Jurnal Teknik Industri, Vol. 28, No. 1, June 2026
ISSN 1411-2485 print / ISSN 2087-7439 online
DOI: https://doi.org/10.9744/jti.28.1.92-102

A Multi Objective Parallel Machine Scheduling Model for Bicycle Painting Problem

Dave Charlton Limantara[1], I Gede Agus Widyadana[1*]
[1)] Faculty of Industrial Engineeding, Industrial Engineering Department, Petra Christian University
Jl. Siwalankerto 121-131, Surabaya, 60236, Indonesia
Email: gede@petra.ac.id*
*Corresponding author

Abstract: This paper addresses the complex multiobjective parallel machine scheduling problem within a bicycle painting department. The painting process is characterized by sequence-dependent setup times driven by frequent color changes and the critical need to align painting output with assembly shop requirements to prevent stock accumulation. The research develops a mathematical model with two primary objectives: minimizing the sequence mismatch between painting and assembly and minimizing total setup time. Unlike prior research, this study simultaneously addresses sequence-dependent setups caused by color changes, the management of multiple parallel paint lines within daily capacity limits, and the critical requirement that painting output matches the input sequence needed for assembly. Given the NP-hard nature of the problem, a Genetic Algorithm (GA) method is proposed to solve the problem. Other methods such as Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO), and Variable Neighborhood Search (VNS), are used to compare with the GA. The methods are evaluated using real-world data from a bicycle company. Results indicate that VNS outperforms GA for the objective value. The objective value score of VNS is 91.23% compared to the Genetic Algorithm 86.72%. However, the GA runtimes are significantly faster, five times better than the VNS.

Keywords: scheduling, parallel machine, multiobjective, painting, metaheuristics.

Introduction

Production scheduling problems are frequently encountered in manufacturing systems. The common problem is how to allocate '*n*' jobs to '*m*' machines [1]. Parallel machine scheduling problems are a class of scheduling problems in which jobs can be processed simultaneously on multiple machines [2]. Parallel machine problems are particularly prevalent in modern manufacturing facilities where large-scale production demands necessitate efficient automated scheduling techniques, as manual approaches are often insufficient for optimizing job assignments [3]. There are many problem variations of parallel machines, such as makespan minimization, cost efficiency, and resource utilization, that need sophisticated mathematical modelling and algorithmic approaches to derive optimal or near-optimal solutions. The inherent complexity of parallel machine problems, often classified as NP-hard, necessitates the development of heuristic and meta-heuristic approaches to achieve efficient solutions, especially for large-scale problems [4].

Production scheduling strongly affects lead times, resource use, and the smoothness of flow between departments in a factory. In bicycle manufacturing, the painting department is often one of the hardest places to schedule, because several paint lines must handle a high mix of models, parts, and colours. When the sequence changes from one colour to another, the line may need cleaning and resetting, so the setup time depends on the job order (sequence-dependent setup). This issue is common in real paint-shop scheduling and is a major reason why good sequencing matters as much as good assignment across parallel lines [5].

In practice, these setup losses can become very costly. Too many colour changes can reduce daily throughput, push work into overtime, and create unstable completion times. In addition, painting is not an isolated process: if the completion order in painting does not match the required order in assembly, the factory can face shortages of the right painted parts even when the total painted volume appears sufficient. This creates a schedule mismatch between departments, and it is often driven by the same root cause: complex sequencing under sequence-dependent setup times and limited line capacity [5].

Submitted: May 16, 2026; Revised: June 18, 2026; Accepted: June 19, 2026

Research on paint shop scheduling problems has been done widely due to high demand at some factories, such as car and bicycle factories. There are various objective functions in paint shop scheduling, such as minimizing the emission of chemical pollutants that follow every color change process [6] and bicycle factory [7], smoothing flows through buffers and intermediate stores to meet downstream assembly requirements and avoid bottlenecks [8, 9]. The Paint Shop scheduling problem is interesting and important to be solved since it has some unique constraints characteristics such as sequence dependent setup time where switching between colors imposes non-negligible setup/cleaning times and sequencing penalties, so sequencing models treat setups as dependent on adjacent items [10], fixed hanging positions and the way parts occupy hooks or carriers constraint feasible permutations and affect how items traverse booths [8], constraint rules and the time cost of sequencing in the general assembly shop [9], hanger eligibility, hanger capacity [11] and forbidden sequence [12]. Research on paint shop problems has mostly been done in car industries, such as minimizing the number of costly changeovers of painting guns, resulting from color changes, and synchronizing those with periodic cleanings [13], and production scheduling optimization by considering constraint differences between an automotive painting workshop and a final assembly workshop [14]. The other painting scheduling problem is the automotive parts paint shop scheduling problem, which contains color arrangement restrictions, part arrangement restrictions, bracket restrictions, and multiple objectives [15]; however, this problem does not consider sequencing between the painting shop and assembly line. From painting shops to the assembly line of a car production line, which mostly comprises one color type and one part, the challenges in the previous works exhibit characteristics of a sequencing problem. The other study discusses several part arrangement limits between two production lines that do not follow a specific order.

Unlike previous papers, this study focuses on the bicycle paint shop environment, which has characteristics distinct from car production. The bicycle paint shop has to paint various components of a bicycle, and some parts need to be painted more than once. The sequence of components has to match the assembly line sequence that follows the painting shop. In this paper, the problem becomes more complex by incorporating some constraints, which are (1) sequence-dependent setups are driven by colour changes, (2) multiple parallel paint lines must be scheduled under daily capacity limits, and (3) the completion order in the painting department must be aligned with the assembly department requirements to reduce the mismatch. The problem has two objectives: first is sequencing match painting output and assembling input, and second is minimizing setup time. The bicycle manufacturing factory experiences inefficiency and instability in the painting department when scheduling a wide variety of product colours, as it creates sequence-dependent setup times. At the same time, limited parallel paint lines must also produce outputs in an order that meets the requirements of the assembly department.

Since parallel-machine scheduling with sequence-dependent setups is highly complex, exact optimization often does not scale well for industrial-sized instances. For this reason, heuristic and metaheuristic methods are widely used to find strong solutions in a reasonable time, especially when the setup behaviour depends on the sequence and when real constraints must be included [16]. These methods are also flexible. They can be extended to handle practical features such as limited setup resources, job splitting, and priority rules that reflect real operational needs.

Methods

Model Development

The problem is a real case with a bicycle company, but it can be used for any bicycle company. The problem is about scheduling at a painting shop with parallel machines. The painting process is followed by the assembling process. The problem is that assembling shops have been set up for a specific sequence. Therefore, the output of the painting shop sequence must be the same as the input of the assembling shop sequence. If the sequence is not the same, it piles up stock between those two shops. This problem needs to be minimized. On the other hand, setup time in the painting shop is dependent on the previous job. The model has some assumptions as follows:

1. Production time is known and fixed
2. Each individual job is assigned to exactly one machine among the available parallel paint lines.
3. Setup times are strictly dependent on the order of jobs.
4. The assembly department has a pre-determined, fixed input sequence that follows the customer's order.
5. The model assumes a static scheduling environment. It does not account for dynamic variabilities such as urgent job arrivals, rework, or special exceptions during the planned period.
6. The trade-off between conflicting goals (minimizing mismatch vs. minimizing setup time) can be effectively managed through fixed weightings.

Notations:

Sets	
$\boldsymbol{I, J}$	Set of jobs
$\boldsymbol{K}$	Set of machines
Parameters	
$\boldsymbol{D_i}$	Sequence determined for job $\boldsymbol{i}$ at assembly department
A	Total sequence difference
A_n	Normalized A
A_{max}	Maximum difference in the sequence value
s_{ij}	Set up time when job j is processed after job $\boldsymbol{i}$
S_{Tot}	Total setup time
S_{max}	Maximum setup time
S	Normalized total setup time
W_m	Weight of sequence objective
Ws	Weight of setup objective
M	Large positive number (Big – M)
U_{ijk}	Scheduled set up time job $\boldsymbol{i}$ processed before job $\boldsymbol{j}$ at machine $\boldsymbol{k}$
P_i	Processing time of job $\boldsymbol{i}$
Decision Variables	
$\boldsymbol{x_{i,k}}$	$\begin{cases} \mathbf{1, if\ job}\ \boldsymbol{i}\ \mathbf{is\ assigned\ to\ machine}\ \boldsymbol{k} \\ \mathbf{0}, \boldsymbol{otherwise} \end{cases}$
$\boldsymbol{y_{i,j,k}}$	$\begin{cases} \mathbf{1, if\ job}\ \boldsymbol{i}\ \mathbf{is\ assigned\ before\ job}\ \boldsymbol{j}\ \mathbf{at\ machine}\ \boldsymbol{k} \\ \mathbf{0}, \boldsymbol{otherwise} \end{cases}$
$\boldsymbol{Q_i}$	Sequence of job $\boldsymbol{i}$ as the output of painting department
C_{ik}	Completion time of job $\boldsymbol{i}$ at machine $\boldsymbol{k}$
T_{ik}	Starting time of job $\boldsymbol{i}$ at machine $\boldsymbol{k}$

Objective function

There are two objective functions in this problem. First is sequencing match painting output and assembling input, and second is minimizing setup time. The two objectives are normalized because the two objectives are measured in different units and scales. The first objective function is minimizing the difference in job sequence between the sequence of the finishing department output and the sequence of the assembly department input. The assembly department is the successor of the painting department. The function can be modelled as follows:

$$\boldsymbol{A = \sum_{i=1}^{n} |Q_i - D_i|} \tag{1}$$

Using n as the number of jobs in a set, the maximum difference can be modeled as eq (2). Since the result of the objective function must be normalized, the normalization of eq (1) is shown in eq (3).

$$\boldsymbol{A_{max} = \begin{cases} \dfrac{n^2}{2}, & if\ n\ is\ even \\ \dfrac{n^2 - 1}{2}, & if\ n\ is\ odd \end{cases}} \tag{2}$$

$$\boldsymbol{A_N = \frac{A}{A_{max}}} \tag{3}$$

Equation (2) shows the worst absolute difference value, and eq (3) shows the normalized difference value objective. The second objective function is minimizing setup time. The Objective function is shown in eq (4) and then normalized as eq (5).

$$\boldsymbol{S_{Tot} = \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{m} s_{ij} y_{ijk}} \tag{4}$$

$$\boldsymbol{S_{max} = \sum_{i=1}^{n} max\,(s_{ij}) \qquad j \in J} \tag{5}$$

$$\boldsymbol{S = \frac{S_{Tot}}{S_{max}}} \tag{6}$$

Weighting is used to reflect the strategic priorities of the manufacturing facility and to manage the inherent trade-off between painting efficiency and assembly needs. Since, from an industry perspective, higher efficiency is better, therefore by combining eq (3) and eq (6), we get the total objective function as follows:

$$\boldsymbol{Max\ z = 100\% - (W_m A_N + W_s S)\, x100\%} \tag{7}$$

Where:

$$\boldsymbol{W_m + W_s = 1} \tag{8}$$

Constraints of the model consist of completion time constraints, sequence constraints, allocation constraints, non-recursive constraints, and non-negative constraints. Those constraints can be written as follows:

$$\boldsymbol{C_{ik} = T_{ik} + P_i + U_{ijk}} \qquad \forall\, \boldsymbol{i,k} \tag{9}$$

$$\boldsymbol{T_{jk} \geq C_{ik} - M(1 - y_{ijk})} \qquad \forall\, \boldsymbol{i,k} \tag{10}$$

$$\boldsymbol{\sum_{k=1}^{m} X_{ik} = 1} \qquad \forall\, \boldsymbol{i} \tag{11}$$

$$\boldsymbol{y_{ijk} + y_{jik} \leq 1} \qquad \forall\, \boldsymbol{i,j,k} \tag{12}$$

$$\boldsymbol{\sum_{j=1}^{m} y_{jik} = x_{ik}} \qquad \forall\, \boldsymbol{i,k} \tag{13}$$

$$\boldsymbol{U_{ijk} = s_{ijk} y_{ijk}} \qquad \forall\, \boldsymbol{i,k} \tag{14}$$

$$\boldsymbol{Q_i < Q_j \ \ if\ C_i < C_j} \tag{15}$$

$$\boldsymbol{y_{ijk}, x_{ik} \geq 0} \tag{16}$$

Equation (9) shows that the completion time of job $\boldsymbol{i}$ at machine $\boldsymbol{k}$ is equal to the starting time of job $\boldsymbol{i}$ at machine $\boldsymbol{k}$ plus the operating time of job $\boldsymbol{i}$ plus the setup time. Equation (10) shows that either sequence must be of job $\boldsymbol{i}$ before $\boldsymbol{j}$ or job $\boldsymbol{j}$ before $\boldsymbol{i}$ at machine $\boldsymbol{k}$. Equation (11) shows that one job is only assigned once in all machines, and eq (12) shows the constraints of recursive jobs, if job $\boldsymbol{i}$ is scheduled before job $\boldsymbol{j}$ then job $\boldsymbol{j}$ cannot be scheduled before job $\boldsymbol{i}$. Equation (13) shows that only one sequence can be assigned to one machine. The setup time of job $\boldsymbol{i}$ before job $\boldsymbol{j}$ is applied when job $\boldsymbol{i}$ is run before job $\boldsymbol{j}$ is shown in eq (14). Equation (15) shows the sequence of job $\boldsymbol{i}$ before job $\boldsymbol{j}$, and Equation (16) is a non-negative constraint.

Since the model is a nonlinear combinatorial model and a NP-Hard model, metaheuristic methods are used. The complexity of the model arises from assignment decision, sequencing decision, setup dependency, and multi-objective decision making. The main method used is the Genetic Algorithm, and then it will be compared to other methods. A Genetic Algorithm (GA) is a population-based metaheuristic that iteratively improves candidate schedules using selection, crossover, and mutation. In scheduling, GA is widely used because schedules can be encoded as permutations, and genetic operators can explore diverse sequences efficiently. Recent job-scheduling research applies GA to flexible job shop scheduling with sequence-dependent setup times and additional time constraints, demonstrating that GA-based approaches remain effective when the schedule structure is complex and the search space is large [17]. GA has also been used in unrelated parallel machine scheduling with machine and sequence-dependent setups in industrial contexts, showing practical value when exact methods do not scale to real case sizes [18]. In this paper, some other methods are considered which are Ant Colony, Particle Swarm Optimization and Ant Colony Optimization (ACO) is a constructive metaheuristic where solutions are built step-by-step using probabilistic rules guided by pheromone trails (learning from good solutions) and heuristic information (problem-specific guidance). ACO has been used in scheduling because it naturally fits problems where a schedule can be constructed as a sequence of decisions (which operation/job to place next). Recent studies show ACO variants applied to classical scheduling settings such as open shop scheduling [19], unrelated parallel machine scheduling [20], and to flexible job shop scheduling with multiple time constraints, indicating that ACO remains relevant for job scheduling problems that include setup/transport/delivery-related constraints [21].

Particle Swarm Optimization (PSO) is originally defined for continuous optimization, but in scheduling, it is commonly adapted into Discrete PSO (DPSO) by representing a schedule as a discrete structure (often a permutation or coded sequence) and replacing position updates with discrete operators such as insertion and exchange (swap) moves. DPAO methods continue to be applied to flexible job shop variants (including multi-

objective and uncertain/complex environments), supporting the motivation for a swap-based particle update scheme when the schedule representation is a job sequence [22].

Variable Neighbourhood Search (VNS) is a local-search framework that systematically changes the neighborhood structure during search (e.g., swap, insertion, block moves, reassignment moves), which helps escape local optima that trap single-neighborhood methods. In job scheduling, VNS is frequently used either as a standalone improvement engine or as a hybrid component that intensifies solutions produced by global search methods. Recent work demonstrates strong VNS usage in flexible job shop scheduling through hybrid GA–VNS approaches [23], and in job-shop variants with parallel machines via dominance relations-based VNS [24].

The genetic algorithm is structured around several distinct evolutionary phases, executed iteratively over a defined number of generations. The following is GA algorithm structure:

a. Chromosome representation and initialization
   The algorithm employs an order-based chromosome representation. Each chromosome represents the order of job codes. Due to the nature of the production, the chromosome operates as a multiset where identical job codes may appear multiple times. The initial population is generated using a randomized approach. The baseline job dataset is shuffled randomly for each individual in the population. This ensures maximum genetic diversity at the start of the evolutionary search.

3	1	2	7	6	5	4

Figure 1. Chromosome representation

   Figure 1 shows an example of chromosome representation where the job sequence is 3, 1, 2, 7, 6, 5, 4. Then the job is scheduled to the most available machine. If there are two machines, job 3 is allocated at machine 1 and job 1 is allocated to machine 2. Job 2 is allocated to machine 1 or machine 2 according to the machine that is available first.

b. Fitness Evaluation and Multicriteria Objective
   The fitness evaluation is the core of the algorithm, consisting of the assign jobs decoder and the fitness scoring function. The objective is minimization; a lower score represents a better schedule. The fitness function aggregates several critical production metrics, which are the mismatch penalty, setup time minimization, and colour precedence violations. The fitness function of the GA using equation (7)

c. Selection and Reproduction Mechanisms
   To drive the population toward optimality, the algorithm employs a dual-strategy for reproduction, which includes elitism and tournament selection. For selection, the highest-performing individuals of the current generation are cloned directly into the next generation. This guarantees that subsequent crossover or mutation operations do not inadvertently destroy the best-found solutions. In this GA, the elitism using 10% of the total population. The selection of parents for breeding in the GA using roulette wheel method. The individual with the lowest (best) fitness score in this subset has a higher opportunity to be selected as a parent. This method provides a good balance between exploration and exploitation, preventing premature convergence.

d. Crossover

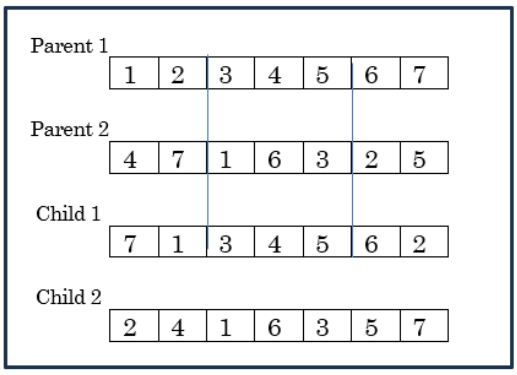

Figure 2. Crossover mechanism

The implementation uses an Ordered Crossover (X). It extracts a random sub-segment from Parent 1 and fills the remaining slots using the relative ordering of Parent 2. The crossover probability is 100% or all parents are crossed over. The example of ordered crossover using the GA is shown in Figure 2.

Figure 2 shows how the OX works. Assume there are parent 1 and parent 2, and the crossover points are points 2 and 5. Therefore, copy genes 3-5 from parent 1 to child 2, then fill gene 1 at child 1 from the first job sequence of parent 2 that has not been allocated to child 1 yet. Continue the procedure until all genes at child 1 are filled. The same scheme can be used to generate child 2.

e. Mutation
A Swap Mutation strategy is utilized. Subject to a mutation probability rate, two indices within the child chromosome are selected at random, and their respective job codes are swapped. This introduces minor random variations to help the algorithm escape local optima. The mutation rate in this paper is set 5%.

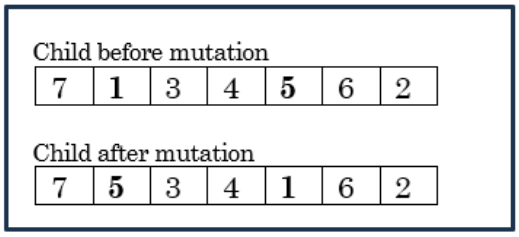

Figure 3. Mutation mechanism

Figure 3 describes one example of mutation where two points are chosen randomly. For example, the random numbers are 2 and 5, then swap the gene at position 2 with the gene at position 5.

f. Stopping Criteria
The evolutionary loop is governed by two stopping criteria: maximum generations and early stopping. For maximum generations, the algorithm naturally terminates upon reaching the predefined generations limit. In this paper, the maximum generation is 100. For early stopping**,** a manual interrupt mechanism is embedded within the loop. At the end of each generation, the algorithm checks for a stop flag. If triggered, the process breaks immediately and returns the best solution discovered up to that point. The stop flag has occurred when there is no improvement after several jobs plus five.

Results and Discussions

This chapter evaluates the proposed scheduling system using modified real data from a bicycle industry. The dataset is used to represent a realistic planning scenario while keeping the experiment reproducible, but it does not fully capture real shop-floor variability such as urgent arrivals, rework, or special exceptions. GA is run using Phyton 3.11. program language. The machine used an Intel i5 CPU with 12 GB RAM; runtime results are specific to this environment. The parameters that are used in this paper are shown in Table 1.

Table 1. GA Parameters

Population Size	100
Generations	100
Tournament Size	5
Elitism Rate	5%
Crossover Rate	100%
Mutation Rate	20%
Initial Setup Time	190 seconds
Same Color Setup Time	0 seconds
Different Color Setup Time	190 seconds

The genetic algorithm parameters (population size = 100, number of generations = 100, tournament size = 5, mutation rate = 0.20, and elitism rate = 5%) were selected based on values commonly reported in recent scheduling studies and preliminary computational experiments such as Sahoo [25]. The mutation rate is set high to hinder the method from easily trapping in a local optimum. Figure 4 shows the average fitness value for each generation using three attempts. The figure shows that for 100 generations, the solution converges.

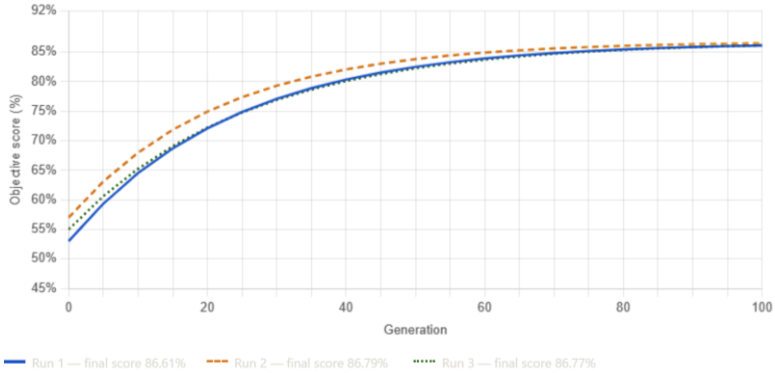

Figure 4. Average objective value for each generation

For setup times, the scheduler uses three parameters: initial setup, same-color setup, and different-color setup. This follows the common idea of sequence-dependent setup time, where transition cost depends on job order, which is especially relevant for paint operations, where color changes drive cleaning and changeover effort. The data that is used in this experiment will be shown in Table 2. Table 2 shows that one product can have more than one painting color and process. The last column shows the desired output that follows the desired input for the assembling process as the next process after painting.

The scheduling problem in this study is fundamentally a trade-off between two competing targets: (1) matching the desired completion order (mismatch minimization), and (2) reducing sequence-dependent changeovers (setup minimization). This trade-off is expected in real production scheduling because setup-dependent sequencing can conflict with due-date and priority-based sequencing.

In this case, we use a weight comparison of 90% mismatch priority and 10% setup. When assembly requirements are urgent, production control typically prioritizes due-date adherence or order completion sequence over local efficiency, because downstream delays propagate quickly through assembly and delivery commitments. In this case, the schedule is expected to produce low mismatch, potentially at the expense of additional color transitions, and therefore higher setup time.

Table 2. Scheduling data

Code	Product	Color 1	Color 2	Color 3	Color 4	Time	Desired output sequence
1	TITAN29ROCK (1) | Frame	Red	black	Blue	Green	5400	1
2	TITAN29ROCK (1) | Fender	Gray				3600	2
3	COLOSSUS27MAX (1) | Frame	green	orange	Black		6000	3
4	COLOSSUS27MAX (1) | Fork	Blue				4200	4
5	MEGAFLOW26CARGO (1) | Frame	Gray	yellow			4800	5
6	MEGAFLOW26CARGO (1) | Carrier	yellow				4500	6
7	ULTRA29RIDGE (1) | Frame	Blue	brown	Green	White	5200	7
8	ULTRA29RIDGE (1) | Fender	green				3800	8
9	JUMBO24HAUL (1) | Frame	Red	white	Black		4000	9
10	JUMBO24HAUL (1) | Carrier	Blue				4300	10
11	GOLIATH25LOAD (1) | Frame	Black	gray	Red	Blue	5500	11
12	GOLIATH25LOAD (1) | Fork	brown				3700	12
13	MAMMOTH26LONGRUN (1) | Frame	Blue	purple			5000	13
14	MAMMOTH26LONGRUN (1) | Fender	purple				3500	14
15	RHINO27POWER (1) | Frame	Black	yellow	Red	White	4500	15
16	RHINO27POWER (1) | Fork	yellow				2000	16

All optimization methods were evaluated under the same configuration case 1, using the same fixed dataset, the same capacity constraints, and the same auto-mode schedule construction logic. Each optimization method was given three attempts to produce a result. This standardization is important because metaheuristic

outcomes are sensitive to evaluation budget and stopping rules, so comparisons should be interpreted under the configured budget rather than as absolute superiority.

Table 3 reports the Score %, where higher is better, and runtime for GA, Ant Colony Optimization (ACO), Particle Swap Optimization, and Variable Neighborhood Search (VNS). The Score% is the objective that is obtained from equation 7. Metaheuristic procedures must be executed several times since they produce stochastic values. Using three attempts for the GA, with a 95% confidence level, the confidence interval is 85.6 < Score% < 87,2. The confidence interval value is narrow enough to make a good conclusion. Under the tested configuration, VNS achieves the highest Score% across attempts but also takes the longest runtime. VNS has 5% better Score% than GA; however, GA produces shorter runtime. ACO and Particle Swap produce lower scores under the tested setup, and one Particle Swap run does not return a score in the recorded results under a determined period. The ACO has 52.5% lower score than the GA, and the PSO has 36.9% lower score than the GA.

Table 3. Scheduling result

Attempt	Genetic algorithm		Ant colony optimization		Particle swap optimization		Variable Neighbourhood search	
	Score%	Time(s)	Score%	Time (s)	Score%	Time (s)	Score%	Time (s)
1st Attempt	86.61	190	37.49	129	53.60	217	90.65	1248
2nd Attempt	86.79	181	41.56	129	55.77	217	91.12	1316
3rd Attempt	86.77	177	44.35	129	-	219	91.93	1259
	86.72	182.67	41.13	129	54.69	217.67	91.23	1274,3

Each method represents a different optimization logic. ACO is known for constructive solution building guided by learned desirability and has strong historical performance for routing and sequencing, but it can be sensitive to parameter settings and instance structure. PSO is originally a continuous optimizer and typically needs adaptation to handle discrete permutations, often requiring careful encoding and repair to avoid infeasible or weak sequences. VNS systematically changes neighborhood structures to escape local minima and is often strong for combinatorial problems, but runtime can grow when many neighborhoods are explored, or feasibility checks are expensive.

The scheduling structure in this study combines discrete sequencing decisions, sequence-dependent setups, and multi-machine capacity constraints. Problems in this class are scaling poorly for exact methods as instance size grows, which is why metaheuristics are commonly used in applied industrial scheduling. GA is a standard choice because it can search for large discrete spaces, handle complex objective terms, and incorporate problem-specific operators for permutation problems.

In this system, GA also fits the "what-if" requirement. Because the representation and evaluation logic are consistent, the same objective calculation can be re-applied after manual schedule edits to recompute mismatch and setup under the same rules. This supports a decision-support workflow where the goal is not only an optimal result, but also a workable mechanism for exploring alternatives under constraints. The important parameters for this scheduling problem are setup times. There are three different setup times: setup times for difference color, setup times for the same color, and initial setup time. The sensitivity analysis for three parameters is shown in Figure 5. Figure 5 shows that the most sensitive setup time parameters are setup times for different colors, and the least sensitive is the initial setup time. It means that in practice, it is important for the company to keep setup time between different colors as low as possible to reduce setup time.

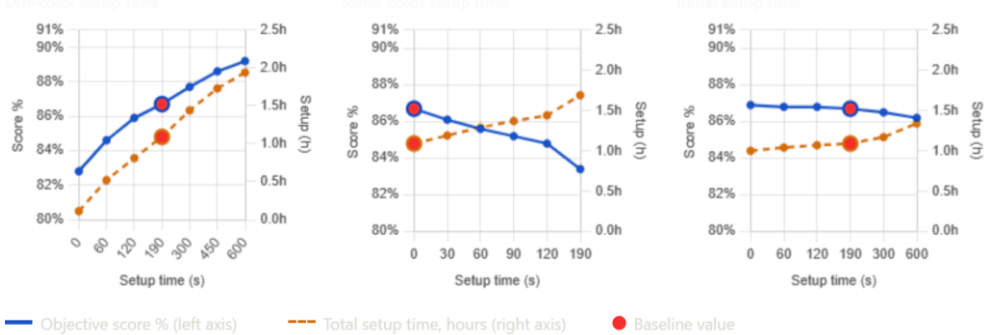

Figure 5. Sensitivity analysis

This study contributes to metaheuristic-based scheduling practice by showing how a GA can be structured around a domain-specific representation (jobs expanded into ordered color segments) and evaluated with penalty logic to protect feasibility. This is consistent with the established principle that metaheuristics are most effective in combinatorial scheduling when they are paired with problem-specific encoding and constraint handling rather than treated as a generic black box.

Practically, the system provides a structured way for planners to convert day-to-day operational priorities into a consistent scheduling rule. In real production control, priorities shift between urgency, efficiency, and a tunable weighting mechanism makes that trade-off explicit and repeatable, rather than relying on informal judgment alone. This can reduce ad-hoc planning behavior and improve consistency of decisions across different planners and shifts.

For the Painting department, the most direct operational benefit is improved control over setup losses under high color variety. When setup time depends on the job sequence, even simple reordering can drive meaningful differences in downtime and throughput; making setup visible in the objective and in the schedule, output supports more deliberate sequencing decisions. In settings where due dates are flexible, the tool supports schedules that reduce changeovers and improve capacity utilization without requiring the planner to manually group and check many alternative sequences.

For cross-functional coordination, the explicit "mismatch" metric creates a shared language between Painting and Assembly. Instead of debating schedule quality subjectively, both teams can discuss measurable consequences of prioritizing one goal over another, which is consistent with production planning practice where alignment improves when performance measures are transparent and jointly interpretable. This helps reduce downstream disruption risk when Assembly sequence matters and clarifies when deviations are acceptable because setup reduction is more valuable.

Operational reliability is also improved through validation and feasibility enforcement. In multi-machine scheduling, infeasible plans can quickly undermine trust in optimization tools. A system that blocks infeasible edits and maintains constraints supports adoption because it reduces planner effort spent on manual checking and rework, which is a common barrier in decision-support deployments. The "what-if" capability further supports real operations where last-minute changes are normal, enabling planners to adjust the plan while preserving feasibility logic.

Finally, the comparison across methods informs practical method selection. In production environments, planners often value a solution that is "good enough" and available quickly over one that is marginally better but slow or inconsistent. The observed pattern that some methods trade higher solution quality for higher runtime supports a practical recommendation: choose the optimization approach that matches the available planning time window and required responsiveness.

Conclusions

The research demonstrates a mathematical model and solution of parallel machine for a painting machine at a bicycle company with multiobjectives. The objectives are minimizing sequence differences between painting department and input of the assembling department, and the second objective is minimization setup machine. A GA algorithm is used since the model is a NP hard problem. Other metaheuristic methods are proposed as comparison of the GA algorithm. Metaheuristic approaches are highly effective for solving industrial-sized parallel machine scheduling problems where exact optimization fails to scale. The results show that VNS outperform the GA but requires much longer runtimes, whereas the Genetic Algorithm (GA) is better suited for "what-if" scenarios and rapid decision-making in a production environment. Incorporating sequence-dependent setups into the scheduling model allows for better control over setup losses and improves throughput by making the costs of colour changeovers visible to planners. The introduction of an explicit "mismatch" metric facilitates better communication between the painting and assembly departments, replacing subjective debate with transparent, measurable performance data. The developed system enhances operational reliability by enforcing feasibility constraints and allowing planners to adjust schedules manually while maintaining optimized logic. Ultimately, the choice of optimization method should align with the specific planning time window and responsiveness required by the manufacturing facility.

References

[1] A Agárdi, and K. Nehéz, "Parallel machine scheduling with Monte Carlo tree search," *Acta Polytechnica*, vol. 61, no. 2, pp. 307 – 312, April 2021, doi: https://doi.org/10.14311/ap.2021.61.0307

[2] M. Gallo, G. Mazzuto, F Ciarapica, and M. Bevilacqua, "Artificial intelligence to solve production scheduling problems in real industrial settings: Systematic literature review," *Electronics*, vol. 12, no. 23, pp. 4732 – 4732, 2023, doi: https://doi.org/10.3390/electronics12234732

[3] M. Moser, N. Musilu, A. Schaerf and F Winter, "Exact and metaheuristic approaches for unrelated parallel machine scheduling," *Journal of Scheduling*, vol. 25, no 5, pp. 507 – 534, 2021. doi: https://doi.org/10.1007/s10951-021-00714-6.

[4] F. F. Boctor, D. Zaatour, and J. Renaud, "Scheduling parallel extrusion lines," *Journal of Project Management*, vol. 9, no. 1, pp. 1 – 16, 2023, doi: https://doi.org/10.5267/j.jpm.2023.11.002

[5] F. Winter, and N. Musliu, "Constraint-based scheduling for paint shops in the automotive supply industry," *ACM Transactions on Intelligent Systems and Technology***,** vol. 12 no. 5, Article 58, 2021 https://doi.org/10.1145/3430710

[6] R Zhang, "Environment-aware production scheduling for paint shops in automobile manufacturing: a multi-objective optimization approach,". *International Journal of Environmental Research and Public Health*. Vol. 15, no. 1, p. 32, 2018 https://doi.org/10.3390/ijerph15010032

[7] Y. Zhang, Y. Shi, and J. Xu, "Research on PBS buffer scheduling strategy problems based on genetic algorithms," 2025 *International Conference on Education, Management and Information Technology* (EMIT 2025), July 2025, doi: https://doi.org/10.1051/itmconf/20257701022/pdf.

[8] S. Vasilis, N. Nikos, A. Kosmas, and M. Dimitris, "A toolbox of agents for scheduling the paint shop in bicycle industry," *Procedia CIRP*, Jan. 2022, doi: https://doi.org/10.1016/j.procir.2022.05.124.D.

[9] H. Zhang, and W Ding "A decomposition algorithm for dynamic car sequencing problems with buffers," *Applied Sciences*. vol 13, no. 12, p. 7336, 2023 https://doi.org/10.3390/app13127336.

[10] V. Siatras, E. Bakopoulos, P. Mavrothalassitis, N. Nikolakis, and K. Alexopoulos, "Production scheduling based on a multi-agent system and digital twin: A bicycle industry case," *Information*, June 2024, doi: https://doi.org/10.3390/info15060337.

[11] I. K. Singgih, O. Yu, B. Kim, J. Koo, and S. Lee, "Production scheduling problem in a factory of automobile component primer painting". *J. Intell. Manuf*, vol. 31, pp. 1483 – 1496, 2020, doi: https://doi.org/10.1007/s10845-019-01524-6

[12]. F. Winter, N. Musliu, E. Demirović, and C. Mrkvicka, "Solution approaches for an automotive paint shop scheduling problem" *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 29, pp. 573–581, 2019, doi: https://doi.org/10.1609/icaps.v29i1.3524

[13] S. Bysko, J. Krystek and S. Bysko, "Two approaches to car sequencing in the paint shop" *J. Phys.: Conf. Ser.* 1780 012028, 2021, doi: https://doi.org/10.1088/1742-6596/1780/1/012028

[14] F. Yu, Y. Peng, J. Li, G Zhou, L. Chen "An analysis of optimization for car pbs scheduling based on greedy strategy state transition algorithm", *Applied Sciences*, vol 13. no. 10, p. 6194, 2023, https://doi.org/10.3390/app13106194

[15] J. Yang, T. Sun, X. Huang, K. Peng, Z. Chen, G. Qian, Z. Qian "Optimizing painting sequence scheduling based on adaptive partheno-genetic algorithm" *Processes*. vol. 9, no. 10, p. 1714, 2021, https://doi.org/10.3390/pr9101714

[16] J. Adan, "A hybrid genetic algorithm for parallel machine scheduling with setup times: A comparative study of metaheuristics on large problem instances," *Journal of Intelligent Manufacturing*, no. 33, pp. 2059–2073, 2022, doi: https://doi.org/10.1007/s10845-022-01959-4 Springer.

[17] X. Wang, and Y. Zhu, "Hybrid genetic algorithm for flexible job shop scheduling with sequence-dependent setup times and job lag times," *IEEE Access*, pp. 99:1-1, 2021, doi: https://doi.org/10.1109/ACCESS.2021.3096007

[18] A. Berthier, A. Yalaoui, H. Chehade, F. Yalaoui, L. Amodeo and C Bouillot, "Unrelated parallel machines scheduling with dependent setup times, machine eligibility, and resource constraints*", Computers & Industrial Engineering*, 170, 108736, 2022, https://doi.org/10.1016/j.cie.2022.108736

[19] M. Kurdi, "Modifying the ant colony algorithm for the open shop scheduling problem with the objective of minimizing makespan" *Knowledge-Based Systems*, 242, 108323. 2022, doi: https://doi.org/10.1016/j.knosys.2022.108323.

[20] F. Pulansari and T. D. Retno "The unrelated parallel machine scheduling with a dependent time setup using ant colony optimization algorithm", *Jurnal Teknik Industri: Jurnal Keilmuan dan Aplikasi Teknik Industri*, vol. 23, no 1, pp. 65-74, 2021, doi: https://doi.org/10.9744/jti.23.1.65-74

[21] S. Yan, G. Zhang, J Sun, and W Zhang, W., "An improved ant colony optimization for solving the flexible job shop scheduling problem with multiple time constraints," *Mathematical Biosciences and Engineering*, vol. 20, no. 4, pp.7519–7547, 2023. doi: https://doi.org/10.3934/mbe.2023325.

[22] R. Wu, Z. Tian, X. Li, C. Wu, H. Tang, and Y. Li, "Improved discrete particle swarm optimization algorithm for solving fuzzy flexible job shop machines and automated guided vehicles fusion scheduling problem." *Engineering Applications of Artificial Intelligence*, 160 (Part B), 111951, 2025, https://doi.org/10.1016/j.engappai.2025.111951.

[23] K. Sun, D. Zheng, H. Song, Z. Cheng, X. Lang, W. Yuan, J. Wang, "Hybrid genetic algorithm with variable neighborhood search for flexible job shop scheduling problem in a machining system." *Expert Systems with Applications*, vol 215, 119359, 2023, doi: https://doi.org/10.1016/j.eswa.2022.119359

[24] X. Wan, and T Jiang, "A dominance relations-based variable neighborhood search for assembly job shop scheduling with parallel machines," *Processes*, 13 (5), 1578, 2025, doi: https://doi.org/10.3390/pr13051578

[25] A. Sahoo, R. K. Dalei, S. K. Rath, U. K. Sahu, and K. Tiwary. "Parameter optimisation of genetic algorithm utilising Taguchi design for gliding trajectory optimisation of missile", *Defense Science Journal*, vol. 74, no. 1, January 2024, pp. 127-142, 2024, doi: https://doi.org/10.14429/dsj.74.18496