Parameter Bound Constrained Single Objective Optimization Problems (CEC2016)*, in *Proceedings of the 2017 International Conference on Intelligent Systems, Metaheuristics \& Swarm Intelligence*. 2017, Association for Computing Machinery. p. 31–34. https://doi.org/10.1145/3059336.3059363

[8] R. Kommadath and P. Kotecha, *Optimization of stirling engine systems using single phase multi-group teaching learning based optimization and genetic algorithm,* Smart Innovations in Communication and Computational Sciences: Proceedings of ICSICCS 2017, https://doi.org/10.1007/978-981-10-8968-8_38

[9] **Kommadath, R.**, C. Sivadurgaprasad, and P. Kotecha. *Single phase multi-group teaching learning algorithm for single objective real-parameter numerical optimization (CEC2016)*. in *2016 IEEE Congress on Evolutionary Computation (CEC)*. 2016. https://doi.org/10.1109/CEC.2016.7743919

[10] **Kommadath, R.**, C. Sivadurgaprasad, and P. Kotecha. *Single phase multi-group teaching learning algorithm for computationally expensive numerical optimization (CEC 2016)*. in *2016 IEEE Congress on Evolutionary Computation (CEC)*. 2016. https://doi.org/10.1109/CEC.2016.7744167

[11] **Kommadath, R**. and P. Kotecha. *Performance evaluation of Moth Flame Optimization on real parameter single objective optimization and computationally expensive optimization*. in *2016 IEEE Region 10 Conference (TENCON)*. 2016. https://doi.org/10.1109/TENCON.2016.7847989

[12] **Remya, K.**, P. Varun, and P. Kotecha. *Performance of Particle Swarm Optimization and Pattern Search on CEC 2013 real parameter single objective optimization*. in *2015 IEEE Workshop on Computational Intelligence: Theories, Applications and Future Directions (WCI)*. 2015. https://doi.org/10.1109/WCI.2015.7495511

Presentations

[1] **R. Kommadath** and P. Kotecha, Solving Scheduling Problems Using Multi-Verse Optimizer, Simultaneous Heat Transfer Search and Tree Growth Algorithm, 52nd Annual Convention of ORSI and International Conference, IIM Ahmedabad, India, 2019

[2] **R. Kommadath**, S.S.Chauhan, G.Sharma, P.Kotecha, Performance analysis of Sanitized-Teaching Learning Based Optimization and Sine Cosine Algorithm on production planning problem, Indian Process Systems Engineering Conference, IIT Madras, India, 2019.

[3] **R. Kommadath** and P. Kotecha, Performance Evaluation of Five Recently Proposed Computational Intelligence Techniques on Multi-objective Benchmark Functions (CEC 09), Research Conclave 2017 "An amalgamation of Academia, Industry & Start-up", IIT Guwahati, India, 2017

[4] **R. Kommadath** and P. Kotecha, "Optimal Pollutant Trading using Computational Intelligence Techniques," presented at the *REFLUX* 2017, IIT Guwahati, India, 2017

Other journal publications

[1] Maharana, D., **R. Kommadath**, and P. Kotecha, *An innovative approach to the supply-chain network optimization of biorefineries using metaheuristic techniques.* Engineering Optimization, 2023. **55**(8): p. 1278-1295. DOI: https://doi.org/10.1080/0305215X.2022.2080204

[2] Nagraj, S.M., **R. Kommadath,** P. Kotecha and R. Anandalakshmi, *Multi-objective optimization of vapor absorption refrigeration system for the minimization of annual operating cost and exergy destruction.* Journal of Building Engineering, 2022. **49**: p. 103925. DOI: https://doi.org/10.1016/j.jobe.2021.103925

[3] Maharana, D., **R. Kommadath**, and P. Kotecha, *A mixed-integer linear programming model with multi-unit strategy for distributed biorefinery superstructures with economic and social benefits.* Clean Technologies and Environmental Policy, 2022. **24**(6): p. 1903-1925. DOI: https://doi.org/10.1007/s10098-022-02296-z.

[4] Tesfaye, M., R. Patwa, R. Kommadath, P. Kotecha, and V. Katiyar, *Silk nanocrystals stabilized melt extruded poly (lactic acid) nanocomposite films: Effect of recycling on thermal degradation kinetics and optimization studies.* Thermochimica Acta, 2016. **643**: p. 41-52. DOI: https://doi.org/10.1016/j.tca.2016.09.008

[5] Maharana, D., **R. Kommadath**, and P. Kotecha, *A multi-unit model for the biorefinery supply chain focusing on life cycleanalysis-based capacity planning for the processing units*, Biomass conversion and biorefinery, (Accepted)

Other conference papers

[1] Punnathanam, V., **R. Kommadath**, and P. Kotecha. *Extension and performance evaluation of recent optimization techniques on mixed integer optimization problems*. in *2016 IEEE Congress on Evolutionary Computation (CEC)*. 2016. https://doi.org/10.1109/CEC.2016.7744348

[2] Maharana, D., **R. Kommadath**, and P. Kotecha. *Dynamic Yin-Yang Pair Optimization and its performance on single objective real parameter problems of CEC 2017*. in *2017 IEEE Congress on Evolutionary Computation (CEC)*. 2017. https://doi.org/10.1109/CEC.2017.7969594