University Of AL-Qadisiyah

Available online at www.qu.edu.iq/journalcm

JOURNAL OF AL-QADISIYAH FOR COMPUTER SCIENCE AND MATHEMATICS

ISSN:2521-3504(online) ISSN:2074-0204(print)

# Survey for Open Shop Scheduling

***Tabark Nameer Abd Ali [1*] Luma S.Hasan [2]***

*[1]Al-Qadisiyah University,College of Computer Science and Information Technology,Al-Qadisiyah , Iraq .E-mail: (cm.post23.14@qu.edu.iq )*

*[2]Al-Qadisiyah University,College of Computer Science and Information Technology ,Al-Qadisiyah , Iraq .E-mail: (luma.hasan@qu.edu.iq )*

ARTICLEINFO

*Article history:*
Received: 13 /1/2025
Rrevised form: 29 /1/2025
Accepted : 3 /2/2025
Available online: 30 /3/2025

*Keywords:*

Open shop scheduling,

Operations,

Open shop,

Makespan

**ABSTRACT**

In the Open Shop Scheduling, a group of jobs are assigned to several machines, and each job can be handled on any machine without following a set order. The main difficulty is figuring out the best scheduling configuration to reduce the makespan or the amount of time needed to finish all jobs from the beginning of the first work to the end of the last. Multiple jobs requiring the same machine simultaneously can cause scheduling conflicts, which must be avoided while considering work order flexibility among different machines. Scheduling considerations in OSS are extremely complicated because every job must be executed on every machine exactly once. The goal is to determine an efficient sequence that balances the workload across available machines, minimizing makespan while adhering to various operational constraints. This survey provides a comprehensive analysis of Open Shop Scheduling, a widely studied topic in scientific research focusing on minimizing makespan, synthesizes key contributions from the literature, highlights recent advancements, and outlines potential avenues for future research.

MSC..

https://doi.org/10.29304/jqcsm.2025.17.11965

## 1. Introduction

The main focus of the crucial field of scientific research known as Open Shop Scheduling (OSS) is optimizing the distribution of tasks across several machines or workstations [1]. OSS offers more freedom than rigid scheduling paradigms like flow shop or job shop scheduling, enabling each job to be processed in any order on any machine.

This intrinsic adaptability creates a difficult problem, especially when the goal is to reduce the makespan or the amount of time needed to finish all jobs from the start of the first to the end of the last. The OSS problem is characterized by the need to determine an optimal schedule where each job is executed on a selected machine exactly once. The challenge arises from minimizing makespan [2][13] and managing potential conflicts arising from simultaneous job demands on the same machine. Therefore, the key to OSS is to develop an effective sequencing plan that complies with several restrictions and optimizes the resources. Minimizing makespan is still a top priority because of OSS's substantial influence on productivity and operational efficiency. Organizations can improve service delivery, lower operating costs, and increase throughput by minimizing the makespan. With an emphasis on

∗Corresponding author

Email addresses:

Communicated by 'sub etitor'

makespan minimization, this review seeks to give a thorough overview of OSS, clarifying the problem description and the approaches.

## 2. Open Shop Scheduling (OSS)

It is a complex combinatorial optimization problem that belongs to the broader family of job scheduling problems, including flow and job shop scheduling. In OSS, a set of jobs must be processed on multiple machines, but unlike flow or job shops, there is no predetermined sequence of operations for each job on the machines. This lack of precedence constraints in OSS provides more flexibility. Still, it adds complexity,

as any job can be processed on any machine at any time as long as each job is processed on all machines exactly once. The primary objective of Open Shop Scheduling is to optimize the allocation of jobs to machines to minimize one or more performance measures, which are critical in industrial, manufacturing, and computing environments [9] [10].

OSS's most commonly studied performance measures include minimizing the makespan, total completion time, total tardiness, and the number of tardy jobs. The makespan($C_{max}$) is one of the key objectives in OSS and represents when the last job finishes processing on the machines. Minimizing the makespan increases efficiency by reducing the time required to complete all jobs. This is particularly important in industries where minimizing downtime and maximizing throughput is critical for maintaining competitive advantage. When minimized, the overall flow time of jobs in the system is decreased. The sum of the completion times of all jobs is another important performance metric, and reducing in-process inventory and enhancing resource usage is closely related to this statistic. Different tasks frequently carry different levels of relevance or priority in real-world applications. In OSS, this is achieved by giving jobs weights and reducing the weighted sum of completion times. The weighted performance metrics are particularly applicable in real-world situations where certain tasks may be more important than others because of client priority, deadlines, or financial impact. Similarly, being late is a major problem, particularly in systems where deadlines must be fulfilled. The tardiness of a job, defined as the difference between its completion time and its due date, is penalized to ensure that jobs are completed as close to their deadlines as possible. Minimizing total tardiness and weighted total tardiness is crucial in environments where late completion can incur significant penalties or reduce customer satisfaction.

The number of tardy jobs and the maximum lateness are crucial performance metrics ($L_{max}$). The focus is on reducing the number of jobs that exceed their due dates, while the latter aims to minimize the worst-case scenario, where the job with the greatest delay is reduced as much as possible. These measures ensure that the scheduling system is efficient and robust to delays, which can be crucial in industries where delays in one part of the process can have a cascading effect on subsequent operations. Given its flexibility and complexity, the Open Shop Scheduling problem has been a topic of extensive research with various heuristics, metaheuristics, and exact algorithms proposed to solve different variants of the problem. The challenge lies in the NP-hard nature of the OSS problem, which means that finding optimal solutions for large instances is computationally infeasible, and hence, practical approaches often rely on approximation methods or heuristic techniques that provide near-optimal solutions within reasonable computational times [9] [11].

Solutions to OSS are widely applied in areas such as manufacturing, telecommunications, and even cloud computing, where the efficient allocation of resources is essential to operational success. Open Shop Scheduling is a vital optimization problem in operations research, with numerous practical applications in industry and services. Its performance measures, from minimizing to reducing tardy jobs and tardiness, ensure that operations run smoothly, deadlines are met, and resources are used effectively.

Over the years, numerous open shop scheduling problem variants have been explored. While the standard problem is well-defined, many studies have extended it by adding or modifying assumptions and constraints. For instance, although related to open shop scheduling, flexible manufacturing systems (FMS) involve additional complexities and are often treated separately in the literature [8]. Research into open shop scheduling, as depicted in Figure 1. can be broadly categorized into three main classes of solution methods: exact algorithms, deterministic heuristics, and metaheuristics. This simple classification effectively captures the diversity of approaches to solve the open shop scheduling problem. A significant body of work has focused on minimizing the makespan, often using branch and bound (B&B) algorithms, which are considered one of the most powerful exact methods for solving scheduling problems.

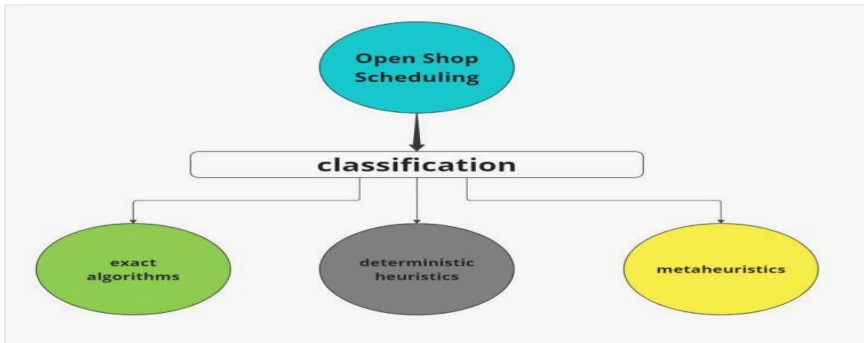


**Fig 1. Open Shop subdivision [16]**

To clarify more through Gantt chart 1. similar to the Open Shop Scheduling (OSS) problem's machines. Each job in OSS must go through a group of machines, but there is no predetermined order in which they must be completed, giving scheduling freedom. The horizontal axis of the chart represents time, while the vertical axis represents machines (M1, M2, etc.). Tasks (J1, J2, etc.) are represented by colored bars on the chart, each indicating how long a specific work will take on a particular machine [5].

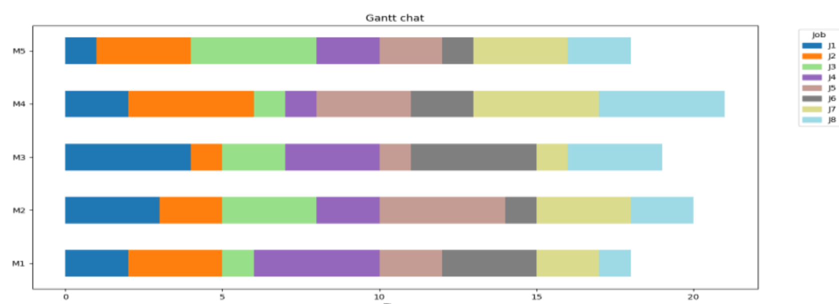


**Fig 3. Gannet Chart for open shop scheduling with 8 jobs and 5 machines.**

## 2.1 Performance Measures in Open Shop Scheduling Optimization

The open shop scheduling problem is a significant challenge in scientific research, particularly in optimizing the use of resources in manufacturing and service environments. Open shop scheduling primarily aims to assign jobs to machines to optimize specific performance criteria. In practice, these criteria are expressed through several performance measures. Some of the key measures commonly addressed include:

**Minimizing the makespan ($C_{max}$)**: Minimize Completion Time ($C_{max}$): This is a very important metric that indicates the maximum time to complete all the jobs in the schedule. It is represented mathematically as follows:

$$C_{\max} = \max(C_i) \quad \ldots \quad (1)$$

Where $C_i$ is the completion time for a job i.

**- Minimizing the sum of completion times**: Another important metric is the total flow time, which is the sum of the completion times for all jobs, given by:

$$\sum_{i=1}^{n} Ci \qquad \ldots \quad (2)$$

This measure seeks to minimize the cumulative time jobs spend in the system, directly impacting operations efficiency and minimizing in process inventory.

**- Minimizing the weighted sum of completion times:** When jobs have varying importance or priorities, a weighted sum approach is employed. Each job i is assigned a weight Wi based on its priority, and the objective is to minimize the weighted sum:

$$\sum_{i=1}^{n} WiCi \qquad \ldots \quad (3)$$

This method ensures that higher-priority jobs are processed more quickly, reflecting the organization's or production system's specific needs.

**- Minimizing tardiness and lateness**: Tardiness is the amount by which the completion time of a job exceeds its due date. The tardiness $T_i$ for a job i is calculated as:

$$T_i = \max(0, C_i - d_i) \qquad \ldots \quad (4)$$

Ci is the completion time, and di is the due date for a job i. The total tardiness is minimized by optimizing the scheduling of jobs to ensure they meet or fall within their deadlines:

$$\sum_{i=1}^{n} Ti \qquad \ldots \quad (5)$$

This helps in reducing penalties or losses due to late job completion.

**- Minimizing the weighted sum of tardiness**: Similar to the weighted sum of completion times, this measure accounts for job priorities. The weighted tardiness $W_iT_i$ is minimized, where Wi represents the importance of job i. The objective is:

$$\sum_{i=1}^{n} WiTi \qquad \ldots \quad (6)$$

Which ensures that critical jobs experience minimal delays.

**-Minimizing the number of tardy jobs**: This performance measure focuses on reducing the number of completed jobs after their due dates. It is calculated by summing an indicator variable $U_i$, where $U_i = 1$ if $C_i > d_i$; otherwise, $U_i = 0$. The goal is to minimize:

$$\sum_{i=1}^{n} Ui \qquad \ldots \quad (7)$$

representing the total number of late jobs.

**- Minimizing the weighted number of tardy jobs**: In situations where some jobs are more critical than others, their importance is weighted. The objective is to minimize:

$$\sum_{i=1}^{n} WiUi \qquad \ldots \quad (8)$$

Wi represents the job's importance, and $U_i$ indicates whether a job is tardy.

**- Minimizing the maximum lateness ($L_{max}$)**: Lateness $L_i$ is defined as:

$$L_i = C_i - d_i \qquad \ldots \quad (9)$$

Representing the difference between a job's completion time and its due date. The objective is to minimize the maximum lateness across all jobs:

$$L_{max} = \max(L_i) \quad \ldots \quad (10)$$

It improves overall timeliness and system reliability by ensuring that even the most delayed jobs have the fewest minutes of lateness.

## 2.2 Applications of OSS

Open Shop Scheduling (OSS) has various applications across various industries, making it a crucial area of study in scientific research. OSS is particularly helpful in the manufacturing sector as it optimizes workflows requiring numerous components that must be processed through several stages. By allowing manufacturers to adjust to shifting production needs, job sequencing flexibility boosts overall productivity and efficiency. The healthcare industry also uses it to enhance patient scheduling across hospital departments. Healthcare facilities can decrease wait times and enhance patient care, improving health outcomes by effectively allocating resources and setting up appointments. In telecommunications, OSS is essential for controlling data packet routing across networks. Through optimum data transmission operation scheduling that ensures timely information delivery, OSS contributes to better network performance and resource utilization [15]. OSS is also useful in transportation and logistics, where it may be applied to optimize vehicle routing for pickups and deliveries, allowing businesses to lower expenses and improve operational efficiency by minimizing makespan [13].

Overall, OSS's versatility makes it applicable to various fields, where effective scheduling contributes to improved operational performance and resource utilizations.

## 2.3 Related Work

The literature review thoroughly examines the Open Shop Scheduling Problem (OSSP), highlighting various algorithms and approaches to optimize scheduling activities in different circumstances. The review highlights both traditional and modern techniques for reducing makespan and enhancing scheduling efficiency by combining studies that use strategies like Bat Algorithm, Longest Processing Time (LPT), Genetic Algorithms, and Quantum Computing with more recent methods like Whale Optimization and Quantum Computing. This review's broad coverage of benchmark problems and datasets, such as the well-known Taillard cases, is one of its main advantages since it offers a solid foundation for evaluating the effectiveness of various algorithms. Furthermore, the paper includes innovative techniques like hybrid algorithms and the application of Graph Attention Models, demonstrating the changing terrain of OSSP research. Information on the scalability and flexibility of different approaches, including algorithms evaluated on small, medium, and large-scale datasets, further enhances the review's practical usefulness. To close the gap between theoretical developments and real-world applications in manufacturing, logistics, and healthcare, the paper also highlights the interdisciplinary character of scheduling challenges. The breadth and depth of the literature covered in this review provide a valuable foundation for future research, particularly in identifying trends, evaluating algorithmic performance, and exploring new avenues for optimization in OSSP. Despite its comprehensive scope, the literature review has some limitations. One disadvantage is the lack of in-depth analysis comparing the computational complexities of different algorithms, which can be crucial for understanding their practical applicability in large-scale problems. Additionally, while the review includes a variety of algorithms, it does not fully explore the limitations or potential biases in the datasets used, such as Taillard instances, which might not represent all real-world scenarios. Moreover, the review leans heavily on performance metrics like makespan without considering other important factors such as resource constraints, energy consumption, or multi-objective optimisation, which are increasingly relevant in scheduling research today. Finally, while cutting-edge approaches are mentioned, the integration of emerging technologies like machine learning have not been thoroughly examined, leaving gaps in exploring future Open Shop Scheduling Problem research trends.

M. B. Shareh et al., proposed in [5] the Bat Algorithm for solving open shop scheduling problems(OSSP). Specifically, standard open shop benchmarks from Taillard criteria are used in the experiments, along with different problem scenarios like 4x4, 5x5, and so on, involving various jobs and machines. The optimal makespan obtained by the algorithm was 193. Examining the Bat Algorithm's performance compared to other benchmark algorithms is essential, even though it showed a makespan of 193. Testing the Bat Algorithm across a larger range of benchmarks or problem scenarios, especially in real-world applications where the problem size and complexity fluctuate, could be a crucial factor for future research. This would provide valuable insights into its scalability and computational efficiency compared to other leading methods.

In [12], M. Sophia et al. used the Longest Processing Time (LPT) algorithm to solve the Open Shop Scheduling Problem to minimise the total completion time. For a case of 3 jobs and 3 machines, with varying processing durations on every machine for every workload, the total completion time using the LPT algorithm was 35 hours, with 4 hours of idle time.

M. M. Ahmadian et al . [13] discussed minimizing makespan and the time to complete all jobs. The article reviews a body of literature spanning four decades and illustrates the complexities associated with this type of scheduling and the various solutions developed over time. The makespan values presented that the proposed method (BA_OS) is highly competitive with other established algorithms across small and large problem instances; the optimal makespan solution was 937.

S. Aghighi et al. [14] used an open shop scheduling problem with Reverse Flows, which aims to minimize the makespan, i.e., reduce the total time required to complete all tasks on all available machines. Several algorithms were tested on small, medium, and large datasets. Software such as GAMS was used to solve small examples. In contrast, algorithms such as Differential Evolution (DE), Genetic Algorithm (GA), and Whale Optimization Algorithm (WOA)were used to solve large examples of algorithms such as DE, GA, WOA, BA, etc., were compared to evaluate the effectiveness of the solutions. DE proved to be very effective in finding good solutions compared to other algorithms.

A. M. Cañadas et al., [15] discuss the Open Shop Scheduling (OSS) Problem and use an algebraic approach to solve this problem, where minimizing makespan is the primary objective. The use of multiple models of time matrices associated with operations performed on a set of machines (e.g., three jobs on different machines). These models are based on the schedules of jobs on machines, where the processing times are generated randomly using specific distributions in some cases. The experimental results presented in the study showed that 95% of the calculated distributions satisfy the minimum $*\beta$ of Makespan.

In [16], W. Kubiak used the problem of "Open Shop Scheduling" on machines to achieve the shortest total duration makespan. Makespan refers to the time it takes to perform all jobs in a given system using the available resources. It appears that the focus was on minimizing makespan using various algorithms, such as the network flow algorithm, which aims to schedule operations in the best way on machines to reduce the total time. The results indicate that the minimum makespan depends on multiple factors, such as the complexity of the resource graph and conflicts between operations, taking into account the use of approximation methods to obtain acceptable results when the problem cannot be solved with mathematical precision. Optimal scheduling with a makespan value of 5 for some schedules with no waiting periods and other makespan values such as 4 and 8 in different contexts of scheduling problems are found, and it is also shown how these values can be achieved via specific computational techniques to reduce the time and reach feasible schedules.

In [17], H. M. Gu et al. deal with the Open Shop Scheduling (OSS) and minimising Makespan using the Whale Optimization Algorithm with Local Search (HWOA); the Makespan results for some cases were as follows: 10 jobs and 5 machines: the minimum Makespan value was 175 and 30 jobs and 5 machines: the minimum value was 482 and 20 jobs and 10 machines: the minimum value was 334 and 50 jobs and 5 machines: the minimum value was 764. These values reflect the minimum Makespan for each case using the HWOA algorithm.

A. Atay et al., proposed in [18] multiple tasks that require allocation to a set of resources such as machines, and several well-known standard datasets have been used in the literature to solve open-shop scheduling problems. Algorithms based on cooperative game theory have been used with resource allocation to minimise the total cost and waiting time between tasks. It is an algorithm based on task rescheduling to achieve time and cost savings based on the cooperation of multiple players in the system. Experiments have been conducted on different cases, and the results have shown that improvements in task re-arrangement lead to significant time savings.

In [19], J. Li et al. used the Graph Attention Model (GAM) with discount memory integration to solve the Open Shop Scheduling Problem (OSSP) to reduce the total time to complete all operations (makespan). The proposed algorithm game (attention model with discount memory) achieved excellent results, as it reached a solution quality that approximates the lower bound of the total time. Compared to traditional tools such as or-tools, the proposed algorithm provides similar solution quality but with less computational time.

In [20] H. Wang and W. Chen compares scheduling algorithms used in pick-and-drop robot tasks in logistics environments. The research focuses on minimizing the total make-span, utilizing a variety of algorithms to allocate functions between pick-and-drop robots. The open-shop scheduling problem shares some characteristics with the pick-and-drop robot scheduling problem (TPS) regarding the order of execution of tasks between robots. The value of $C_{max}$, which represents the makespan in a given example, is equal to 3 in one case. This means the time required to complete all the tasks is 3-time units. It was also shown that there is another case where $C_{max} = 2$ could not be achieved, as the schedule that can complete the tasks now does not exist.

In [21] Z. Zhuang et al., Improved the solution of the scheduling problem using a combination of complex networks and dynamic heuristics, it presents a general framework for transforming the scheduling problem into a network model, where processes are represented as nodes in a network, and the time constraints between these nodes are defined as edges connecting them. The goal is to find the optimal order of visits to these nodes to complete all tasks quickly (minimizing the makespan). We used several types of scheduling, Flow Shop Scheduling, Job Shop Scheduling, and Open Shop Scheduling, and it was shown that we could solve the Open Shop problem with linear constraints in polynomial time, i.e. in a more efficient way compared to other types of scheduling.

In [22], K. R. A. Kumar and J. E. R. Dhas used to solve the Job Shop Scheduling Problem(JSSP) the Gannet Optimization Algorithm (GOA). The study investigated 23 benchmark problems, typical in JSSP research, including well-known instances like LA12 to LA39. These problems were used to evaluate the efficiency of GOA in minimizing manufacturing time, a common goal in scheduling. The open shop scheduling problem (OSSP) was studied using three simulation algorithms (SA - A1), which gave good but not the best results; the discrete Firefly algorithm (DFA -A2), which achieved better results than SA in several cases, and a hybrid algorithm (A3): was the most effective, achieving the best results in 76% of the experiments. Thus, the main result is that the hybrid algorithm (A3) is the best for minimizing the average flow time in open shop scheduling. Thus, the hybrid algorithm combining DFA and SA was the most efficient in solving OSSP problems and minimizing the average flow time, especially in more complex cases with increasing tasks and machines.

L. Binkowski and C. Tutschku [23] used hard constraints to solve Open-Shop Scheduling. Analyses were performed using quantum algorithms to address this problem. The algorithm was applied to a small model using an IBM Q system on one computer, and the results were positive, as good solutions were reached with reduced time. Some errors were observed in samples close to the optimal solution, but the performance was promising. A noise-free simulator was used to analyses the results, where the simulator showed a higher ability to reach optimal solutions compared to the real quantum device that suffers from some noise. Thus, the study highlights the potential of using quantum computing to improve complex schedules and reduce task completion time.

In [24], H. Alghamdi et al. proposed Algorithms used to solve the open-shop scheduling problem, such as Genetic Algorithms (GA) and Particle Swarm Optimization (PSO), which can also be effective in solving university scheduling problems, as they help in finding optimal solutions under multiple constraints, such as schedule conflicts or resource availability.

In [25], A.M. Tsirlin used a job-shop scheduling problem where patients must visit different points of service (POS), and the goal is to minimize waiting time. The branch-and-bound method is applied to find an optimal solution by considering upper- and lower-bound estimates.

M.A. Belmamoune et al. proposed in [26] the Job Shop Scheduling Problem (JSSP) using a reinforcement learning algorithm, specifically a Q-Learning algorithm. We introduce a new representation of the state of the scheduling environment and utilise task ordering rules like Shortest Processing Time (SPT) and Longest Processing Time (LPT), among others, as the actions performed by the agent based on the scheduling environment. A set of several standard scenarios within a subset of benchmark instances from the famous OR Library solved a scheduling challenge. The main goal for this type of challenge is to minimize makespan.

In [27], A. Bouzidi et al. proposed to solve the Open Shop Scheduling Problem (OSSP), an NP-Hard problem, using the Cat Swarm Optimization (CSO) algorithm. This algorithm minimizes Makespan or the total time required to process all jobs by assigning jobs to resources (such as machines) as efficiently as possible. The Cat Swarm Optimization (CSO) algorithm, which has two modes, is based on mimicking how cats would behave when looking for the best solution: Seeking Mode and Tracing Mode. The algorithm has been improved to increase its efficiency by adding operations such as switching locations and updating using searching and tracing.

In [28], Yu Fu and Amarnath Banerjee used the stochastic programming model for service scheduling with uncertain demand, and it is applied to clinic scheduling based on the open access policy. Stochastic Integer Programming (SIP) optimized patient allocation among available time slots. The Sample Average Approximation (SAA) method was adopted to calculate optimal solutions based on expected random data, and scheduling was optimised to handle uncertain demand in clinics. The model showed strong results in handling increased demand from conditions such as the COVID-19 pandemic. Costs related to long waiting times and patients not attending their appointments were reduced.

S. K. N. V Shakhlevich proposed in [29]. It deals with shop scheduling problems, including the Open Shop Scheduling Problem (OSSP), where flexible tasks whose processing time can be flexibly distributed among machines are dealt with. The study used algorithms based on the optimal division of operations among machines with flexible distribution of processing times and focused on minimizing the total time to complete all tasks (Makespan).

L. R. Abreu et al.[30] used Genetic Algorithms to Solve Open Shop Scheduling Problems (OSSP) with Sequence-Dependent Setup Times. The goal of these algorithms is to minimize Makespan (total time to complete all tasks), Using Genetic Algorithms in parallel with construction algorithms such as Bounded Insertion Constructive Heuristic (BICH) and Minimal Insertion Heuristic (MIH). Also, using examples such as the GP03-01 dataset, one of the examples shows a Makespan of 1040 time units. The proposed algorithm outperforms other solutions, such as Mixed Linear Programming (MILP), and other methods, such as Electromagnetic Heuristic (EH).

In [31], T. F. Abdelmaguid's Dynamic Multiprocessor Open Shop Scheduling was addressed to minimize Makespan (total time to complete all tasks) and Mean Weighted Flow Time (MWFT). The data used in the experiments included 30 small random populations and an Exact Algorithm based on the mixed integer linear Programming (MILP) model was used. IBM ILOG CPLEX software was used to solve the generated sub-problems.

V. H. H. Lopes Costa Lima [32] discussed the problem of open workshop scheduling using hybrid algorithms, including Tabu Search and ils; the proposed algorithm was tested on 140 cases known in the literature, including datasets used In Taillard, Guéret, And Prins experiments, where jobs were assigned to different machines. The time required to complete the tasks was measured, and tests were conducted on other cases; the proposed algorithm was successful in improving the makespan results, where the makespan ranged between 1399 time units in one of the experimental cases, and this was dependent on the number of machines and the assigned tasks. The results indicate that the algorithm provided competitive results with little deviation from the optimal solutions.

In [33], Wiesław Kubiak discussed NP-hard problems in scheduling, particularly focusing on two-machine open shops. It mentions that the makespan minimization for this type of problem is NP-hard, even when operations are unit-time.

Specifically, for a two-machine open shop with unit-time operations and time lags, the problem is shown to be computationally complex. According to the results, the makespan problem in the open workshop with time gaps is NP-hard in its strongest form, even if there are only two machines and every operation takes the same time. Because of its complexity, the problem cannot be solved efficiently with traditional algorithms, so researchers are concentrating on creating competitive and approximate algorithms to solve it. As a result, a competitive algorithm was created to solve the makespan problem in open workshops with time gaps, and an approximate method was created to solve the problem.

In [34] J. Dong et al., an approximate polynomial time method (EPTAS) is used to solve the multistage open shop scheduling problem to minimize the makespan or the total sum of completion times. The suggested algorithm is effective when applied to jobs planned on multistage open shops. The method is applied to the scheduling problem to minimize the makespan of the time the final job finishes.

In [35], S. Hassan et al. discussed the two primary goals of the open shop scheduling problem (OSSP): minimizing the overall make-span and task delay. Large difficulties were solved by applying mathematical and approximation techniques to data from various workshops and procedures. Multi-objective mixed linear programming (MOMILP), multi-objective parallel genetic algorithm (MOPGA), and multi-objective parallel simulated cooling (MOPSA) were among the techniques employed to minimize these goals. These algorithms' performances on both small and large issues were compared. On big problems, MOPGA was better at finding the best answers, whereas MOPSA was quicker but marginally less effective at producing high-quality solutions.

A. A. Rahmani Hosseinabadi et al., [36] In the discussion of the Open Shop Scheduling Problem (OSSP) and its solution using the Extended Genetic Algorithm (EGA), multiple datasets were used to test the effectiveness of the proposed algorithm. In the dataset Test_4_4, the optimal Makespan achieved was 193, which aligns with the optimal solution found in the standard benchmarks (Taillard Benchmark).

G. Ni and L. Chen [37] used to solve the three-machine proportionate open shop and mixed shop scheduling problem in proportional operating environments. The algorithms developed in the paper appear to be based on data on machine job scheduling with given processing times. For makespan (the maximum time required to complete all jobs), improvements are achieved using approximate algorithms. Key results include a three-machine proportionate open shop problem: the approximate algorithm is improved from 7/6 to 13/12. Three-machine mixed shop problem: an improved approximate algorithm is presented with 7/6 + α, better than the previous results 5/3.

K. Nip and Z.Wang [38] discuss the problem of scheduling two-machine shops under linear constraints. This type of problem aims to schedule the tasks on two machines to minimize the Makespan (the total time required to complete all tasks). Several approximate algorithms are presented to solve two-machine shop scheduling problems, including a 2-ratio approximation algorithm, and Polynomial-Time Approximation Schemes (PTAS) have been developed that can improve the optimal solution.

In [39] R.Reijnen et al. used the job shop scheduling problem (JSSP) and its variants, such as dynamic scheduling (DyJSSP) and distributed scheduling (DiJSSP), and used benchmark datasets such as the Taillard dataset, which are often used to test solutions in job shop scheduling problems and most notably Makespan optimizations, so methods such as DRL algorithms and GNNs were used to develop solutions that reduce Makespan time while improving performance across multiple benchmark datasets. In addition, several studies compared the performance of their techniques with techniques such as L2D and PDR (prioritisation rules) and demonstrated superior performance

in several experimental cases. For example, using GNNs with DRL was faster and more accurate than traditional optimization methods such as Tabu Search and Genetic Algorithms. Hybrid methods, such as using neural networks with local search methods, were also proposed, improving solutions' accuracy at the expense of longer running time.

K. Ho et al.[40] discussed the Open Shop Scheduling Problem (OSSP) using a set of benchmark datasets, including the Taillard (TA) dataset, one of the most popular datasets used to test scheduling algorithms and also for makespan (the total time to complete all tasks), a new algorithm called Residual Scheduling was used. Experiments showed that this algorithm achieved Zero gaps in 49 out of 50 JSP problems containing more than 150 processes distributed across 20 machines. On the TA dataset, the average makespan gaps using the RS algorithm were lower than other methods, such as L2D and Schedule Net, with gaps ranging from 0.026 to 0.177, depending on the problem size.

A.H. Lal et al. proposed in [41] an Open-ended Scheduling Problem (OSSP) focusing on minimizing the average flow time. 50 OSSP problems were used with random processing times generated using a random number generator, and three algorithms were used to reduce the average flow time: Simulated Cooling (SA) - denoted as A1, and Discontinuous Firefly Algorithm (DFA) - denoted as A2 and A hybrid algorithm combining DFA and SA - denoted as A3. The results showed that the hybrid algorithm A3 provided the best results in 76% of the cases, compared to A1 and A2. The best results were found in all cases with more than 7 jobs and 25 machines, reflecting the effectiveness of this approach.

In [42], A. Kekli used a hybrid algorithm combining Cat Swarm Optimization (CSO) and Genetic Algorithm (GA) to solve the Open Shop Scheduling Problem with the Vehicle Routing Problem (OSSP-VRP). This algorithm minimizes makespan (maximum time to complete operations) and total vehicle distance travelled. For the Taillard dataset, the hybrid algorithm HCSO-GA reached better solutions in 70% of cases than other algorithms, with relatively low gaps between the computed and the best-known solutions. In the Taillard 10×10. In this case, the algorithm achieved $C_{max}$ = 871, which is better than previous algorithms.

**Table 2. Systematic analysis of the related work**

| Ref. | Author | Year | Dataset | Proposed Method | Results |
|---|---|---|---|---|---|
| [5] | M. B. Shareh et al. | 2020 | Taillard | Bat Algorithm | The optimal makespan found by the algorithm is 193. |
| [12] | M. Sophia et al. | 2024 | For 3 jobs and 3 machines | LPT (Longest Processing Time) | The LPT algorithm's total completion time was 35 hours, with 4 hours of idle time. |
| [13] | M. M. Ahmadian et al. | 2021 | **Taillard** | BA_OS algorithm | For a 15×15 problem, the optimal makespan solution was 937. |
| [14] | S. Aghighi et al. | 2024 | **Taillard** | Differential Evolution (DE), Genetic Algorithm (GA), and Whale Optimization Algorithm (WOA). | For small problems, the makespan ranged from about 222 to 317 seconds. For medium problems, it ranged from 472 to 10,845 seconds. For large problems, it was much higher, reaching about 362,466 seconds in some cases. |
| [15] | A. M. Cañadas et al. | 2023 | 3 jobs on different machines | Algebraic approach | 95% of the calculated distributions satisfy the minimum ∗β of Makespan. |
| [16] | W. Kubiak | 2022 | A set of data used in a scientific study or experiment. | Network Flow Algorithm. | Some schedules with no waiting periods have a Makespan value of 5, while other makespan values, such as 4 and 8, are found in different contexts of scheduling problems. |

| | | | | | |
|---|---|---|---|---|---|
| [17] | H. M. Gu et al. | 2019 | Test 10 _ 5<br>Test 30 _ 5<br>Test 20 _10<br>Test 50 _ 5 | **Whale Optimization Algorithm with Local Search (HWOA)**. | The lowest makespan value was :<br>175, 482, 334,764. |
| [18] | A. Atay<br>et al. | 2021 | standard datasets | **Algorithms based on cooperative game theory** | improvements in task re-arrangement lead to significant time savings. |
| [19] | J. Li et al. | 2020 | standard datasets | algorithm gam-dm (attention model with discount memory) | reached a solution quality that approximates the lower bound of the total time. |
| [20] | H. Wang and W. Chen | 2021 | Random dataset | Decoupled heuristics.<br>Coupled heuristics. | The time required to complete all the tasks is 3 Time units. It was also shown that there is another case where Cmax = 2 could not be achieved, as the schedule that can complete the tasks now does not exist. |
| [21] | Z. Zhuang et al. | 2019 | nodes in a network | A combination of complex networks and dynamic heuristics | It was shown that we can solve the Open Shop problem with linear constraints in polynomial time, i.e., more efficiently than other types of scheduling. |
| [22] | K. R. A. Kumar and J. E. R. Dhas | 2023 | **Taillard** | Gannet Optimization Algorithm (GOA) | The makespan for LA 12 is 1038, for LA 13 is 1149, and for LA 14 is 1291. |
| [23] | L. Binkowski and C. Tutschku | 2023 | Implemented on IBM Q | Variable Quantum Algorithms (VQAs) | focuses on presenting new algorithms and applying them to small OSSP cases. |
| [24] | H. Alghamdi et al. | 2020 | University Scheduling | (Genetic Algorithms).<br>(Particle Swarm Optimization). | Help find optimal solutions under multiple constraints, such as schedule .conflicts or resource availability |
| [25] | A.M. Tsirlin | 2022 | Health dataset | branch-and-bound algorithm. | Find the optimal scheduling solution that minimises patient waiting time across different Service Points. |
| [26] | M. A. Belmamoune et al. | 2023 | Benchmark dataset | Q-learning, Deep Q-Networks (DQN), and cooperative DQN agents. | The Q-learning algorithm improved minimising makespan compared to traditional dispatching rules like FIFO, SPT, and LPT. |
| [27] | A. Bouzidi et al. | 2019 | **Guéret dataset** | Cat Swarm Optimization (CSO) | The best-known solutions, the best solutions found by the CSO method, relative percentage deviation (RPD), and average execution time. |
| [28] | Yu Fu and<br>Amarnath<br>Banerjee | 2021 | Datminimisingocal clinic | Stochastic Integer Programming (SIP)<br>Sample Average Approximation (SAA) | The model showed strong results in handling increased demand from conditions such as the COVID-19 pandemic. Costs related to long waiting times and patients not attending their appointments were reduced. |
| [29] | S. K. N. V Shakhlevich | 2019 | Taillard | algorithms based on the optimal division of operations among machines with flexible distribution of processing times | With unlimited flexibility, faster and less complex solutions are reached compared to cases of restricted flexibility. |

| | | | | | |
|---|---|---|---|---|---|
| [30] | L. R. Abreu et al. | 2020 | GP03-01 dataset | Genetic Algorithm | The makespan is 1040 time units. |
| [31] | T. F. Abdelmaguid | 2020 | 30 small random group | Exact Algorithm | The total Makespan time ranged between 242 and 257 time units. |
| [32] | V. H. H. Lopes Costa Lima | 2020 | Taillard, Guéret Prince | Tabu Search Algorithm.<br><br>Iterated Local Search (ILS). | The Makespan ranged between 1399 time units. |
| [33] | Wiesław Kubiak<br><br>et al. | 2022 | Benchmark dataset | Approximate and competitive algorithms | 3/2 approximate algorithm: TThe solution provided by this algorithm will be, at most, 50% larger than the optimal solution. 2/2 competitive algorithm: This algorithm achieves results that are twice as close to the optimal solution. |
| [34] | J. Dong et al. | 2022 | Taillard | Efficient Polynomial Time Approximation Algorithm (EPTAS). | EPTAS provides approximate solutions that are very close to the optimal solution with improvements up to a high efficiency level. Still, it does not offer a fixed or specific value for a given makespan. |
| [35] | S. Hassan et al. | 2022 | Data from multiple processes and workshops | Multi-Objective Parallel Genetic Algorithm (MOPGA) | While MOPSA was faster, it was slightly less efficient regarding solution quality. |
| [36] | A. A. Rahmani Hosseinabadi et al. | 2019 | Test_4_4 | Extended Genetic Algorithm EGA | The optimal Makespan achieved was 193. |
| [37] | G. Ni and L. Chen | 2020 | Three-Machine Proportionate Open Shop and Mixed Shop | Approximate algorithms | The approximate algorithm has been improved from 7/6 to 13/12. For the three-machine mixed shop problem, an enhanced approximate algorithm is presented with 7/6 + α, better than the previous results of 5/3. |
| [38] | K. Nip and Z. Wang | 2021 | The tasks on two machines | Polynomial-Time Approximation Schemes - PTAS | A 2-ratio approximation algorithm and Polynomial-Time Approximation Schemes (PTAS) have been developed that can improve the optimal solution. |
| [39] | R. Reijnen<br><br>et al. | 2024 | Taillard | DRL algorithms and GNNs | GNNs with DRL were faster and more accurate than traditional optimisation methods such as Tabu Search and Genetic Algorithms. |
| [40] | K. Ho et al. | 2023 | Taillard | a new algorithm called Residual Scheduling | There were zero gaps in 49 of 50 JSP problems containing more than 150 processes distributed across 20 machines. On the TA dataset, the average makespan gaps using the RS algorithm were lower than those using other methods, such as L2D and Schedule Net ,with gaps ranging from 0.026 to 0.177,.depending on the problem size |
| [41] | A. H. Lal<br><br>et al. | 2019 | 50 OSSP problems were used with random processing times generated using a random number generator | Simulated Cooling (SA)<br><br>Discontinuous Firefly Algorithm (DFA)<br><br>A hybrid algorithm that combines DFA and SA | Compared to SA and DFA, the hybrid algorithm A3 gave the best results in 76% of cases. The best results were found in all cases with more than 7 jobs and 25 machines. |

| | | | | | |
|---|---|---|---|---|---|
| [42] | A. Kekli | 2023 | Taillard | C at Swarm Optimization (CSO) and Genetic Algorithm (GA) | the hybrid algorithm HCSO-GA reached better solutions in 70% of cases than other algorithms, with relatively low gaps between the computed and best-known solutions. In the Taillard 10×10 .case, the algorithm achieved $C_{max}$ = 871, which is better than previous algorithms. |

## 3 Future Directions

In the future research domains of scheduling problems, there are many opportunities for improvements in both theoretical constructs and practical uses. Several research studies suggest that scheduling models could be more applicable if they were made with more realistic assumptions. For example, Shareh et al. 's research demonstrates the importance of factoring setup times and equipment maintenance into the maintenance models -- factors often overlooked in the existing models [5]. Further indexing indicates that different methods and variants of different algorithms improve the depth of scheduling results. For example, Aghighi et al. highlight the DE algorithm's effectiveness, which can be further enhanced by considering different objective functions and no-wait multi-objective scheduling problems [14]. This revelation suggests that a more detailed exploration of hybrid strategies that entangle various optimizations may be useful to investigators in the future. Furthermore, Canadas et al. argue that Brauer configuration algebras should be used as a lens to broaden the investigation of open shop scheduling issues. One possible way to do this is to review how these algebraic frameworks correspond to real-life scheduling problems, such as public transportation networks [15]. The techniques' adaptability is evidenced further by proposing other metaheuristic techniques, like those proposed by Bouzidi et al., for different kinds of applications in the open shop scheduling context [27]. Additionally, research into online scheduling techniques is becoming more popular. According to Fu et al., models created for clinical scheduling can be modified for use in more general contexts, such as online job shop scheduling with time windows [28]. Developing models that can adjust to changing needs in various service situations is one important area for future research that this adaptability suggests. Incorporating real-time restrictions, like fluctuating machine speeds and task release times, is crucial. According to Wang et al., taking these limitations into account can make scheduling models more realistic, pointing to the need for more responsive and dynamic scheduling systems [20]. Future research in these areas can greatly advance scheduling approaches and guarantee their continued applicability and efficacy in ever-more complicated operating environments.

## 4 Discussion

Because of its intrinsic complexity and applicability to various industrial applications, the Open Shop Scheduling (OSS) problem has drawn much interest from the operations research community. Despite the advances in OSS research, several areas could still be explored more deeply. For instance, a great opportunity to increase the efficiency of solutions is related to hybrid approaches, where the computational benefits of precise algorithms with heuristic and metaheuristic methods can be combined. Moreover, since industries must still respond to rapidly evolving demands, further investigation would be extremely beneficial to enable OSS solutions to operate in dynamic and uncertain environments. All in all, devising new creative scheduling.

## 5 Conclusion

The open shop scheduling issue has existed for over 40 years and has been a problem that seeks to minimise the manufacturing period. Even if not many studies were done on this topic initially, much research has been done in the past 30 years. We have demonstrated our findings on several conventional open shop scheduling problems and noted the impact of machine count on problem complexity. As a result, scheduling challenges need to be considered in many management and planning concepts. In conclusion, As OSS continues to be critical in various industries, future research should focus on incorporating more realistic constraints, such as machine setup times, and embracing multi-objective approaches to meet the demands of modern scheduling environments. By advancing these methods, OSS solutions can become more adaptable, scalable, and practical for real-world applications, driving continued innovation in complex scheduling problems.

## References


[1] J. Dick, J. M. Schumann, B. Nuseibeh, L. Athens, J. Zobel, and A. M. Qazi, "Scheduling: theory, algorithms, and systems," 2022.

[2] K. R. Baker and D. Trietsch, Principles of sequencing and scheduling. John Wiley & Sons,2018.

[3] S. M. Goldansaz, F. Jolai, and A. H. Z. Anaraki, "A hybrid imperialist competitive algorithm for minimising makespan in a multi-processor open shop," Applied Mathematical Modelling, vol. 37, no. 23, pp. 9603–9616, 2013.

[4] T. Gonzalez and S. Sahni, "Open shop scheduling to minimise finish time," Journal of the ACM (JACM), vol. 23, no. 4, pp. 665–679, 1976.

[5] M. B. Shareh, S. H. Bargh, A. A. R. Hosseinabadi, and A. Slowik, "An improved bat optimisation algorithm to solve the tasks scheduling problem in an open shop," Neural Computing and Applications, vol. 33, pp. 1559–1573, 2021.

[6] C. Low and Y. Yeh, "Genetic algorithm-based heuristics for an open shop scheduling problem with setup, processing, and removal times separated," Robotics and Computer-Integrated Manufacturing, vol. 25, no. 2, pp. 314–322, 2009.

[7] A. H. Lal, V. KR, and A. N. Haq, "The mean flow time in open shop scheduling," Journal of Advances in Management Research, vol. 17, no. 2, pp. 251–261, 2020.

[8] I. Pietri and R. Sakellariou, "Mapping virtual machines onto physical machines in cloud computing: A survey," ACM Computing Surveys (CSUR), vol. 49, no. 3, pp. 1–30, 2016.

[9] M. L. Pinedo and M. L. Pinedo, "Design and implementation of scheduling systems: More advanced concepts," Scheduling: Theory, Algorithms, and Systems, pp. 485–508, 2016.

[10] J. Blazewicz, K. H. Ecker, E. Pesch, G. Schmidt, and J. Weglarz, Scheduling computer and manufacturing processes. Springer Science & Business Media, 2013.

[11] E. L. Lawler, J. K. Lenstra, A. H. R. Kan, and D. B. Shmoys, "Sequencing and scheduling: Algorithms and complexity," Handbooks in operations research and management science, vol. 4, pp. 445–522, 1993.

[12] M. Sophia, S. Jayakumar, and M. Lakshmi, "Minimizing the total completion time for hypothetical cases in open shop scheduling problem."

[13] M. M. Ahmadian, M. Khatami, A. Salehipour, and T. Cheng, "Four decades of research on the open-shop scheduling problem to minimise the makespan," European Journal of Operational Research, vol. 295, no. 2, pp. 399–426, 2021.

[14] S. Aghighi, E. Mehdizadeh, and S. T. A. Niaki, "No-wait open-shop scheduling problems with reverse flows," Journal of Engineering Research, 2024.

[15] A. M. Cañadas, O. M. Mendez, J.-C. Riaño-Rojas, and J.-D. Hormaza, "An algebraic approach to the solutions of the open shop scheduling problem," Computation, vol. 11, no. 5, p. 94, 2023.

[16] W. Kubiak, Book of Open Shop Scheduling. Springer, 2022.

[17] H.-M. Gu, R. Hu, B. Qian, H.-P. Jin, and L. Wang, "Whale optimisation algorithm with local search for open shop scheduling problem to minimise the makespan," in Intelligent Computing Theories and Application: 15th International Conference, ICIC 2019, Nanchang, China, August 3–6, 2019, Proceedings, Part II 15. Springer, 2019, pp. 678–687.

[18] A. Atay, P. Calleja, and S. Soteras, "Open shop scheduling games," European Journal of Operational Research, vol. 295, no. 1, pp. 12–21, 2021.

[19] J. Li, X. Dong, K. Zhang, and S. Han, "Solving open shop scheduling problem via graph attention neural network," in 2020 IEEE 32nd International Conference on Tools with Artificial Intelligence (ICTAI). IEEE, 2020, pp. 277–284.

[20] H. Wang and W. Chen, "Task scheduling for transport and pick robots in logistics: A comparative study on constructive heuristics," Autonomous Intelligent Systems, vol. 1, no. 1, p. 17,2021.

[21] Z. Zhuang, Z. Lu, Z. Huang, C. Liu, and W. Qin, "A novel complex network-based dynamic rule selection approach for open shop scheduling problem with release dates," Math. Biosci.Eng, vol. 16, no. 5, pp. 4491–4505, 2019.

[22] K. Kumar and J. Dhas, "Mitigation of makespan time in job shop scheduling problem using a gannet optimisation algorithm." EAI Endorsed Transactions on Scalable Information Systems, vol. 10, no. 5, 2023.

[23] G. Koßmann, L. Binkowski, C. Tutschku, and R. Schwonnek, "Open-shop scheduling with hard constraints," arXiv preprint arXiv:2211.05822, 2022.

[24] H. Alghamdi, T. Alsubait, H. Alhakami, and A. Baz, "A review of optimisation algorithms for university timetable scheduling," Engineering, Technology & Applied Science Research,
vol. 10, no. 6, pp. 6410–6417, 2020.

[25] A. Tsirlin and A. Balunov, "Job-shop scheduling problem and its solution method," 2022.

[26] M. A. Belmamoune, L. Ghomri, and Z. Yahouni, "Solving a job shop scheduling problem using learning algorithm," in International Workshop on Service Orientation in Holonic and Multi-Agent Manufacturing. Springer, 2022, pp. 196–209.

[27] A. Bouzidi, M. E. Riffi, and M. Barkatou, "Cat swarm optimisation for solving the open shop scheduling problem," Journal of Industrial Engineering International, vol. 15, no. 2, pp.367–378, 2019.

[28] Y. Fu and A. Banerjee, "A stochastic programming model for service scheduling with uncertain demand: an application in open-access clinic scheduling," in Operations Research Forum, vol. 2. Springer, 2021, pp. 1–32.

[29] S. Knust, N. V. Shakhlevich, S. Waldherr, and C. Weiß, "Shop scheduling problems with pliable jobs," Journal of Scheduling, vol. 22, pp. 635–661, 2019.

[30] L. R. Abreu, J. O. Cunha, B. A. Prata, and J. M. Framinan, "A genetic algorithm for scheduling open shops with sequence-dependent setup times," Computers & Operations Research, vol.113, p. 104793, 2020.

[31] T. F. Abdelmaguid, "Bi-objective dynamic multiprocessor open shop scheduling: an exact algorithm," Algorithms, vol. 13, no. 3, p. 74, 2020.

[32] V. H. L. C. Lima and B. de Athayde Prata, "Um algoritmo híbrido busca tabu/ils para o problema de sequenciamento da produção em ambiente open shop," Exacta, vol. 19, no. 4, pp.729–744, 2021.

[33] W. Kubiak, "On the complexity of open shop scheduling with time lags," Journal of Scheduling, vol. 26, no. 3, pp. 331–334, 2023.

[34] J. Dong, R. Jin, G. Lin, B. Su, W. Tong, and Y. Xu, "An efficient polynomial-time approximation scheme for parallel multi-stage open shops," arXiv preprint arXiv:2205.14407, 2022.

[35] S. H. Shams Lahroudi, F. Mahalleh, and S. Mirkamali, "Multiobjective parallel algorithms for solving biobjective open shop scheduling problem," Complexity, vol. 2022, no. 1, p. 5043058,2022.

[36] A. A. Rahmani Hosseinabadi, J. Vahidi, B. Saemi, A. K. Sangaiah, and M. Elhoseny, "Extended genetic algorithm for solving open-shop scheduling problem," Soft computing, vol. 23, pp. 5099–5116, 2019.

[37] G. Ni and L. Chen, "Improved scheduling for the three-machine proportionate open shop and mixed shop minimum makespan problems," IEEE Access, vol. 8, pp. 186 805–186 812, 2020.

[38] K. Nip and Z. Wang, "A complexity analysis and algorithms for two-machine shop scheduling problems under linear constraints," Journal of Scheduling, vol. 26, no. 6, pp. 543–558, 2023.

[39] I. G. Smit, J. Zhou, R. Reijnen, Y. Wu, J. Chen, C. Zhang, Z. Bukhsh, W. Nuijten, and Y. Zhang, "Graph neural networks for job shop scheduling problems: A survey," arXiv preprint arXiv:2406.14096, 2024.

[40] K.-H. Ho, J.-Y. Cheng, J.-H. Wu, F. Chiang, Y.-C. Chen, Y.-Y. Wu, and I.-C. Wu, "Residual scheduling: A new reinforcement learning approach to solving job shop scheduling problem," IEEE Access, 2024.

[41] A. H. Lal, V. KR, and A. N. Haq, "The mean flow time in open shop scheduling," Journal of Advances in Management Research, vol. 17, no. 2, pp. 251–261, 2020.

[42] A. Kekli, H. E. Nouri, and O. B. Driss, "Hybrid cat swarm optimisation with genetic algorithm to solve the open shop scheduling with vehicle routing problem," 2023.