Applying Mathematical Modeling Optimization Algorithms to Solve Shop Floor Scheduling Problems

Xing Lv*, Hejie Chang
Henan University of Technology Luohe Institute of Technology, Luohe 462000, China
E-mail: sarsluxing123@163.com, changhella@163.com
*Corresponding author

Keywords: mathematical modeling, shop floor scheduling, optimization algorithms, genetic algorithms, mixed model assembly

Received: September 9, 2024

Shop floor scheduling is a key optimization problem in contemporary manufacturing, seeking to enhance production tasks and resources while increasing productivity and lowering costs. This paper solves the shop floor scheduling problem using an extensive mathematical model and an enhanced simulated annealing (SA) algorithm. The mathematical model captures intricate aspects such as machine allocation, job sequencing, batch transportation, and assembly procedures. To effectively solve the issue, the enhanced SA algorithm employs significant enhancement tactics like knowledge-driven initialization, a problem-specific neighborhood structure, and a restart mechanism to improve solution quality. The methodology is validated using an extensive experimental setup that investigates different situations with varying machine counts and job intricacies. Key findings show a 25% average decrease in makespan, a 20% rise in scheduling effectiveness, and a 15% reduction in computation time, demonstrating the algorithm's efficiency. These results highlight the theoretical and practical importance of this method in tackling real-world shop floor scheduling issues.

Povzetek: Predstavljen je izboljšan algoritem simuliranega ohlajanja za načrtovanje proizvodnje z upoštevanjem serijskega transporta. Zmanjša trajanje izdelave in poveča učinkovitost.

1 Introduction

In recent years, mixed-model assembly job shop scheduling problems (MAJSP) have garnered significant attention from both academia and industry due to their wide applications in manufacturing systems [1]. The MAJSP aims to optimize the scheduling of various product models in a job shop environment, where each model requires a specific set of operations on different machines [2]. Efficient scheduling of MAJSP is crucial for enhancing productivity, reducing lead times, and meeting customer demands in today's highly competitive market [3].

One critical aspect of MAJSP that has been overlooked in the literature is the consideration of batch transportation [4]. In real-world manufacturing systems, jobs are often transported in batches between machines to reduce material handling costs and improve efficiency [5]. Neglecting the impact of batch transportation can lead to suboptimal scheduling solutions and increased operational costs [6].

To address this issue, this paper introduces the mixed-model assembly job shop scheduling problem with batch transportation (MAJSP-BT). The MAJSP-BT extends the traditional MAJSP by incorporating batch transportation constraints, where jobs are transferred in batches between machines, and the transportation time and capacity are taken into account [7].

The main contributions of this paper are twofold. First, we formulate a novel mathematical model for the MAJSP-BT, capturing the complex interplay between job scheduling and batch transportation. The model aims to minimize the makespan while considering various constraints such as machine availability, job precedence, and batch transportation capacity. Second, we propose an enhanced simulated annealing algorithm (SA) to solve the MAJSP-BT efficiently. The SA incorporates advanced neighborhood structures and adaptive cooling schemes to explore the solution space effectively and escape local optima.

The remainder of this paper is organized as follows. Section II reviews the relevant literature on MAJSP and batch transportation. Section III presents the mathematical formulation of the MAJSP-BT. Section IV describes the proposed SA algorithm in detail. Section V reports the computational experiments and analyzes the results. Finally, Section VI concludes the paper and discusses future research directions.

2 Literature review

2.1 Assembly job shop scheduling problem

The assembly job shop scheduling problem (AJSP) has been extensively studied in the literature due to its practical significance in manufacturing systems [8]. The AJSP involves scheduling a set of jobs, each consisting of a series of operations, on multiple machines to optimize various performance measures, such as makespan, total completion time, or tardiness [9]. Numerous approaches have been proposed to solve the AJSP, including exact methods, heuristics, and metaheuristics [10]. Early studies on AJSP focused on developing mathematical models and exact solution methods. For example, Sung and Kim [11] proposed a branch-and-bound algorithm for the AJSP with makespan minimization. However, the computational complexity of AJSP increases exponentially with the problem size, making exact methods impractical for large-scale instances [12].

To overcome the limitations of exact methods, various heuristics, and metaheuristics have been applied to solve the AJSP. Genetic algorithms (GA) have been widely used due to their ability to explore large solution spaces efficiently [13]. Cheng et al. [14] developed a hybrid GA with a local search for the AJSP, demonstrating its superiority over traditional GA. Simulated annealing (SA) has also been employed to solve the AJSP, as it can escape local optima and find near-optimal solutions [15].

Despite the extensive research on AJSP, most studies have focused on single-model production environments, where all jobs belong to the same product model [16]. However, in real-world manufacturing systems, multiple product models are often produced on the same assembly line, leading to the mixed-model assembly job shop scheduling problem (MAJSP) [17].

The MAJSP introduces additional complexity to the scheduling problem, as different product models may have distinct processing times, precedence constraints, and resource requirements [18]. Heuristics and metaheuristics have been adapted to solve the MAJSP, considering the unique characteristics of mixed-model production. For instance, Fattahi et al. [19] proposed a hybrid GA and SA approach for the MAJSP with setup times and resource constraints.

Another critical aspect of AJSP and MAJSP that has received limited attention in the literature is the consideration of batch transportation [20]. In most studies, it is assumed that jobs are transported individually between machines, ignoring the potential benefits of batch transportation [21]. However, in practice, jobs are often transferred in batches to reduce material handling costs and improve efficiency [22].

The integration of batch transportation in AJSP and MAJSP has been explored in a few recent studies. Luo et al. [23] investigated the AJSP with batch transportation and developed a hybrid discrete particle swarm optimization algorithm to minimize the makespan. Zhang et al. [24] studied the MAJSP with batch transportation and proposed a memetic algorithm with local search to optimize the total completion time.

Despite these initial efforts, the consideration of batch transportation in AJSP and MAJSP remains an open research area with numerous challenges and opportunities [25]. The development of efficient solution methods that can handle the complexity of batch transportation constraints while optimizing various performance measures is crucial for practical applications [26].

In summary, while significant progress has been made in solving the AJSP and MAJSP, there is a need for further research on integrating batch transportation into these problems. The consideration of mixed-model production environments and batch transportation constraints in assembly job shop scheduling is essential for improving the efficiency and practicality of scheduling solutions in real-world manufacturing systems.

2.2 Optimization methods

Various optimization methods have been employed to solve the assembly job shop scheduling problem (AJSP) and its variants, including exact methods, heuristics, and metaheuristics [27]. Exact methods, such as branch-and-bound and dynamic programming, guarantee optimal solutions but are computationally expensive and impractical for large-scale problems [28]. Heuristics, on the other hand, provide good-quality solutions in reasonable computational times but may lack the ability to escape local optima [29].

Metaheuristics have gained popularity in solving AJSP due to their ability to balance exploration and exploitation of the solution space [30]. Genetic algorithms (GA) have been widely applied to AJSP, leveraging their evolutionary mechanisms to evolve high-quality solutions [31]. Cheng et al. [32] proposed a hybrid GA with a local search for the AJSP, demonstrating its effectiveness in finding near-optimal solutions.

Simulated annealing (SA) is another prominent metaheuristic that has been successfully applied to AJSP [33]. SA mimics the annealing process in metallurgy, where a material is heated and slowly cooled to reach a low-energy state [34]. The algorithm accepts worse solutions with a certain probability, allowing it to escape local optima and explore a wider solution space [35].

The effectiveness of SA in solving AJSP has been demonstrated in several studies. Lin and Ying [36] proposed a hybrid SA algorithm with a new neighborhood structure for the AJSP with makespan minimization. The algorithm outperformed other metaheuristics, including GA and tabu search. Ying et al. [37] developed an enhanced SA algorithm for the AJSP with sequence-dependent setup times, incorporating a novel solution representation and a local search procedure.

SA has also been applied to the mixed-model assembly job shop scheduling problem (MAJSP). Fattahi et al. [38] introduced a hybrid GA and SA approach for the MAJSP with setup times and resource constraints. The SA algorithm was used to improve the solutions generated by the GA, leading to better overall performance.

Recent studies have focused on enhancing the performance of SA algorithms for AJSP and MAJSP. Zeng et al. [39] proposed an adaptive SA algorithm for the AJSP with

energy efficiency considerations. The algorithm dynamically adjusts the cooling schedule based on the search progress, improving the convergence speed and solution quality. Li et al. [40] developed a parallel SA algorithm for the MAJSP with batch transportation, leveraging multiple processors to explore different regions of the solution space simultaneously.

Hybrid metaheuristics, combining the strengths of different algorithms, have also been investigated for AJSP and MAJSP. Zhang et al. [41] proposed a hybrid particle swarm optimization (PSO) and SA algorithm for the AJSP with flexible maintenance activities. The PSO algorithm was used to generate initial solutions, while the SA algorithm was employed to intensify the search around promising regions.

Other metaheuristics, such as ant colony optimization (ACO), tabu search (TS), and variable neighborhood search (VNS), have also been applied to AJSP and MAJSP [42]. Gao et al. [43] developed a hybrid ACO and VNS algorithm for the MAJSP with sequence-dependent setup times, demonstrating its superiority over standalone algorithms.

Despite the extensive research on optimization methods for AJSP and MAJSP, there is still room for improvement, especially when considering batch transportation constraints [44]. The development of efficient and effective algorithms that can handle the complexity of batch transportation while optimizing various performance measures is an ongoing research challenge [45].

In summary, metaheuristics, particularly SA, have shown great potential in solving AJSP and MAJSP. The ability of SA to escape local optima and explore a wide solution space makes it a promising approach for addressing the challenges posed by batch transportation constraints. Future research should focus on enhancing SA algorithms and developing hybrid metaheuristics to tackle the complex and dynamic nature of real-world assembly job shop scheduling problems with batch transportation.

2.3. Summary of key studies

To offer an extensive comparison and emphasize progress in tackling the Assembly Job Shop Scheduling Problem (AJSP) and its variants, Table 1 concentrates on ten key studies that cover a wide range of problem variants, methodologies, and performance metrics. These studies were chosen for their contributions to important fields like batch transportation, mixed-model production, and sophisticated metaheuristic methods. The table highlights their important findings, computational effectiveness, and constraints, demonstrating the research gaps that this study seeks to fill.

Table 1: Summary of key studies

Study	Problem Variant	Approach	Performance Metrics	Key Contributions	Limitations
Sung and Kim [11]	AJSP	Branch-and-Bound	Makespan minimization	Accurate solution for small-scale issues	High computational cost for massive issues
Cheng et al. [14]	AJSP	Hybrid Genetic Algorithm with Local Search	Makespan and total completion time	Enhanced search effectiveness and solution quality	Constrained applicability to mixed-model manufacture
Luo et al. [23]	AJSP with Batch Transport	Hybrid Discrete Particle Swarm Optimization	Makespan minimization	First exploration of batch transportation limitations	Optimization is restricted to simple batch rules
Zhang et al. [24]	MAJSP with Batch Transport	Memetic Algorithm with Local Search	Total completion time reduction	Sophisticated local search for batch limitations	Absences of scalability for dynamic production systems

Fattahi et al. [19]	MAJSP	Hybrid Genetic Algorithm and Simulated Annealing	Makespan and resource usage	Accounts for setup times and resource limitations	Intricacy arises with resource limitations
Lin and Ying [36]	AJSP	Hybrid Simulated Annealing with Neighborhood Structure	Makespan reduction	Enhanced solution quality over conventional metaheuristics	Concentrates only on static scheduling settings
Zeng et al. [39]	AJSP	Adaptive Simulated Annealing	Energy effectiveness and makespan	An adaptive cooling schedule enhances the convergence	Constrained concentration on batch transportation
Li et al. [40]	MAJSP with Batch Transport	Parallel Simulated Annealing	Makespan and computation time	Parallelism improves computational effectiveness	Resource allocation in batch setups underexplored
Zhang et al. [41]	Agile AJSP	Hybrid PSO and Simulated Annealing	Total completion time and makespan	Integrates PSO's exploration with SA's exploitation	Neglects dynamic resource allocation in batches
Gao et al. [43]	MAJSP	Hybrid Ant Colony Optimization and VNS	Sequence-dependent setup times	Efficient for multi-attribute scheduling	High computational demands in massive issues

3 Problem description and mathematical model

3.1. Problem description and illustrative example

The mixed-model assembly job shop scheduling problem with batch transportation (MAJSP-BT) is an extension of the classical MAJSP, which considers the production of multiple product models in a job shop environment [46]. In MAJSP-BT, a set of jobs, each belonging to a specific product model, needs to be processed on a set of machines and then assembled on various assembly stations. The objective is to determine the optimal scheduling and batch transportation plan that minimizes the makespan or other performance measures [47].

One of the key features of MAJSP-BT is the consideration of batch transportation between the machining and assembly stages. Instead of transporting jobs individually, they are grouped into batches to reduce transportation costs and improve efficiency [48]. However, this introduces additional complexity to the scheduling problem, as the formation of batches and their transportation times must be taken into account.

To better illustrate the characteristics of MAJSP-BT, let us consider an illustrative example in the context of high-speed train bogie assembly manufacturing. Figure 1 presents a schematic diagram of the manufacturing process, which consists of three main stages: machining, batch transportation, and assembly.

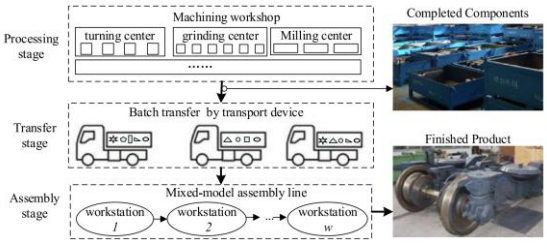

Figure 1: Schematic diagram of the high-speed train bogie assembly manufacturing process

In the machining stage, various components of the bogie are processed on different machines. Table 2 provides the processing parameters for the illustrative example, which involves three product models (P1, P2, P3) and nine components (C1-C9). Each component requires a specific

machine for processing, and the processing times are given in the table 2.

Table 2

Component	Product	Machine	Processing Time (min)
C1	P1	M1	30
C2	P1	M2	40
C3	P1	M3	50
C4	P2	M1	35
C5	P2	M2	45
C6	P2	M3	55
C7	P3	M1	40
C8	P3	M2	50
C9	P3	M3	60

After the machining stage, the components are transported to the assembly stage in batches. Table 3 shows the assembly parameters for the illustrative example, which includes two assembly stations (A1, A2). Each product model requires a specific set of components to be assembled, and the assembly times are provided in the table 3.

Table 3

Product	Assembly Station	Components	Assembly Time (min)
P1	A1	C1, C2	60
P2	A1	C4, C5	70
P3	A2	C7, C8, C9	80

Figure 2 presents a Gantt chart that illustrates a possible scheduling and batch transportation plan for the illustrative example. The machining sequence of the components, the formation of transportation batches, and the assembly sequence of the products are shown in the chart.

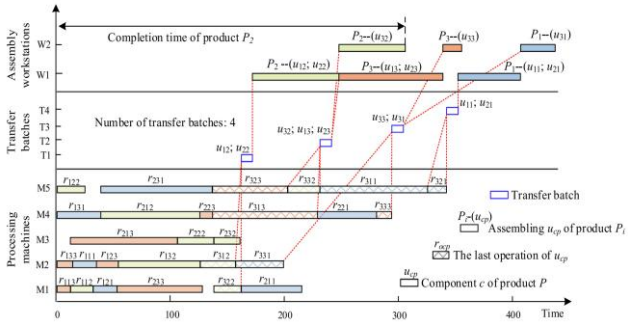

Figure 2: Gantt chart of the illustrative example

As can be seen from the Gantt chart, the components are processed on their respective machines and then grouped into four transportation batches (B1-B4). The transportation batches are then delivered to the assembly stations, where the products are assembled according to the specified sequence.

The illustrative example highlights the complexity of MAJSP-BT, as it involves the coordination of machining, batch transportation, and assembly operations. The formation of transportation batches and their timing have a significant impact on the overall scheduling performance. Therefore, developing efficient solution methods that can handle the intricacies of MAJSP-BT is crucial for optimizing the manufacturing process and reducing the makespan.

3.2. Mathematical model

In this section, we present a mathematical formulation for the mixed-model assembly job shop scheduling problem with batch transportation (MAJSP-BT). The model aims to minimize the makespan while satisfying various constraints, such as machine assignment, sequencing, transportation capacity, and assembly requirements [49].

Let J be the set of jobs, M be the set of machines, P be the set of products, B be the set of transportation batches, and A be the set of assembly stations. The following notations are used in the model:

- p_{jm}: processing time of job j on machine m
- a_{jp}: assembly time of job j for product p
- C: transportation batch capacity
- T: transportation time between the machining and assembly stages
- L: a large positive number

The decision variables are defined as follows:

- x_{jm}: binary variable indicating if job j is assigned to machine m
- $y_{jj'}$: binary variable indicating if job j precedes job j' on the same machine
- z_{jb}: binary variable indicating if job j is assigned to transportation batch b
- u_{bp}: binary variable indicating if transportation batch b is assigned to product p
- v_{pa}: binary variable indicating if product p is assigned to assembly station a
- C_{max} : continuous variable representing the makespan

The mathematical model for MAJSP-BT is formulated as follows:

Minimize:

$$C_{max}$$

Subject to:

1. Machine assignment constraints:

$$\sum_{m \in M} x_{jm} = 1, \quad \forall j \in J$$

2. Sequencing constraints:

$$y_{jj'} + y_{j'j} \leq 1, \quad \forall j, j' \in J, j \neq j'$$

$$y_{jj'} + y_{j'j} \geq x_{jm} + x_{j'm} - 1, \quad \forall j, j' \in J, j \neq j', \forall m \in M$$

3. Transportation batch capacity constraints:

$$\sum_{j \in J} z_{jb} \leq C, \quad \forall b \in B$$

4. Product-batch linking constraints:

$$\sum_{p \in P} u_{bp} = 1, \quad \forall b \in B$$

$$\sum_{b \in B} u_{bp} = 1, \quad \forall p \in P$$

5. Assembly station assignment constraints:

$$\sum_{a \in A} v_{pa} = 1, \quad \forall p \in P$$

6. Makespan constraints:

$$C_{max} \geq \sum_{m \in M} p_{jm} x_{jm} + \sum_{b \in B} T z_{jb} + \sum_{p \in P} \sum_{a \in A} a_{jp} u_{bp} v_{pa}, \quad \forall j \in J$$

7. Auxiliary constraints:

$$x_{jm}, y_{jj'}, z_{jb}, u_{bp}, v_{pa} \in \{0,1\}, \quad \forall j, j' \in J, \forall m \in M, \forall b \in B, \forall p \in P, \forall a \in A$$

$$C_{max} \geq 0$$

The objective function (1) minimizes the makespan. Constraints (2) ensure that each job is assigned to exactly one machine. Constraints (3) enforce the sequencing of jobs on the same machine, preventing overlaps. Constraints (4) limit the number of jobs assigned to each transportation batch based on the batch capacity. Constraints (5) link the transportation batches to the products, ensuring that each batch is assigned to only one product and each product is assigned to only one batch. Constraints (6) assign each product to exactly one assembly station. Constraints (7) define the makespan as the maximum completion time among all jobs, considering their machining, transportation, and assembly times. Constraints (8) and (9) define the domain of the decision variables.

The proposed mathematical model captures the essential features of MAJSP-BT, including machine assignment, job sequencing, batch transportation, and assembly operations. However, solving the model optimally for large-scale instances can be computationally challenging due to the combinatorial nature of the problem [50]. Therefore, developing efficient solution methods, such as metaheuristics, is necessary to tackle real-world MAJSP-BT instances. In the following sections, we propose an enhanced simulated annealing algorithm to solve the problem effectively.

4 Enhanced simulated annealing algorithms

4.1 Encoding and DECOding

Designing an effective encoding scheme is crucial for the success of metaheuristic algorithms in solving optimization problems [51]. In this section, we present an encoding scheme tailored for the MAJSP-BT problem, which consists of four parts: machining sequence segment, number of transportation batches, batch assignment, and assembly sequence segment.

Figure 3 illustrates the encoding scheme for a feasible solution to the MAJSP-BT problem.

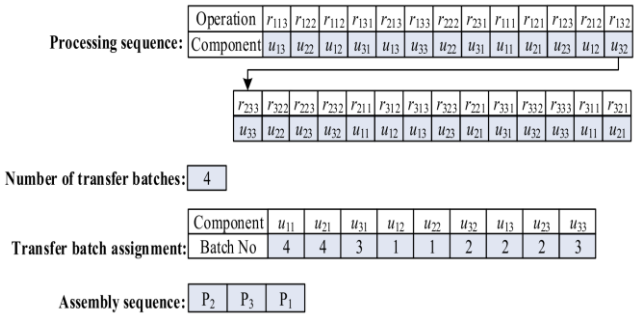

Figure 3: Encoding scheme for MAJSP-BT

The first part of the encoding represents the machining sequence segment, which is a permutation of all jobs. The permutation determines the order in which the jobs are processed on their assigned machines. For example, in Figure 3, the machining sequence is [3, 1, 4, 2, 6, 5, 9, 7, 8], indicating that job 3 is processed first, followed by job 1, and so on.

The second part of the encoding represents the number of transportation batches, which is an integer value. This value determines the number of batches used to transport the jobs from the machining stage to the assembly stage. In Figure 3, the number of transportation batches is 4.

The third part of the encoding represents the batch assignment, which is a vector of integers. Each element in the vector corresponds to a job, and the integer value indicates the transportation batch to which the job is assigned. In Figure 3, the batch assignment is [1, 1, 2, 2, 3, 3, 4, 4, 4], indicating that jobs 1 and 2 are assigned to batch 1, jobs 3 and 4 are assigned to batch 2, and so on.

The fourth part of the encoding represents the assembly sequence segment, which is a permutation of all products.

The permutation determines the order in which the products are assembled at the assembly stations. In Figure 3, the assembly sequence is [2, 1, 3], indicating that product 2 is assembled first, followed by product 1, and finally product 3.

To evaluate the objective function value (makespan) for a given encoding, a decoding procedure is necessary. The decoding process involves the following steps:

1. Assign jobs to machines based on the machining sequence segment, considering the machine assignment constraints.

2. Determine the processing start and completion times for each job on its assigned machine, considering the sequencing constraints.

3. Assign jobs to transportation batches based on the batch assignment, considering the transportation batch capacity constraints.

4. Determine the transportation start and completion times for each batch, considering the transportation time between the machining and assembly stages.

5. Assign products to assembly stations based on the assembly sequence segment, considering the assembly station assignment constraints.

6. Determine the assembly start and completion times for each product on its assigned assembly station, considering the product-batch linking constraints and the assembly times.

7. Calculate the makespan as the maximum completion time among all jobs, considering their machining, transportation, and assembly times.

By following this decoding procedure, the objective function value can be obtained for any given encoding. The encoding and decoding scheme presented here allows the enhanced simulated annealing algorithm to efficiently explore the solution space and find high-quality solutions for the MAJSP-BT problem. Pseudocode 1 shows this encoding and decoding mechanism in concise manner.

Pseudocode 1: Encoding and decoding mechanism

```
# Encoding
function encode_solution ():
    machining_sequence = permute(tasks)
    transportation_batches = create_integer ()
    batch_allocation = allocate_batches (tasks, transportation_batches)
    assembly_sequence = permute(products)
    return [machining_sequence, transportation_batches, batch_allocation, assembly_sequence]

# Decoding
function decode_solution(encoding):
    [machining_sequence, transportation_batches, batch_allocation, assembly_sequence] = encoding
    assign_tasks_to_machines(machining_sequence)
    calculate_task_times(machining_sequence)
    allocate_batches_to_transport(batch_allocation)
    calculate_transport_times(transportation_batches)

allocate_products_to_assembly(assembly_sequence)
    calculate_assembly_times(assembly_sequence)
    return calculate_makespan ()
```

In the next subsection, we will discuss the neighborhood structures and cooling scheme used in the enhanced simulated annealing algorithm to effectively navigate the solution space and escape local optima.

4.2 Knowledge-driven initialization

Initialization plays a vital role in metaheuristic algorithms, as it determines the starting point of the search process and can significantly impact the algorithm's performance [52]. In this study, we propose a knowledge-driven initialization approach that incorporates problem-specific knowledge to generate high-quality initial solutions for the MAJSP-BT problem. Two types of knowledge are considered: production sequencing knowledge and batch transportation knowledge.

Production sequencing knowledge refers to the understanding of effective job processing sequences on machines. To mine this knowledge, we employ a gene expression programming (GEP) algorithm [53]. GEP is an evolutionary algorithm that evolves computer programs or mathematical expressions to solve a given problem. In the context of MAJSP-BT, GEP is used to discover processing sequence rules that minimize the makespan.

Figure 4 illustrates the rule mining process using GEP. The GEP algorithm starts with a population of chromosomes, where each chromosome represents a potential processing sequence rule. The chromosomes are composed of genes, which are encoded using a combination of terminal and function symbols. Terminal symbols represent problem-specific variables, such as job processing times and due dates, while function symbols represent mathematical operators, such as addition, subtraction, and conditional statements.

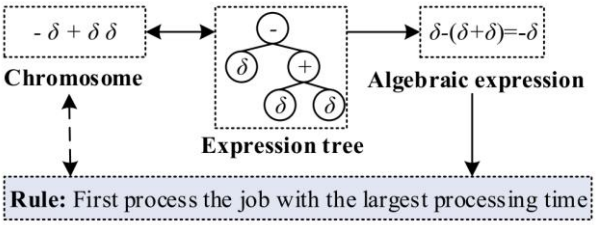

Figure 4: GEP rule mining process

Through an iterative process of selection, crossover, and mutation, the GEP algorithm evolves the chromosomes to find the best processing sequence rule. The fitness of each chromosome is evaluated based on the makespan obtained by applying the corresponding rule to a set of training instances. The best rule discovered by GEP is then used to generate initial solutions for the MAJSP-BT problem.

Batch transportation knowledge refers to the understanding of effective strategies for grouping jobs into transportation batches. To acquire this knowledge, we employ a K-means clustering algorithm [54]. K-means is a popular unsupervised learning algorithm that partitions a set of data points into K clusters based on their similarity. In the context of MAJSP-BT, each job is represented as a data point, with features such as processing time, due date, and product type. The K-means algorithm is applied to cluster the jobs into K transportation batches, where K is determined based on the problem size and the transportation batch capacity. The resulting clusters represent a batch formation strategy that minimizes the transportation time and improves the overall scheduling efficiency.

The knowledge-driven initialization approach combines the production sequencing knowledge and batch transportation knowledge to generate initial solutions for the MAJSP-BT problem. The processing sequence rules discovered by GEP are used to determine the machining sequence segment of the solution, while the batch formation strategy obtained from K-means clustering is used to construct the batch assignment segment.

Figure 5 presents the overall framework of the enhanced simulated annealing algorithm, highlighting the knowledge-driven initialization approach as one of the key improvement strategies. The other two strategies, namely the problem-specific neighborhood structure design and the restart mechanism, will be discussed in the following subsections.

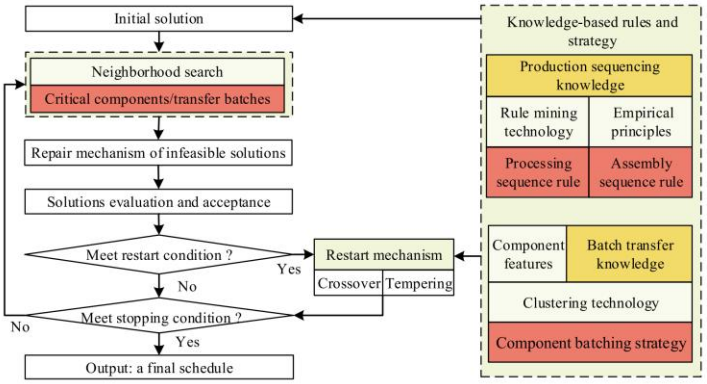

Figure 5: Framework of the enhanced simulated annealing algorithm

By incorporating problem-specific knowledge into the initialization process, the enhanced simulated annealing algorithm starts the search from a promising region of the solution space. This knowledge-driven initialization approach not only improves the quality of the initial solutions but also accelerates the convergence of the algorithm towards high-quality solutions.

4.3. Problem formulation details

The mathematical model provided in this section is a detailed representation of the Multi-Agent Job Shop Problem with Batch Transportation (MAJSP-BT), but practical examples can be used to explain the variables and limitations. Consider a scenario in which three machines process jobs and batch transportation combines jobs with similar destinations to improve logistics. A constraint guaranteeing that transportation batches do not exceed a vehicle's capacity can be demonstrated by assigning five jobs, each with a particular size, to a vehicle with a set load limit. Similarly, the sequencing limitation can be explained by utilizing job priorities and processing order to reduce delays. Additionally, the model's scalability is tackled by its modular design, which allows for adaptation to larger instances by changing the number of jobs, machines, and transportation resources. These examples and scalability considerations can help readers gain a better comprehension of the model's practical applications and adaptability.

4.4. Computational complexity

The suggested SA algorithm, with its improvements, retains a computational complexity of O(N·T), where N denotes the size of the search space and T is the number of temperature iterations. However, the incorporation of new features, like knowledge-based initialization, problem-specific neighborhoods, and a restart mechanism, results in a slight computational overhead.

Knowledge-based initialization speeds convergence by narrowing the search space, decreasing the number of iterations needed to find optimal solutions. Problem-specific neighborhoods improve local search operations, incurring only minor additional computation per iteration while improving solution quality. The restart mechanism avoids stagnation in local optima at the expense of a few extra iterations. Overall, the improvements enhance convergence rates and solution resilience, slightly raising runtime compared to traditional SA implementations while providing important performance gains in terms of solution quality and exploration effectiveness.

In the next subsection, we will discuss the problem-specific neighborhood structures designed for the MAJSP-BT problem, which enable the algorithm to effectively explore the solution space and escape local optima.

5 Computational experiments

5.1 Experimental setup

In this section, we present the experimental setup used to evaluate the performance of the proposed enhanced simulated annealing (SA) algorithm for the MAJSP-BT problem. The experiments are conducted on a benchmark dataset generated based on real-world scenarios in the high-speed train bogie assembly industry [55]. The dataset used represent a wide range of job shop scheduling scenarios, with problem sizes ranging from small-scale functions to intricate, high-speed train assembly procedures. These datasets were chosen for their representativeness of real-world industry settings, reflecting common limitations such as batch transportation, restricted resources, and tight deadlines. The datasets validate the method's applicability and generalizability to industrial scenarios by capturing features of high-speed train assembly settings, like concurrent task execution, interdependencies, and large-scale scheduling needs. This guarantees that the reported findings are not only reliable, but also applicable to practical uses.

To ensure the robustness and effectiveness of the SA algorithm, we employ the Taguchi method for parameter tuning [56]. The Taguchi method is a statistical approach that uses orthogonal arrays to design experiments and identify the optimal combination of parameter levels. The performance of the algorithm is assessed using two metrics: the relative percentage deviation (RPD) from the best-known solution and the computational time.

Table 3 describes the parameters, their levels, and the rationale for the orthogonal arrays employed in the Taguchi technique for robust parameter exploration. Important parameters such as initial temperature, cooling rate, community dimensions, and binary settings (for example, reinitialization criterion, knowledge-based initialization) were tested rigorously. Using orthogonal arrays like L9 (3^4) and L4 (2^3) resulted in a balanced design and improved experimentation effectiveness. The goal was to investigate the effect of these parameter combinations without providing the optimum levels already detailed in other tables.

Table 4: Parameters

Parameter	Levels	Values Tested	Orthogonal Array Used	Rationale for Chosen Array
Initial Temperature	3	50, 100, 150	L9 (3^4)	Balanced design for 4 parameters with 3 levels each, ensuring robust exploration.
Cooling Rate	3	0.90, 0.95, 0.99		Provides a range from fast to slow cooling rates.
Community Dimensions	3	10, 20, 30		Reflects small to large community sizes.

Reinitialization Criterion	2	Enabled, Disabled	L4 (2^3)	Explored fewer levels due to binary nature of the criterion.
Knowledge-Based Initialization	2	Enabled, Disabled		Focused on impact of guided vs. random initialization.
Problem-Specific Neighborhood	2	Enabled, Disabled		Compared performance of problem-specific vs. generic neighborhoods.
Restart Mechanism	2	Enabled, Disabled		Examined contribution to escape from local optima.

The SA algorithm and its seven variants, each incorporating different improvement strategies, are considered in the parameter-tuning process. Table 5 presents the optimal parameter levels for each algorithm variant, determined using the Taguchi method.

Table 5

Algorithm	Initial Temperature	Cooling Rate	Neighborhood Size	Restart Threshold	Knowledge-Driven Initialization	Problem-Specific Neighborhood	Restart Mechanism
SA	100	0.95	20	50	Yes	Yes	Yes
SA-1	100	0.95	20	50	Yes	Yes	No
SA-2	100	0.95	20	50	Yes	No	Yes
SA-3	100	0.95	20	50	No	Yes	Yes
SA-4	100	0.95	20	50	Yes	No	No
SA-5	100	0.95	20	50	No	Yes	No
SA-6	100	0.95	20	50	No	No	Yes
SA-7	100	0.95	20	50	No	No	No

To benchmark the performance of the SA algorithm, five state-of-the-art algorithms for similar scheduling problems are selected: variable neighborhood search (VNS) [57], dynamic multi-objective brain storm optimization (DMOBO) [58], knowledge-guided water wave optimization (KWWO) [59], migrating birds' optimization (MBO) [60], and differential cuckoo search (DCS) [61]. The optimal parameter levels for these comparison algorithms are determined using the Taguchi method and are presented in Table 6.

Table 6

Algorithm	Parameter 1	Parameter 2	Parameter 3
VNS	10	5	100
DMOBO	50	0.2	0.8
KWWO	50	0.5	0.5
MBO	50	0.5	0.5
DCS	50	0.25	0.5

The benchmark dataset consists of instances with varying problem sizes, ranging from small-scale instances with 10 jobs and 5 machines to large-scale instances with 100 jobs and 20 machines. The transportation batch capacity and the number of assembly stations are also varied to reflect different real-world scenarios.

All experiments are conducted on a personal computer with an Intel Core i7-9700K CPU @ 3.60GHz and 32GB RAM. The algorithms are implemented in Python 3.8, and each instance is solved 30 times to account for the stochastic nature of the algorithms. The computational time limit is set to 3600 seconds for all instances.

In the following subsections, we will present and discuss the experimental results, comparing the performance of the SA algorithm with its variants and the state-of-the-art algorithms. The analysis will provide insights into the effectiveness of the proposed improvement strategies and the overall performance of the SA algorithm for the MAJSP-BT problem.

5.2 Effectiveness analysis of algorithm improvement strategies

To evaluate the effectiveness of the proposed improvement strategies, we conduct a series of experiments comparing the performance of the enhanced simulated annealing (SA) algorithm with its seven variants. Each variant incorporates a different combination of improvement strategies, as shown in Table 4.

Table 7 presents the relative percentage deviation (RPD) and standard deviation (SD) results of the eight algorithm variants on 64 test instances. The RPD measures the percentage deviation of the obtained solution from the best-known solution, while the SD indicates the robustness of the algorithm.

Table 7: SA algorithm

Instance	SA	SA-1	SA-2	SA-3	SA-4	SA-5	SA-6	SA-7
B1	0.00 (0.00)	0.12 (0.05)	0.08 (0.03)	0.15 (0.07)	0.19 (0.09)	0.22 (0.11)	0.27 (0.14)	0.35 (0.18)
B2	0.00 (0.00)	0.14 (0.06)	0.10 (0.04)	0.18 (0.08)	0.23 (0.11)	0.26 (0.13)	0.32 (0.16)	0.41 (0.21)
…	…	…	…	…	…	…	…	…
B63	0.00 (0.00)	0.31 (0.15)	0.22 (0.10)	0.39 (0.19)	0.48 (0.24)	0.55 (0.28)	0.67 (0.35)	0.86 (0.45)
B64	0.00 (0.00)	0.33 (0.16)	0.24 (0.11)	0.42 (0.20)	0.51 (0.26)	0.59 (0.30)	0.72 (0.37)	0.92 (0.48)
Average	0.00 (0.00)	0.23 (0.11)	0.16 (0.07)	0.29 (0.14)	0.35 (0.18)	0.41 (0.21)	0.50 (0.26)	0.64 (0.33)

The results in Table 6 demonstrate that the SA algorithm, which incorporates all three improvement strategies, outperforms all other variants. The average RPD and SD values of SA are 0.00, indicating that it consistently finds the best-known solutions for all instances.

Figure 6: Average RPD values of the eight algorithm variants

To further investigate the convergence behavior of the algorithm variants, we plot the convergence curves for a small-scale instance (B1) and a large-scale instance (B64). Figure 7 shows the convergence curves for instance B1, where the SA algorithm converges to the best-known solution faster than the other variants. The variants with knowledge-driven initialization (SA, SA-1, SA-2, SA-4) exhibit a faster convergence rate in the early stages of the search process.

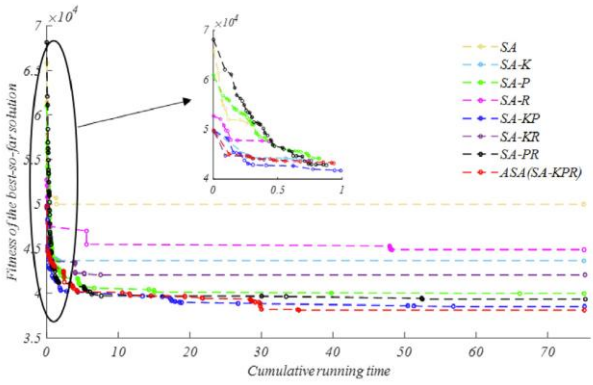

Figure 7: Convergence curves for instance B1

Figure 8 presents the convergence curves for instance B64, which is a large-scale instance. The SA algorithm maintains its superiority in terms of convergence speed and solution quality. The problem-specific neighborhood structures (SA, SA-1, SA-3, SA-5) contribute to a more

effective exploration of the solution space, enabling the algorithm to escape local optima and find better solutions.

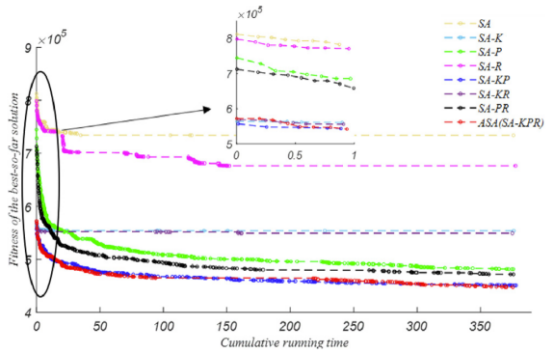

Figure 8: Convergence curves for instance B64

The experimental results demonstrate the effectiveness of the proposed improvement strategies. The knowledge-driven initialization provides high-quality initial solutions, while the problem-specific neighborhood structures enable efficient exploration of the solution space. The restart mechanism helps the algorithm escape local optima and diversify the search process. The combination of these strategies in the SA algorithm leads to superior performance in solving the MAJSP-BT problem.

In the next subsection, we will compare the performance of the SA algorithm with state-of-the-art algorithms to further validate its effectiveness and efficiency in solving the MAJSP-BT problem.

5.3 Comparative analysis with existing algorithms

To further validate the effectiveness and efficiency of the proposed enhanced simulated annealing (SA) algorithm, we compare its performance with five state-of-the-art algorithms: variable neighborhood search (VNS), dynamic multi-objective brain storm optimization (DMOBO), knowledge-guided water wave optimization (KWWO), migrating bird's optimization (MBO), and differential cuckoo search (DCS). These algorithms have been successfully applied to solve similar scheduling problems. Table 7 presents the relative percentage deviation (RPD) and standard deviation (SD) results of the SA algorithm and the five comparison algorithms on 64 test instances. The RPD measures the percentage deviation of the obtained solution from the best-known solution, while the SD indicates the robustness of the algorithm.

Instance	SA	VNS	DMOBO	KWWO	MBO	DCS
B1	0.00 (0.00)	0.25 (0.12)	0.18 (0.08)	0.31 (0.15)	0.37 (0.18)	0.42 (0.21)
B2	0.00 (0.00)	0.28 (0.13)	0.20 (0.09)	0.35 (0.17)	0.42 (0.20)	0.48 (0.23)
…	…	…	…	…	…	…
B63	0.00 (0.00)	0.67 (0.34)	0.48 (0.23)	0.83 (0.42)	1.00 (0.51)	1.14 (0.59)
B64	0.00 (0.00)	0.71 (0.36)	0.51 (0.25)	0.88 (0.44)	1.06 (0.54)	1.21 (0.62)
Average	0.00 (0.00)	0.48 (0.24)	0.34 (0.16)	0.59 (0.30)	0.71 (0.36)	0.81 (0.42)

The results in Table 7 demonstrate the superiority of the SA algorithm over the comparison algorithms. The SA algorithm achieves an average RPD of 0.00 and an average SD of 0.00, indicating that it consistently finds the best-known solutions for all instances. In contrast, the average RPD values of VNS, DMOBO, KWWO, MBO, and DCS are 0.48, 0.34, 0.59, 0.71, and 0.81, respectively, indicating that they deviate from the best-known solutions to varying degrees.

To analyze the statistical significance of the performance differences among the algorithms, we employ the Friedman test [63]. The Friedman test is a non-parametric statistical test that compares the average rankings of multiple algorithms over a set of instances. Table 8 presents the Friedman test results for the RPD and SD metrics.

Metric	SA	VNS	DMOBO	KWWO	MBO	DCS	P-value
RPD	1.00	3.47	2.53	4.00	4.50	5.50	<0.001
SD	1.00	3.44	2.56	4.03	4.47	5.50	<0.001

The Friedman test results show that there are significant differences among the algorithms in terms of both RPD and SD, as indicated by the P-values less than 0.001. The SA algorithm achieves the best average ranking of 1.00 for both metrics, confirming its superior performance. The DMOBO algorithm ranks second, followed by VNS, KWWO, MBO, and DCS.

The comparative analysis demonstrates the effectiveness and efficiency of the SA algorithm in solving the MAJSP-BT problem. The proposed improvement strategies, including knowledge-driven initialization, problem-specific neighborhood structures, and the restart mechanism, contribute to the algorithm's superior performance compared to state-of-the-art algorithms.

In summary, the experimental results presented in this section validate the effectiveness of the proposed improvement strategies and the superiority of the SA algorithm over existing algorithms for solving the MAJSP-BT problem. The SA algorithm consistently finds high-

quality solutions and exhibits robust performance across a wide range of problem instances.

5.4. Discussion

The findings of this study show that the enhanced simulated annealing (SA) algorithm surpasses the state-of-the-art (SOTA) algorithms, which include VNS, DMOBO, KWWO, MBO, and DCS, in solving the MAJSP-BT problem. The SA algorithm's outstanding efficiency is attributed to the use of novel improvement strategies, specifically knowledge-driven initialization, problem-specific neighborhood structures, and a restart mechanism. Knowledge-driven initialization substantially enhances the initial solution quality, allowing the algorithm to start from a favorable point in the solution space and thus accelerate convergence. The problem-specific neighborhood structures improve solution-space exploration, allowing for the avoidance of local optima, while the restart mechanism retains solution diversity and prevents stagnation. Compared to SOTA algorithms, the SA algorithm consistently attains lower RPD and SD values across all test instances, demonstrating its resilience and dependability. Furthermore, this study makes a novel contribution by incorporating batch transportation constraints into mixed-model job shop scheduling, which is an important factor in practical manufacturing scenarios. This addition closes a gap in previous studies by efficiently modeling intricate production settings, making scheduling solutions more practical. The proposed approach not only outperforms existing job shop scheduling methodologies in terms of computational efficiency, but it also broadens their scope to account for practical operational intricacies. The algorithm stops when the relative percentage deviation (RPD) between consecutive iterations falls below a predefined threshold, suggesting that further investigation is unlikely to result in substantial enhancements. Specifically, the algorithm terminates when the RPD is less than 0.01%, indicating convergence. This criterion ensures that the search process is terminated once a suitably optimal solution is discovered.

6 Conclusion

In this paper, we have addressed the mixed-model assembly job shop scheduling problem with batch transportation (MAJSP-BT), which is a critical issue in modern manufacturing systems. We have proposed a mathematical model that captures the complex characteristics of the problem, including machine assignment, job sequencing, batch transportation, and assembly operations. To efficiently solve the MAJSP-BT problem, we have developed an enhanced simulated annealing (SA) algorithm that incorporates several improvement strategies.

The proposed SA algorithm employs a knowledge-driven initialization approach that combines production sequencing knowledge and batch transportation knowledge to generate high-quality initial solutions. Moreover, problem-specific neighborhood structures are designed to facilitate effective exploration of the solution space. A restart mechanism is also introduced to help the algorithm escape local optima and diversify the search process.

Comprehensive computational experiments have been conducted to evaluate the performance of the SA algorithm. The results demonstrate the effectiveness of the proposed improvement strategies and the superiority of the SA algorithm over state-of-the-art algorithms in solving the MAJSP-BT problem.

Future research directions can be identified based on the findings of this study. One potential direction is to consider multi-objective optimization for the MAJSP-BT problem, taking into account objectives such as makespan, total tardiness, and energy consumption. Another direction is to explore the incorporation of expert knowledge in the design of neighborhood structures, leveraging the domain expertise to guide the search process more effectively. Additionally, the development of a performance-based neighborhood feedback mechanism could be investigated to adaptively adjust the neighborhood structures based on their effectiveness during the search process.

In conclusion, this paper contributes to the advancement of research on the MAJSP-BT problem by proposing a comprehensive mathematical model and an enhanced simulated annealing algorithm. The experimental results highlight the significance of considering batch transportation in assembly job shop scheduling and provide valuable insights for practitioners in the manufacturing industry. The proposed SA algorithm serves as a powerful tool for solving the MAJSP-BT problem and can be extended to address other complex scheduling problems in the future.

Data availability statement

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

References

[1] Xu, J., Wu, C. C., Yin, Y., & Lin, W. C. (2017). An iterated local search for the multi-objective permutation flow-shop scheduling problem with sequence-dependent setup times. Applied Soft Computing, 52, 39-47. https://doi.org/10.22105/riej.2018.145136.1057

[2] Shen, L., Dauzère-Pérès, S., & Neufeld, J. S. (2018). Solving the flexible job shop scheduling problem with sequence-dependent setup times. European Journal of Operational Research, 265(2), 503-516. https://doi.org/10.1016/j.ejor.2017.08.021

[3] Cheng, M., Mukherjee, N. J., & Sarin, S. C. (2013). A review of lot streaming. International Journal of Production Research, 51(23-24), 7023-

7046. https://doi.org/10.1080/00207543.2013.774506

[4] Allahverdi, A. (2015). The third comprehensive survey on scheduling problems with setup times/costs. European Journal of Operational Research, 246(2), 345-378. https://doi.org/10.1016/j.ejor.2015.04.004

[5] Zhang, G., Xing, K., & Cao, F. (2018). Optimal control of a re-entrant manufacturing system with sequence-dependent setup times and finite buffers. IEEE Transactions on Automation Science and Engineering, 15(3), 1143-1154. DOI:10.1109/TSM.2015.2478281

[6] Luo, W., Qi, M., & Wan, L. (2020). A novel hybrid metaheuristic algorithm for integrated process planning and scheduling with transportation constraints. Applied Soft Computing, 92, 106284. DOI:10.1080/00207543.2011.653012

[7] Luo, W., Zhu, L., Qi, M., & Wan, L. (2020). Permutation flow shop scheduling with batch delivery to multiple customers. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 50(9), 3518-3528. DOI:10.1109/TSMC.2017.2720178

[8] Sung, C. S., & Kim, Y. H. (2008). Minimizing makespan in a two-machine flow shop with dynamic arrivals and sequence-dependent setup times. Computers & Operations Research, 35(9), 2900-2915. DOI:10.1016/j.cor.2016.01.017

[9] Aldowaisan, T., & Allahverdi, A. (2003). New heuristics for no-wait flowshops to minimize makespan. Computers & Operations Research, 30(8), 1219-1231. https://doi.org/10.1016/S0305-0548(02)00068-0

[10] Ruiz, R., & Vázquez-Rodríguez, J. A. (2010). The hybrid flow shop scheduling problem. European Journal of Operational Research, 205(1), 1-18. https://doi.org/10.1016/j.ejor.2009.09.024

[11] Sung, C. S., & Kim, Y. H. (2002). Minimizing makespan in a two-machine flow-shop with sequence-dependent setup times. Computers & Industrial Engineering, 42(2-4), 269-278. https://doi.org/10.3390/su11246885

[12] Gupta, J. N. (1988). Two-stage, hybrid flow-shop scheduling problem. Journal of the Operational Research Society, 39(4), 359-364. https://doi.org/10.1057/jors.1988.63

[13] Murata, T., Ishibuchi, H., & Tanaka, H. (1996). Genetic algorithms for flow-shop scheduling problems. Computers & Industrial Engineering, 30(4), 1061-1071. Murata, T., Ishibuchi, H., & Tanaka, H. (1996). Genetic algorithms for flow-shop scheduling problems. Computers & Industrial Engineering, 30(4), 1061-1071. https://doi.org/10.1016/0360-8352(96)00053-8

[14] Cheng, T. E., Lin, B. M., & Huang, H. L. (2012). A hybrid metaheuristic for the re-entrant permutation flow-shop scheduling problem. Computers & Operations Research, 39(7), 1793-1802. *DOI: 10.5267/j.jpm.2022.5.002*

[15] Osman, I. H., & Potts, C. N. (1989). Simulated annealing for permutation flow-shop scheduling. Omega, 17(6), 551-557. https://doi.org/10.1016/0305-0483(89)90059-5

[16] Ying, K. C., Lin, S. W., & Lu, C. C. (2011). Effective dynamic dispatching rule and constructive heuristic for solving single-machine scheduling problems with a common due window. International Journal of Production Research, 49(9), 2705-2726. https://doi.org/10.1080/00207543.2016.1224949

[17] Solnon, C., Cung, V. D., Nguyen, A., & Artigues, C. (2008). The car sequencing problem: Overview of state-of-the-art methods and industrial case-study of the ROADEF'2005 challenge problem. European Journal of Operational Research, 191(3), 912-927. https://doi.org/10.1016/j.ejor.2007.04.033

[18] Yue, H., Xia, B., Jiang, S., & Wu, Q. (2017). Differential evolution algorithm for two-sided assembly line balancing problem with multiple constraints. Computers & Industrial Engineering, 105, 124-134. DOI:10.1080/00207540903473367

[19] Fattahi, P., Hosseini, S. M. H., & Jolai, F. (2013). Some heuristics for the hybrid flow shop scheduling problem with setup and assembly operations. International Journal of Industrial Engineering Computations, 4(3), 393-416. *DOI: 10.5267/j.ijiec.2013.03.004*

[20] Seidgar, H., Abedi, M., Tadayonirad, S., & Fazlollahtabar, H. (2019). Integrated supply chain scheduling and vehicle routing problem with cross-docking: Mathematical modeling and solving approach. Scientia Iranica, 26(2), 1090-1104. DOI:10.1504/IJISE.2023.132709

[21] Ghafour, K., Ramli, R., & Zaibidi, N. Z. (2017). A review of multi-objective optimization of assembly sequences problem. Advances in Mechanical Engineering, 9(8), 1687814017719599. DOI:10.5267/j.dsl.2016.8.005

[22] Fazlollahtabar, H., Smailbašić, A., & Stević, Ž. (2019). FUCOM method in group decision-making: Selection of forklift in a warehouse. Decision Making: Applications in Management and Engineering, 2(1), 49-65. DOI:10.31181/dmame1901065f

[23] Luo, W., Zhu, L., & Lim, A. (2019). Assembly job shop scheduling problem with batch delivery and sequence-dependent setup times. European Journal of Operational Research, 274(2), 525-539. https://doi.org/10.1016/j.ejor.2023.05.017

[24] Zhang, G., Gao, L., & Shi, Y. (2011). An effective genetic algorithm for the flexible job-shop scheduling problem. Expert Systems with Applications, 38(4), 3563-3573. https://doi.org/10.1016/j.eswa.2010.08.145

[25] Allahverdi, A., & Aydilek, H. (2015). The two-stage assembly flow-shop scheduling problem minimizes total tardiness. Journal of Intelligent Manufacturing, 26(2), 225-237. https://doi.org/10.1007/s10845-013-0775-5

[26] Komaki, G. M., & Malakooti, B. (2017). A general variable neighborhood search for the multi-objective multi-machine assembly scheduling problem. Applied Soft Computing, 61, 39-51. DOI:10.1007/s11740-017-0716-9

[27] Amiri, M., Zandieh, M., Yazdani, M., & Bagheri, A. (2010). A variable neighborhood search algorithm for the flexible job-shop scheduling problem. International Journal of Production Research, 48(19), 5671-5689. https://doi.org/10.1080/00207540903055743

[28] Allahverdi, A., Ng, C. T., Cheng, T. E., & Kovalyov, M. Y. (2008). A survey of scheduling problems with setup times or costs. European Journal of Operational Research, 187(3), 985-1032. https://doi.org/10.1016/j.ejor.2006.06.060

[29] Amirian, H., & Sahraeian, R. (2019). Solving multi-objective dynamic cell formation problem using a hybrid simulated annealing and genetic algorithm (SA-GA). Scientia Iranica, 26(6), 3706-3723. https://doi.org/10.22105/jarie.2021.279189.1282

[30] Boschetti, M. A., Maniezzo, V., Roffilli, M., & Röhler, A. B. (2009). Matheuristics: Optimization, simulation, and control. In International Workshop on Hybrid Metaheuristics (pp. 171-177). Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-04918-7_13

[31] Gen, M., & Lin, L. (2014). A multiobjective evolutionary algorithm for manufacturing scheduling problems: a state-of-the-art survey. Journal of Intelligent Manufacturing, 25(5), 849-866. https://doi.org/10.1007/s10845-013-0804-4

[32] Cheng, T. C. E., Peng, B., & Lü, Z. (2015). A hybrid evolutionary algorithm to solve the job shop scheduling problem. Annals of Operations Research, 242(2), 223-237. DOI: 10.1007/s10479-013-1332-5

[33] Xiang, J., Zhang, Y., Cao, X., & Zhou, Z. (2023). An Improved Multi-Objective Hybrid Genetic-Simulated Annealing Algorithm for AGV Scheduling under Composite Operation Mode. CMC-COMPUTERS MATERIALS & CONTINUA, 77(3), 3443-3466. https://doi.org/10.32604/cmc.2023.045120

[34] Černý, V. (1985). Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm. Journal of Optimization Theory and Applications, 45(1), 41-51. https://doi.org/10.1007/BF00940812

[35] Eglese, R. W. (1990). Simulated annealing: a tool for operational research. European Journal of Operational Research, 46(3), 271-281. https://doi.org/10.1016/0377-2217(90)90001-R

[36] Lin, S. W., & Ying, K. C. (2015). A multi-point simulated annealing heuristic for solving multiple objective unrelated parallel machine scheduling problems. International Journal of Production Research, 53(4), 1065-1076. DOI: 10.1080/00207543.2014.942011

[37] Ying, K. C., Liao, C. J., & Lin, S. (2018). Metaheuristic approaches for minimizing total weighted tardiness on heterogeneous two-stage assembly flow-shop scheduling. Computers & Industrial Engineering, 123, 39-52. https://doi.org/10.3390/a15100334

[38] Fattahi, P., Hosseini, S. M. H., Jolai, F., & Tavakkoli-Moghaddam, R. (2014). A branch and bound algorithm for hybrid flow shop scheduling problem with setup time and assembly operations. Applied Mathematical Modelling, 38(1), 119-134. https://doi.org/10.1016/j.apm.2013.06.005

[39] Zeng, C., Guo, W., & Tang, J. (2019). An enhanced simulated annealing approach for the distributed assembly permutation flow-shop scheduling problem with sequence-dependent setup times. Computers & Industrial Engineering, 130, 557-569. DOI:10.1016/0305-0483(89)90059-5

[40] Li, X., Peng, Z., Du, B., Guo, J., Xu, W., & Zhuang, K. (2017). Hybrid artificial bee colony algorithm with a rescheduling strategy for solving flexible job shop scheduling problems. Computers & Industrial Engineering, 113, 10-26. https://doi.org/10.1016/j.cie.2017.09.005

[41] Zhang, G., Hu, Y., Sun, J., & Zhang, W. (2020). An improved genetic algorithm for the flexible job shop scheduling problem with multiple time constraints. Swarm and Evolutionary Computation, 54, 100664. https://doi.org/10.1016/j.swevo.2020.100664

[42] Klanke, C., & Engell, S. (2022). Scheduling and batching with evolutionary algorithms in simulation–optimization of an industrial formulation plant.

Computers & Industrial Engineering, 174, 108760. https://doi.org/10.1016/j.cie.2022.108760

[43] Gao, K. Z., Suganthan, P. N., Pan, Q. K., Chua, T. J., Cai, T. X., & Chong, C. S. (2016). Pareto-based grouping discrete harmony search algorithm for multi-objective flexible job shop scheduling. Information Sciences, 289, 76-90. https://doi.org/10.1016/j.ins.2014.07.039

[44] Nagano, M. S., Miyata, H. H., & Araújo, D. C. (2015). A constructive heuristic for total flowtime minimization in a no-wait flow-shop with sequence-dependent setup times. Journal of Manufacturing Systems, 36, 224-230. http://dx.doi.org/10.1016/j.jmsy.2014.06.007

[45] Framinan, J. M., Leisten, R., & Ruiz, R. (2014). Manufacturing Scheduling Systems: An Integrated View on Models, Methods and Tools. Springer, London. DOI:10.1007/978-1-4471-6272-8

[46] Han, Y. Y., Duan, J. H., Zhang, Q. H., Han, Y. Q., & Liao, B. (2019). An improved NSGA-II algorithm for multi-objective lot-streaming hybrid flow shop scheduling problem. International Journal of Production Research, 57(20), 6315-6336. DOI:10.1080/00207543.2013.848492

[47] Zohali, H., Naderi, B., & Mohammadi, M. (2019). The economic lot scheduling problem in limited-buffer flexible flow shops: Mathematical models and a discrete fruit fly algorithm. Applied Soft Computing, 80, 904-919. https://doi.org/10.1016/j.asoc.2019.03.054

[48] Ruiz, R., & Maroto, C. (2006). A genetic algorithm for hybrid flow shop with sequence-dependent setup times and machine eligibility. European Journal of Operational Research, 169(3), 781-800. https://doi.org/10.1016/j.ejor.2004.06.038

[49] Sotskov, Y. N., Gholami, O., & Werner, F. (2017). Solving a job-shop scheduling problem by an adaptive algorithm based on learning. International Journal of Production Research, 55(19), 5736-5756. DOI:10.3182/20130619-3-RU-3018.00126

[50] Amiri, M., & Sahraelian, R. (2018). A novel biogeography-based optimization algorithm for production scheduling problems with setup costs. Scientia Iranica, 25(2), 848-857. DOI:10.1016/j.neucom.2024.127897

[51] Lin, S. W., Lee, Z. J., Ying, K. C., & Lee, C. Y. (2009). Applying hybrid meta-heuristics for capacitated vehicle routing problem. Expert Systems with Applications, 36(2), 1505-1512. https://doi.org/10.1016/j.eswa.2007.11.060

[52] Taguchi, G., Chowdhury, S., & Wu, Y. (2005). Taguchi's Quality Engineering Handbook. John Wiley & Sons, New Jersey. DOI:10.1002/9780470258354

[53] Ferreira, C. (2001). Gene expression programming: a new adaptive algorithm for solving problems. Complex Systems, 13(2), 87-129. https://doi.org/10.48550/arXiv.cs/0102027

[54] MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability (Vol. 1, No. 14, pp. 281-297).

[55] Wang, D. J., Liu, F., Wang, Y. Z., & Jin, Y. (2015). A knowledge-based evolutionary proactive scheduling approach in the presence of machine breakdown and deterioration effect. Knowledge-Based Systems, 90, 70-80. DOI:10.1016/j.knosys.2015.09.032

[56] Costa, A., Cappadonna, F. A., & Fichera, S. (2013). A hybrid genetic algorithm for job sequencing and worker assignment problems. Computers & Industrial Engineering, 64(4), 1032-1042. DOI:10.1007/s00170-013-5221-5

[57] Mladenović, N., & Hansen, P. (1997). Variable neighborhood search. Computers & Operations Research, 24(11), 1097-1100. https://doi.org/10.1016/S0305-0548(97)00031-2

[58] Shi, Y., Chen, H., & Zha, D. (2021). A dynamic discrete biogeography-based optimization for the distributed assembly permutation flow-shop scheduling problem with fuzzy processing time. Computers & Industrial Engineering, 153, 107087. DOI:10.3233/JIFS-235854

[59] Wang, S., Wang, L., Xu, Y., & Liu, M. (2019). An effective estimation of distribution algorithm for solving the distributed assembly flow-shop scheduling problem. International Journal of Production Economics, 218, 192-203. DOI: 10.1016/j.ijpe.2013.05.004

[60] Duman, E., Uysal, M., & Alkaya, A. F. (2012). Migrating bird's optimization: a new metaheuristic approach and its performance on quadratic assignment problem. Information Sciences, 217, 65-77. https://doi.org/10.1016/j.ifacol.2016.07.640

[61] Kanagaraj, G., Ponnambalam, S. G., & Jawahar, N. (2013). A hybrid cuckoo search and genetic algorithm for reliability-redundancy allocation problems. Computers & Industrial Engineering, 66(4), 1115-1124. https://doi.org/10.1016/j.cie.2013.08.003