


## Article

# Application of a Non-Dominated Sorting Genetic Algorithm to Solve a Bi-Objective Scheduling Problem Regarding Printed Circuit Boards

Yung-Chia Chang <sup>1</sup>, Kuei-Hu Chang <sup>2,\*</sup>  and Ching-Ping Zheng <sup>1</sup>

<sup>1</sup> Department of Industrial Engineering and Management, National Yang Ming Chiao Tung University, Hsinchu 300, Taiwan; jasmine.chang@nycu.edu.tw (Y.-C.C.); minnie8668@gmail.com (C.-P.Z.)

<sup>2</sup> Department of Management Sciences, R.O.C. Military Academy, Kaohsiung 830, Taiwan

\* Correspondence: evenken2002@yahoo.com.tw

**Abstract:** An unrelated parallel machine scheduling problem motivated by the scheduling of a printed circuit board assembly (PCBA) under surface mount technology (SMT) is discussed in this paper. This problem involved machine eligibility restrictions, sequence-dependent setup times, precedence constraints, unequal job release times, and constraints of shared resources with the objectives of minimizing the makespan and the total job tardiness. Since this scheduling problem is NP-hard, a mathematical model was first built to describe the problem, and a heuristic approach using a non-dominated sorting genetic algorithm (NSGA-II) was then designed to solve this bi-objective problem. Multiple near-optimal solutions were provided using the Pareto front solution and crowding distance concepts. To demonstrate the efficiency and effectiveness of the proposed approach, this study first tested the proposed approach by solving test problems on a smaller scale. It was found that the proposed approach could obtain optimal solutions for small test problems. A real set of work orders and production data was provided by a famous hardware manufacturer in Taiwan. The solutions suggested by the proposed approach were provided using Gantt charts to visually assist production planners to make decisions. It was found that the proposed approach could not only successfully improve the planning time but also provide several feasible schedules with equivalent performance for production planners to choose from.

**Keywords:** multi-objective; unrelated parallel machine scheduling problem; non-dominated sorting genetic algorithm II; machine eligibility restrictions; resource constraints

**MSC:** 68U35; 90B35; 68W50



**Citation:** Chang, Y.-C.; Chang, K.-H.; Zheng, C.-P. Application of a Non-Dominated Sorting Genetic Algorithm to Solve a Bi-Objective Scheduling Problem Regarding Printed Circuit Boards. *Mathematics* **2022**, *10*, 2305. <https://doi.org/10.3390/math10132305>

Academic Editor: Frank Werner

Received: 3 June 2022

Accepted: 28 June 2022

Published: 1 July 2022

**Publisher's Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



**Copyright:** © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

In recent years, newer generations of electronic products have been coming out quickly, and the manufacturing of products gradually tends toward a large variety of items with small quantities of each. As the product life cycles are shortened, enterprises must respond to the customer requirements more quickly. In compliance with complex and diversified production restrictions, production plants must import more advanced and cost-efficient production techniques [1]. Efficient scheduling can optimize the allocation and planning of the resources of production lines. This helps to increase the manufacturing efficiency and on-time delivery rate. It also enhances an enterprise's customer satisfaction and competitive advantage in the industry. Therefore, in a scheduling system, multiple performance indicators should be used as the criteria for scheduling with different considerations and requirements, which presents a multi-objective scheduling problem.

The objectives in a multi-objective problem may be positively correlated, uncorrelated, or negatively correlated. A multi-objective scheduling problem is more complicated than a single-objective scheduling problem. The approaches used for multi-objective problem

solving in the existing literature can be divided into three executive modes [2]. The first one is the prior approach. The decision maker has to label the importance of each objective before solving it. The prior approach is divided into weighted and hierarchical computing modes. In the weighted mode, the decision maker gives different weights based on the importance of each objective with a weighted sum of 1. Multiple objectives are merged into a single-objective problem. In the hierarchical model, the objectives are sorted based on their importance. Then, hierarchical solving is performed, and the optimal solution obtained in the previous level is used in the next level as a constraint. The second approach to multi-objective problem solving is the posterior approach. It aims to obtain a set of non-dominated solutions (also known as Pareto solutions). This set of solutions is better than the other feasible solutions for each objective. The decision maker selects an eligible optimal solution from this set of solutions. The third one is the interactive approach. The decision maker expresses their preferred target value for each objective in the solution process. The overall computing mode extracts a feasible trade-off solution based on the objective equation and the decision-maker's preference. In the actual scheduling work, the production environment is likely to vary with material availability, machine status, manpower, quality, and customer requirements. It is unlikely to measure the relative importance to allow for giving weights according to the objectives. Therefore, many scholars combined the posterior approach with a meta-heuristic to solve the multi-objective scheduling problem, e.g., multi-objective ant colony optimization [3], modified multi-objective teaching–learning-based optimization-refined learning scheme [4], and multi-objective particle swarm optimization [5]. The essence is that the optimal solution is not a single value; instead, it is a solution set. Each solution in the solution set is a feasible optimal solution, and the only difference is which objective equation each optimal solution prefers.

The unrelated parallel machine scheduling problem is a type of parallel machine scheduling problem. It is often seen in manufacturing industries, such as printed circuit board assembly [6], wafer fabrication [7], the textile industry [8], the transport industry [9], semiconductor wafer manufacturing [10], and supply chain scheduling problems [11]. Multiple machines can work simultaneously and independently in a parallel machine production environment. Each job is processed on only one machine once. In terms of unrelated parallel machines, the processing time of each job on different machines is different, and there is no specific scale value [12]. As each machine may have several jobs to be processed, the unrelated parallel machine scheduling problem should determine the machine tool for each job and the processing sequence of jobs on each machine.

Garey and Johnson [13] indicated that unrelated parallel machine scheduling problems are NP-hard problems. When the problem scale is expanded, it is hard to obtain the optimal solution within a reasonable time using mathematical programming. In the face of this problem, most scholars often use a meta-heuristic to obtain the approximate solution; the tabu search method [14–16], genetic algorithm [17], particle swarm optimization [18], variable neighborhood search [19], and ant colony optimization [20] are some examples.

This study took a famous computer hardware manufacturer in Taiwan (hereinafter referred to as Company G) as the research subject. The company specializes in research and development, manufacturing, and sales of electronic products, such as mainboards, graphic cards, and PCs. PCB assembly is required in the manufacturing processes of these electronic products. The process comprises surface mount technology (SMT), a dual inline package, testing, and packaging. The electronic parts are soldered on the surface of a circuit board with solder paste in the SMT process. The SMT process of Company G comprises multiple production lines. These production lines have different configurations and manufacturing capacities. If each production line is regarded as a machine, the processing time for each product on each machine is different. This manufacturing environment can be regarded as unrelated parallel machines. In addition, the products have different complexities. Some products can be processed only on some machines, meaning that there are machine eligibility restrictions. Before processing each product, the materials and parts for the

product should be put in the feeder one by one, and the settings should be confirmed. The required setup time varies with the differences in the processed products. There are constraints due to the sequence-dependent setup times. Some advanced products have parts mounted on both sides of the PCB. They should be split into the front panel and backplane during processing, and the backplane must be processed before the front panel is processed. These are called precedence constraints. Additionally, before processing each product, it is necessary to make sure that all required materials are ready. Therefore, each job has unequal job release times. Besides these processing constraints, a stencil applicable to the product should be used as a fixture in the SMT process to print the solder paste on the circuit board. The apertures and positions in the stencils are different. The total number of stencils available for a particular model is limited. Thus, this production environment has extra shared resource constraints. These constraints increase the difficulty of arranging the production schedule to best utilize the production capacity while ensuring on-time delivery in practice. Without proper tools to assist the production scheduling, it usually took Company G several hours of cross-checking for one feasible production schedule. When the situation changed, such as late deliveries from suppliers, a change in product due dates, machine breakdown, and the like, it required several more hours for Company G to rearrange the production schedule in response to those changes.

The scheduling problem in the SMT process of Company G was regarded as an unrelated parallel machine scheduling problem in this study. This problem had machine eligibility restrictions, sequence-dependent setup times, precedence constraints, unequal job release times, and extra shared resource constraints. Many different performance indicators are used to measure the quality of production schedules. Among them, the makespan and total tardiness were chosen in this study as the two objectives to be minimized for the production schedules. Optimizing the former was chosen to ensure the balanced use of machines, while optimizing the latter was undertaken to make sure that if there were jobs that could not be completed before their due dates, they would not be delayed for too long. As this problem, even with just one objective, is NP-hard, the time for solving this problem was expected to exponentially increase with the problem scale. Therefore, this study used the heuristic algorithm based on the scheduling logic of Company G, i.e., the current scheduling method, to generate the initial solution meeting the scheduling constraints. It also used the non-dominated sorting genetic algorithm II (NSGA-II) to optimize the scheduling objective through the test data and the actual work orders data to be scheduled by Company G. This was chosen to evaluate the feasibility and effectiveness of this algorithm. NSGA-II uses the posterior approach to deal with multi-objective optimization. It is an effective approach that generates a set of Pareto solutions, instead of one solution, for decision makers to ponder and thus is more acceptable in practice.

This paper is organized as follows. Section 2 reviews the related literature and Section 3 presents the studied problem along with the research methods. The results and analysis of the numerical experiments are organized in Section 4. Section 5 concludes this paper with a research summary and points out future directions.

## 2. Preliminaries

### 2.1. Multi-Objective Optimization Scheduling Problem

In an actual scheduling problem, multiple performance indicators should be considered together, wherefrom the scheduling problem of multi-objective optimization is derived. The objectives of a multi-objective problem conflict with each other. In this type of problem, the objectives should be accepted or rejected. The obtained optimal solution is a feasible set. Decision makers can assign weights based on the importance of each objective and determine a satisfactory optimal solution from the solution set. A multi-objective optimization problem can be represented using the following mathematical model [21]:

$$\min / y = f(x) = (f_1(x), f_2(x), \dots, f_n(x)) \quad (1)$$

$$\text{subject to } x = (x_1, x_2, \dots, x_m) \in X \quad (2)$$

$$y = (y_1, y_2, \dots, y_n) \in Y \quad (3)$$

where  $x$  is the decision vector,  $X$  is the parameter space,  $y$  is the objective vector, and  $Y$  represents the objective space. Multi-objective optimization problems aim at finding all non-dominated solutions. Without loss of generality, considering a maximization problem with  $n$  objectives and two decision vectors  $a$  and  $b$  ( $a, b \in X$ ),  $a$  is said to dominate  $b$  only if all the objective values in  $a$  are at least as high as the respective values in  $b$  and there exists at least one objective value in  $a$  that is greater than that in  $b$ , as represented in Equation (4):

$$(\forall i \in \{1, 2, \dots, n\} : f_i(a) \geq f_i(b)) \wedge (\exists j \in \{1, 2, \dots, n\} : f_j(a) > f_j(b)) \quad (4)$$

Furthermore, if no other decision vectors of the parameter space could dominate  $a$ , then  $a$  is called a non-dominated solution.

Shahvari and Logendran [22] discussed the unrelated parallel machine scheduling problem of batch machines. They considered the sequence-dependent and machine-dependent setup times. The available machine time and unequal job release times were dynamic. The objective was to minimize the total weighted makespan and total weighted tardiness. They used the linear combination to reduce the bi-objective problem to a single-objective problem. Bozorgirad and Logendran [23] discussed the hybrid flow line scheduling problem with different unequal job release times, different unequal job release times of machines, and sequence-dependent setup times. The objective was to minimize the weighted makespan and weighted tardiness. The bi-objective problem was simplified via weighting. Their study used the simulated annealing, tabu search, genetic algorithm (GA), and two-stage GA methods to solve the problem and the results were compared. The results showed that the GA produced a better solution than the other three algorithms. Zheng and Wang [24] discussed the green energy manufacturing industry's unrelated parallel machine scheduling problem and considered the resource constraints. The objectives included minimizing the makespan and total carbon emissions. A collaborative multi-objective fruit fly optimization algorithm was used to solve this problem. Critical-path-based carbon saving was used to increase the modes of job processing time on a non-critical path. Carbon emissions could be reduced without increasing the makespan to obtain the Pareto solution with two better objective functions.

Afzalirad and Rezaeian [25] proposed an unrelated parallel machine scheduling problem and considered unequal job release times, sequence-dependent setup times, and machine eligibility restrictions. The objective was to minimize the weighted average processing time and weighted average tardiness. They used NSGA-II and multi-objective ant colony optimization to solve this problem. Gao et al. [26] mentioned that the scheduling problem should avoid late delivery. The problem can lead to untimely manufacturing of products, increasing the inventory cost. Therefore, for an elastic job-shop scheduling problem with available machine restrictions, they took the makespan and total delay or lead time as solution objectives. They used the harmony search algorithm to solve this problem, which was provided with the concept of domination. After every iteration, if the new solution could dominate one or more non-dominated solutions, it replaced the previous best solution. Additionally, Gao et al. also used local searching for different objectives to optimize the current solution and increase the solution efficiency. Aiming at the total tardiness and total setup cost of the machine, Zhang et al. [27] combined multi-objective particle swarm optimization with local searching to solve a batch processing parallel machine scheduling problem with sequence-dependent setup times. In addition to clustering the obtained feasible solutions based on domination, the distribution pattern of feasible solutions in the solution space was considered. The solutions that could not dominate others were sorted using crowded distance to ensure that feasible solutions could be extensively distributed in the solution space.

## 2.2. Unrelated Parallel Machine Scheduling Problem

The unrelated parallel machine scheduling problem involves a set of jobs to be processed on one machine in a simple job environment. Each job is processed only once, and each machine only processes one job at a time. This problem aims to obtain the scheduling result that optimizes the objective while satisfying production constraints. The computing modes for this problem include mathematical programming and heuristic algorithms while considering the restrictions of a real production environment. In recent literature/years, most scholars used heuristic and meta-heuristic algorithms for large-scale scheduling problems.

McNaughton [28] studied the unrelated parallel machine scheduling problem in the first place. As the production environment became complex, many scholars discussed the unrelated parallel machine scheduling problem with sequence-dependent and machine-dependent setup times, aiming at minimizing the makespan according to different scheduling restrictions. Yilmaz Eroglu et al. [29] used a GA and the proposed local search to solve this problem. They compared it with the ant colony optimization, and the result showed that the GA generated/provided a better solution. Ezugwu and Akutsah [30] used the firefly algorithm to obtain the initial solution and used the local search to change the working order, which optimized the initial solution to obtain a robust near-optimal solution. Ewees et al. [31] used the salp swarm algorithm to solve the problem. They used the firefly algorithm to optimize the obtained preliminary solution probabilistically. The result showed that the hybrid algorithm could enhance the algorithm's solving ability and shorten the computing time.

Mir and Rezaeian [32] discussed the existing restrictions of sequence-dependent and machine-dependent setup times and unequal job release times. They considered the job deterioration problem, where processing a delayed job would take more processing time. The scheduling problem with the learning effect allows for the possibility that the processing time can be shortened with operator experience. They set the objective as minimizing the total machine-loading capacity. They combined the particle swarm optimization with a GA to solve the problem. Two particles were crossed over in every iteration and the optimum solution was mutated. The study of Mir and Rezaeian mentioned that the diversity of solutions could be increased by using the logic of crossover and mutation to work out a better solution. Chen et al. [7] also discussed a similar problem but with the job expired time, which can often be found in semiconductor fabrication. They applied a hybrid tabu search to solve this problem. Yepes-Borrero et al. [33] considered the machine-dependent and sequence-dependent setup times and the machine-dependent and sequence-dependent extra resource constraints, and set the objective as minimizing the makespan. They proposed the greedy randomized adaptive search procedure. The solution procedure was divided into two stages. In stage 1, the machine assignment and job sorting were performed. The solution obtained at stage 1 was checked in stage 2 for the extra resource constraints and repair was conducted. A local search was performed for the repaired solution to optimize the current solution. Zheng and Wang [34] discussed the scheduling problem with extra resource constraints and aimed at minimizing the makespan. They used the two-stage adaptive fruit fly optimization algorithm to solve the problem. The proposed heuristic algorithm was used to generate the initial solution in stage 1. The solution was used as the initial center position of the colony in stage 2. A food source was randomly looked for near the center position using the sense of smell. Odorousness was calculated, then the possible location with maximum odorousness was sought using vision, and the center position of the fruit fly was renewed.

Bektur and Saraç [35] considered the sequence-dependent setup times and machine eligibility restrictions. Compared with general setup time restrictions, there was only one set of common servers for the setup operation before processing. There would be idle time if there were more than two machines to set up simultaneously. This study used the dispatching rules of apparent tardiness cost with setups (ATCS) to generate the initial solution. The tabu algorithm was used and the annealing algorithm was simulated



to optimize the initial solution to minimize the total weighted tardiness. The findings showed that when compared with the randomly generated initial solution, the initial solution generated using the dispatching rules of ATCS could obtain a better solution faster. Al-Harkan and Qamhan [36] discussed the unrelated parallel machine scheduling problem with sequence-dependent setup times, unequal job release times, extra resource constraints, and the minimized makespan as the objective. They used a two-stage hybrid variable neighborhood search with simulated annealing (TSVNS\_SA) for solving. The proposed heuristic algorithm obtained an initial solution in stage 1. This initial solution was optimized using TSVNS\_SA in stage 2. Fanjul-Peyro [37] also studied an unrelated parallel machine scheduling problem with sequence-dependent setups and shared resources. A mixed-integer programming model was formed to minimize the makespan. To solve the problem, a three-phase exact algorithm was designed in which the machine assignment and the job priority were determined in the first two stages and the jobs were assigned while considering the setups of the shared resources.

### 2.3. Non-Dominated Sorting Genetic Algorithm II (NSGA-II)

Non-dominated sorting genetic algorithm II (NSGA-II) was proposed by Deb et al. [38]. It was used to solve many problems, including scheduling [39,40], redundancy allocation [41,42], reducing carbon emissions [43], battery thermal management system management [44], passive heat supply tower management [45], and vehicle routing [46,47]. NSGA-II originated from the GA based on the natural evolution process, where the fittest individuals are selected for reproduction to produce offspring of the next generation via selection, crossover, and mutation.

In the multi-objective optimization problems, the decision vectors that are nondominated within the entire search space are denoted as Pareto optimal and constitute the so-called Pareto-optimal front. All other decision vectors are called dominated solutions. On the Pareto-optimal front, multiple solutions exist that could not dominate each other. NSGA-II defined the crowding distance of solution  $i$ , where solution  $i$  is on the Pareto-optimal front, denoted as  $CD_i$ , which measures the distance of solution  $i$  to its neighbors.  $CD_i$  is calculated using Equation (5) [38]:

$$CD_i = \sum_{m=1}^M \frac{|f_m^{i+1} - f_m^{i-1}|}{f_m^{max} - f_m^{min}} \quad i = 2, 3, \dots, l-1 \quad (5)$$

In Equation (5),  $M$  is the number of objectives;  $f_m^i$  is the fitness value of the  $m$ th objective of solution  $i$ ;  $f_m^{max}$  and  $f_m^{min}$  represents the maximum and minimum values of the fitness of the  $m$ th objective of the solution, respectively; and  $l$  represents the number of solutions on the Pareto-optimal front. In each of the  $M$  objectives, the crowding distances of the solutions with the largest and the smallest values of the fitness are set as infinity. The larger value of the crowding distance of a solution means that it is more distant from other solutions and hence more representative. Therefore, the solutions on the Pareto-optimal front could be ranked using this approach to identify the most representative feasible solution in each solution space. In each iteration of the evolving process, selecting the solutions with larger crowding distances could help to improve the diversity of the population. NSGA-II defined a crowded-comparison operator, denoted as  $\prec_n$ , as described in Equation (6):

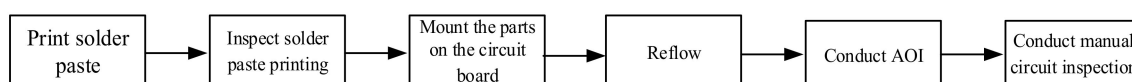
$$(a_{rank} < b_{rank}) \text{ or } ((a_{rank} = b_{rank}) \text{ and } (CD_a > CD_b)) \quad (6)$$

In Equation (6),  $a_{rank}$  and  $b_{rank}$  represent the nondomination ranks of solutions  $a$  and  $b$ , respectively;  $CD_a$  and  $CD_b$  represent the crowding distance of these two solutions, respectively.  $a \prec_n b$  means solution  $a$  is preferred over solution  $b$  under crowded comparison. This means that the solution with a lower rank is preferable between the two, and the one located in a lesser crowded region is preferable should both solutions belong to the same front.

### 3. Research Method and Procedure

#### 3.1. Problem Description

This study aimed to investigate the bi-objective unrelated parallel machine scheduling problem with machine eligibility restrictions, sequence-dependent setup times, precedence constraints, unequal job release times, and extra shared resource constraints. It took the SMT process of Company G as the subject of implementation, where the SMT process is shown in Figure 1. The NSGA-II was used to build the scheduling optimization system with a minimum makespan and tardiness. First of all, the PCB to be assembled was fed into the solder paste screen printer and printed with solder paste. When the solder paste printing inspection machine confirmed the correct solder paste printing, the parts were mounted on the circuit board by the mount. Then, the solder paste was fused using reflow. The quality was controlled using automated optical inspection (AOI) and manual circuit inspection.



**Figure 1.** Schematic diagram of the SMT process.

The SMT process of Company G had nine production lines. Regarding the present work order scheduling method, the field production management personnel assigned work orders to specified production lines based on the production environment restrictions and their experience. The work orders were divided into planned production and build-to-order production according to acceptance or rejection. The main difference between them was that the work order of build-to-order production was determined according to the due date specified by the customer, while the delivery time of the work order of planned production was determined according to the production plan. Therefore, the scheduling was relatively complex. The field production management personnel spent much of their working time on scheduling. According to the production environment setup of Company G, the bi-objective unrelated parallel machine scheduling problem investigated in this study had the following restrictions:

- (1) The work order processing time is different in various production lines.
- (2) The feasible machine tool for each work order is different.
- (3) The work order has a sequence-dependent setup time.
- (4) Work order precedence constraints.
- (5) Different unequal job release times of work orders.
- (6) The work order processing requires an extra shared resource.

Pinedo [12] proposed three parameters for describing the scheduling problem:  $\alpha | \beta | \gamma$ , where  $\alpha$  represents the machine environment setup,  $\beta$  represents the scheduling restriction, and  $\gamma$  represents the objective of the scheduling problem. To solve this scheduling problem, this study took the actual production environment configuration for the SMT process of Company G as the subject of implementation. The following assumptions were made for this problem:

- (1) Each machine only processes one work order at a time.
- (2) When a work order is assigned to a machine and processed, the processing may not be interrupted.
- (3) The workable machine for each work order and the processing time are confirmed and known.
- (4) Unequal job release times and work order delivery times are confirmed and known.
- (5) The setup time of different product types is confirmed and known.
- (6) The reworking characteristic of work orders is disregarded.
- (7) Machine fault and maintenance are disregarded.
- (8) The extra shared resources are renewable sources.

For the above scheduling restrictions and problem assumptions, this study used NSGA-II to build a scheduling optimization system to minimize the makespan and total tardiness. It was expected to assist the field production management personnel in scheduling and increase the SMT process's manufacturing efficiency.

### 3.2. Building a Mathematical Model

This study constructed a mixed-integer programming (MIP) model to describe this production scheduling issue.

#### Subscripts

$i$ : machine number,  $i \in M$ ;  
 $h, j, k$ : work order number,  $h, j, k \in N$ ;  
 $q$ : resource number,  $q \in V$ ;  
 $t$ : time,  $t \in H$ ;  
 $o$ : virtual work order.

#### Sets

$M$ : machine set,  $i = \{1, 2, \dots, |M|\}$ ;  
 $N$ : work order set,  $h, j, k = \{1, 2, \dots, |N|\}$ ;  
 $V$ : resource set,  $q = \{1, 2, \dots, |V|\}$ ;  
 $H$ : time set,  $t = \{1, 2, \dots, |H|\}$ .

#### Parameters

$p_{ij}$ : processing time of work order  $j$  on machine  $i$ ;  
 $s_{jk}$ : setup time between work order  $j$  and work order  $k$  of subsequent processing;  
 $r_j$ : start time of work order  $j$ ;  
 $d_j$ : delivery time of work order  $j$ ;  
 $E_{ij} = \begin{cases} 1, & \text{If work order } j \text{ can be processed by machine } i \\ 0, & \text{otherwise} \end{cases}$ ;  
 $PR_{hj} = \begin{cases} 1, & \text{If work order } h \text{ and work order } j \text{ have priority restrictions and} \\ & \text{work order } h \text{ processing is completed, then work order } j \text{ can be processed} \\ 0, & \text{otherwise} \end{cases}$ ;  
 $AR_q$ : available quantity of resource  $q$ ;  
 $res_{jq}$ : quantity of resource  $q$  required to process work order  $j$ ;  
 $G$ : maximum value.

#### Decision variables

$T_j$ : delay time of work order  $j$ ;  
 $x_{ij} = \begin{cases} 1, & \text{If work order } j \text{ is scheduled to be processed on machine } i \\ 0, & \text{otherwise} \end{cases}, i \in M; j \in N$ ;  
 $x_{ijk} = \begin{cases} 1, & \text{If work order } j \text{ is scheduled to be processed on machine } i, \\ & \text{and the processing of work order } k \text{ is continued} \\ 0, & \text{otherwise} \end{cases}, i \in M; j, k \in N$ ;  
 $x_{i0j} = \begin{cases} 1, & \text{If work order } j \text{ is scheduled to be processed on machine } i, \\ & \text{and it is the first work order} \\ 0, & \text{otherwise} \end{cases}, i \in M; j \in N$ ;  
 $z_{jt} = \begin{cases} 1, & \text{If work order } j \text{ is scheduled to be processed at time } t \\ 0, & \text{otherwise} \end{cases}, j \in M; t \in H$ .

#### Objective equations

$$\text{Min } C_{\max} \quad (7)$$

$$\text{Min } \text{Tardiness} = \sum_{j=1}^n T_j \quad (8)$$

#### Constraints

$$\sum_{i \in M} x_{ij} = 1, \quad \forall j \in N \quad (9)$$

$$\sum_{i \in M} \sum_{j \in \{0\} \cup \{N\}} x_{ijk} = 1, \quad \forall k \in N \quad (10)$$

$$\sum_{j \in N, j \neq k} (x_{ikj} + x_{i0j}) = x_{ij}, \quad \forall i \in M; \forall k \in N \quad (11)$$



$$\sum_{t=p_{ik}}^{|H|} (t * z_{kt}) \geq \sum_{t=p_{ij}}^{|H|} (t * z_{jt}) + \sum_{i \in M} \left( \sum_{k \in N, k \neq j} s_{jk} * x_{ijk} \right) + \sum_{i \in M} (p_{ik} * x_{ik}) + G \left( \sum_{i \in M} x_{ijk} - 1 \right), \quad \forall j \in N; k \neq j \quad (12)$$

$$\sum_{t=p_{ik}}^{|H|} (t * z_{kt}) \geq r_k + \sum_{i \in M} \left( \sum_{k \in N, k \neq j} s_{jk} * x_{ijk} \right) + \sum_{i \in M} (p_{ik} * x_{ik}), \quad \forall j, k \in N, k \neq j \quad (13)$$

$$\sum_{t \in H} z_{jt} = 1, \quad \forall j \in N \quad (14)$$

$$\sum_{t \in H} (t * z_{jt} * PR_{h,j}) - \sum_{t \in H} (t * z_{ht} * PR_{j,h}) \geq PR_{h,j} * \left( \sum_{i \in M} \sum_{j \in N, j \neq k} s_{kj} * x_{ikj} + \sum_{i \in M} (p_{ij} * x_{ij}) \right) \quad \forall h, j \in N, h \neq j \quad (15)$$

$$\sum_{j \in N} \sum_{i \in M} \sum_{s=t}^{t+p_{ij}-1} (res_{jq} * z_{js} * x_{ij}) \leq AR_q \quad \forall q \in V; \forall t \in H \quad (16)$$

$$x_{ijk} \leq E_{ik}, \quad \forall i \in M; \forall j, k \in N, k \neq j \quad (17)$$

$$x_{i0j} \leq E_{ij}, \quad \forall i \in M; \forall j \in N \quad (18)$$

$$x_{ij} \leq E_{ij}, \quad \forall i \in M; \forall j \in N \quad (19)$$

$$T_j \geq \sum_{t \in H} (t * z_{jt}) - d_j, \quad \forall j \in N \quad (20)$$

$$C_{max} \geq t * z_{jt}, \quad \forall j \in N; \forall t \in H \quad (21)$$

$$T_j \geq 0, C_{max} \geq 0, x_{ij}, x_{ijk}, z_{jt} \in \{0, 1\} \quad \forall i \in M; \forall j, k \in N \quad (22)$$

Equations (7) and (8) are objective equations that minimize the makespan and minimize the total tardiness of the work order, respectively. Equation (9) makes sure that each work order will be assigned to a machine and processed. Equation (10) makes sure that each work order will be assigned to at most one machine and processed, and each work order will be followed by only one work order. Equation (11) specifies that the work orders will be continuously processed on the machine. If work order  $k$  is assigned to machine  $i$ , then work order  $j$  will certainly be processed earlier than work order  $k$  on machine  $i$ . Otherwise, it is the first work order processed on machine  $i$ . Equation (12) ensures that the makespan of the next processed work order  $k$  is not smaller than the makespan of work order  $j$  plus the setup time between work orders  $j$  and  $k$  plus the processing time thereof. Equation (13) represents the unequal job release times of the work orders, making sure that work order  $k$  is not processed until the unequal job release times are added by the setup time for the previous work order  $j$  and its processing time. Equation (14) makes sure that each work order will be processed in a one-time interval only, meaning each work order will be processed only once. Equation (15) represents the precedence constraints, which makes sure that of the two work orders with precedence constraints, work order  $j$  will not be processed until work order  $h$  is processed. Equation (16) is the extra shared resource constraint, making sure that the quantity of each resource to be used in the same time interval will not exceed the available quantity of the resource. Equations (17)–(19) are the machine eligibility restrictions, ensuring that each work order will be processed by its eligible machine. Equation (20) calculates the work order tardiness. Equation (21) calculates the makespan of a work order. Equation (22) specifies the constant positivity of the tardiness and makespan, and the decision variable is binary.

According to the environment setup and scheduling restriction of the SMT process of Company G described using the MIP mathematical model and the fact that the unrelated parallel machine scheduling problem was demonstrated to be an NP-hard problem, solving a large-scale problem will take a long time. As a result, this study only used a mathematical model to solve the test data on a smaller problem scale. To optimize the scheduling system of Company G, this study used NSGA-II to solve the actual work orders to be scheduled for Company G on a larger problem scale.

### 3.3. Scheduling Optimization Algorithm Implementation Procedure

The scheduling result meeting the production environment restriction was obtained through the following procedure. The objectives were set as the makespan and total tardiness. The scheduling result was optimized to obtain the optimal solution where the two performance indicators were minimized. The solving process of this problem is described below.

#### (1) Encoding mode

This study used the chromosome encoding mode, where one chromosome represents a set of feasible schedules. Each gene in the chromosome represented one work order. The gene coding used integer coding in which each integer corresponds to a work order number. As the unrelated parallel machine scheduling problem included machine assignment and job sorting, the chromosome was expressed in the form of a 2D matrix. If there are  $n$  work orders and  $m$  machines to be scheduled, there will be  $n$  genes and a matrix with  $m$  columns. Each column included the work order corresponding to the machine. The genes from left to right represented the processing sequence of the work orders for the machine.

#### (2) Initial solution generation

The population size should be determined before the algorithm's execution. This means that the total number of chromosomes in the population is defined. The population size varies with the complexity of the problem. A good initial population can increase the crossover probability of superior offspring and shorten the computing time of an algorithm. An initial diversified population can enhance the global search capability of an algorithm [48]. An appropriate population size should simultaneously maintain the diversity of solutions and reasonable computing time.

To cause the population to have diversity and excellent chromosomes, this study used the feasible solution generated by a heuristic algorithm as one chromosome in the initial population of NSGA-II. This algorithm was set up and executed according to the work order scheduling logic and production line condition of Company G. It was defined as the current scheduling method of Company G. In terms of the current scheduling logic, the work order processing sequence was determined by the senior production management personnel of the subject company. The logic used by the production management personnel for work order scheduling was that the priority of the work orders in build-to-order production was higher than that of the work orders of planned production. The work orders were sorted in descending order according to the ratio of the quantity delivered to the quantity processed. If the ratio was the same, processed quantities were sorted in descending order. The work orders were sorted in ascending order based on the available processing time; the shorter the available processing time was, the shorter the distance from the unequal job release time point of the work order to the work order delivery time. Therefore, shorter jobs should be scheduled first to avoid delays in work orders. The unequal job release time points of work orders were sorted in order of precedence. For an earlier unequal job release time point, the earlier the work order should be scheduled to avoid the idle state for a long time.

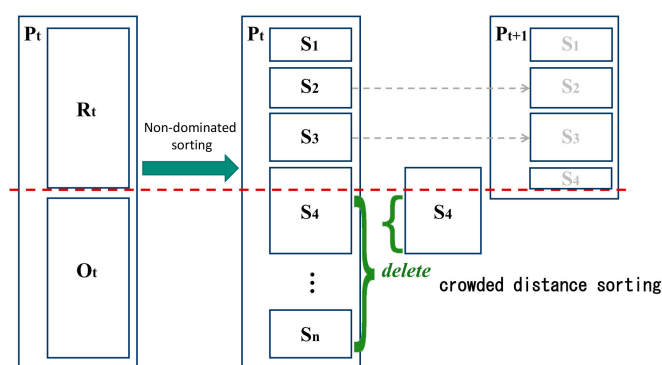
#### (3) Fitness value calculation

To measure the performance and fitness value of each chromosome, a fitness function should be designed according to the objective equation of the problem to be solved. In an iterative process, the fitness value is the criterion for the elimination of chromosomes. A chromosome with a better fitness value has a higher probability of being maintained. This study used the makespan and total tardiness of a work order as performance indicators and set up the fitness function to execute the non-dominated sorting in the next section, as expressed by Equations (7) and (8).

#### (4) Non-dominated sorting

Except for the initial generation, after each iteration, each chromosome worked out the two fitness values of the makespan and total tardiness according to the fitness function. The

non-dominated sorting of chromosomes was executed based on the Pareto front solution and crowded distance. To renew the population and leave a better solution, the chromosome with a better fitness value had a higher non-dominated rank. The solution of the highest dominated rank was taken as the chromosome of the next generation's population when renewing the population till the chromosome number reached the population size. The concept is shown in Figure 2. First, the fitness values corresponding to the chromosome left from the last generation and the new chromosome  $P_t$  generated after the crossover and mutation were worked out based on the fitness function. Then, the chromosomes were split via domination into the solution sets  $S_1, S_2, \dots, S_n$ . The non-dominated ranks of these solution sets were sorted and brought into the next generation. If the chromosome number of the solution set was larger than the preset population size, the chromosomes were sorted by using the crowded distance. The least representative chromosomes exceeding the population size were removed to generate the next generation's parent population  $P_{t+1}$ .



**Figure 2.** Schematic diagram of non-dominated sorting.

#### (5) Iteration termination condition

Considering the computing time, this study set the maximum number of iterations as the termination condition of an algorithm. If the number of iterations reached the upper bound, the algorithm stopped iterating, and the non-dominated solution sorted by the chromosome of the generation would be the final solution to this problem.

#### (6) Parental chromosome selection

To select the chromosomes for mating and mutating, several chromosomes needed to be selected from the parent population and put in the mating pool as parental chromosomes. To increase the mating probability of chromosomes with better fitness values and maintain the population diversity, this study combined the elite selection method with a roulette wheel selection as the criteria for bringing parental chromosomes into the mating pool. The feasible solution with the best fitness value was unconditionally brought into the mating pool, and the rest of the chromosomes were selected using the roulette wheel selection.

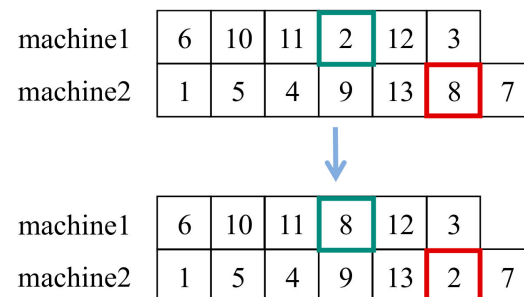
#### (7) Crossover

Two chromosomes were randomly selected from the mating pool for gene exchange to generate the offspring, which was expected to obtain chromosomes with better fitness values. This study combined the one-point crossover with the local search with a pilot calculation function as the crossover method of this algorithm.

#### (8) Mutation

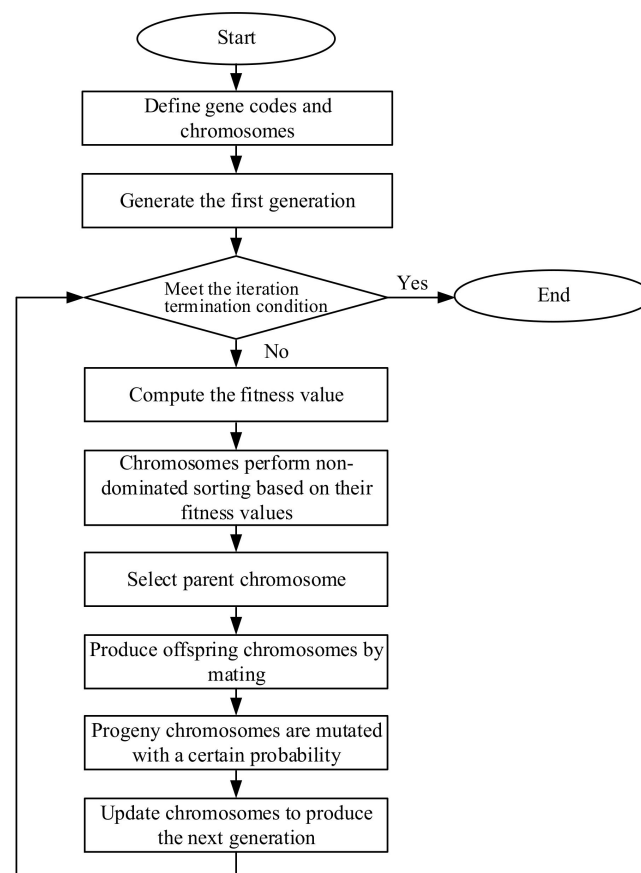
To diversify the population and reduce the probability of the algorithm falling into a local optimum in the iterative process, the offspring chromosomes generated after a crossover will mutate with a given probability. The mutation method involves a one-point mutation, as shown in Figure 3. A gene was selected randomly, and another gene was selected randomly from the workable machine for the work order corresponding to the gene for the exchange to complete a chromosome mutation. The chromosome after the

mutation also complied with the restriction of a workable machine. If the chromosome conflicted with other scheduling restrictions, the machine with the mutation would be coupled with the penalty value of idle time so that the fitness value was worse than other chromosomes and gradually eliminated in the iterative process.



**Figure 3.** Schematic diagram of mutation.

One iteration was completed using the above procedure. Each iteration generated new chromosomes, and the algorithm was iterated continuously till the algorithm reached the iteration termination condition or the maximum number of iterations. When the iteration termination condition was reached, the chromosome with the highest non-dominated rank in the population was the optimal solution set for this problem, i.e., non-dominated solution. The chromosome in the solution set was decoded to obtain the optimal scheduling result. The said implementation procedure of the algorithm is shown in Figure 4.



**Figure 4.** Execution flow chart of non-dominated sorting genetic algorithm.

#### 4. Data Implementation and Results Analysis

This study used Python 3.8 to write the program, which was implemented on a computer with an Intel i7-7700 CPU and 32 GB RAM.

##### 4.1. Data Sources

Table 1 shows partial information provided by Company G about the work orders that were scheduled for the week of January 2021. The data contained 81 work orders.

**Table 1.** Partial information about the pending work orders.

Work Order	Product Type	Material Number	Board	Due Time	WO_Qty	SH_Qty	SMT_Start Time	Machine Set	NL1	NL2	NL3	NL4	NL5	NL6	NL7	NL8	NL9
KHCNF6	SC		B		150	450			0	582	0	582	0	0	0	0	0
KHCNF6	SC		T		150	450			0	437	0	437	0	0	0	0	0
LH3N01	SC		B		80	80			0	582	0	582	0	0	0	0	0
LH3N01	SC		T		80	80			0	437	0	437	0	0	0	0	0
KGCNA6	SB		B		140	140			0	0	0	0	0	0	0	0	125
KGCNA6	SB		T		140	140			0	0	0	0	0	0	0	0	74
KGCNA7	SB		B		120	120			0	0	0	0	0	125	0	0	125
KGCNA7	SB		T		120	120			0	0	0	0	0	70	0	0	70

As the data are confidential company information, and thus, they were intentionally obscured.

The production planner at Company G arranged the daily work orders scheduled by the company's information system to appropriate production lines according to their own working experience. Due to production restrictions, even very experienced production management personnel would require about two hours to determine the scheduling. Before producing a work order, all of the materials for the work order must be ready. It is very common to find a type of product on which hundreds of components are required to be mounted using SMT in Company G. If a material (such as an integrated circuit chip) cannot arrive as scheduled after the scheduling, or any unexpected situation influences the production of a work order, the production planner has to spend time adjusting the production schedule, which involves checking the availability of material for other work orders, rearranging the production line, re-picking hundreds of new components, and so on. When the scheduling cannot be adjusted in time, the entire production line may be idle, productivity is lost, and may even lead to late delivery.

The parameters of scheduling are described in Table 2. The production lines needed to be set up before producing work orders of different product types (ProductType). The production setup time varied with the type of products of the previous work order. The work orders of the same model (MaterialNumber) needed to use the same stencil (MachineSet). Board represents the front side or back side of the work order. For the same work order number (WorkOrder), the front side of the board cannot be processed until the back side is processed. The work order cannot be processed before the job release times (SMT\_StartTime), which represent the times when all the required materials of that work order are prepared. The fields of NL1~NL9 represent the processable quantity of the types of products associated with the work order per hour on production lines 1 to 9. It can then be converted into the processing time required for each work order on each production line. If a value is 0, the associated work order cannot be processed on that production line.

To test the efficiency of the algorithm proposed in this study based on the numerical distribution of the actual order data of Company G, 12 test problems were randomly generated using a discrete uniform distribution from the delivery time, quantity processed, job release times, and NL1~NL9 fields. The distribution of parameters to be used to generate test problems is organized in Table 3. Since we did not consider the detailed settings of each production line, without loss of generality, we refer to each production line as one "machine" hereafter.

**Table 2.** Definition of each column of pending work order information.

Title	Definition
WorkOrder	Work order code
ProductType	Product category
MaterialNumber	Product type
Board	Plate type (T means front side of the board, B means the back)
DueTime	Delivery time (hours)
WO_Qty	Processing quantity (pieces)
SH_Qty	The required shipping quantity (in pieces)
SMT_StartTime	Time to start processing
MachineSet	The stencil required
NL1~NL9	Processing quantity per hour in line 1 to line 9 (pieces)

**Table 3.** Test data generation methods.

Title	DueTime	WO_Qty	SMT_StartTime	NL1~NL9
Random distribution	$U(2, 5)$	$10 \times U(0, 60)$	$U(1, 3)$	$U(0, 600)$

#### 4.2. Algorithm Parameter Settings

The parameters of a non-dominated sorting GA would significantly influence the solution efficiency of the algorithm and the quality of an optimal solution. The parameters for this algorithm were set up, including the population size, crossover rate, mutation rate, and iteration termination condition. The following data were tested, and the parameters were set using the actual work orders to be scheduled for Company G.

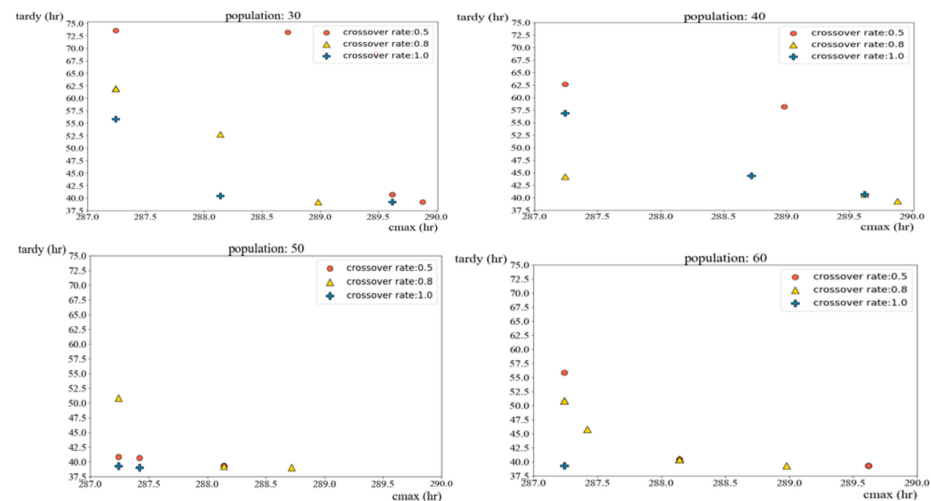
##### (1) Population size and crossover rate

With an increase in the population size, the diversity of feasible solutions increases. It enhances the global search capability of an algorithm, but the execution time of an algorithm also increases. In contrast, the execution time of an algorithm can be shortened, but the diversity of feasible solutions may be reduced. The crossover rate determines the number of chromosomes to be mated in the population. However, the crossover method used in this study was local search, and the number of chromosomes participating in crossover had a considerable influence on the computing time of the algorithm.

Grefenstette [49] indicated that a better solution could be obtained when the population size was larger than 30. The crossover rate was mostly set between 0.5 and 1.0 [50]. According to Wang et al. [51], a better solution was obtained when the crossover rate was set at 0.8. As the problem discussed in this study had a lot of restrictions, considering the execution time of an algorithm, this study tested the variation of fitness value in 12 parameter combinations with population sizes of 30, 40, 50, and 60 and crossover rates set at 0.5, 0.8, and 1.0.

Figure 5 shows the Pareto front solutions obtained by executing the algorithm using different crossover rates when the population sizes were set at 30, 40, 50, and 60. The horizontal coordinate represents the makespan and the vertical coordinate represents the total tardiness. As shown in Figure 5, when the population size was set at 30 and 40, the obtained Pareto front solutions were found at higher positions. A better Pareto front solution could be obtained when the population size was larger than 50. There was no significant difference between the population sizes of 50 and 60, and when the population size is 50 or 60, the crossover rate was set as 1.0. In this case, a better Pareto front solution could be obtained. The algorithm's execution time was shorter at a constant crossover rate and smaller population size. To obtain a better solution in a short time, the population size was set at 50 with a crossover rate of 1.0.





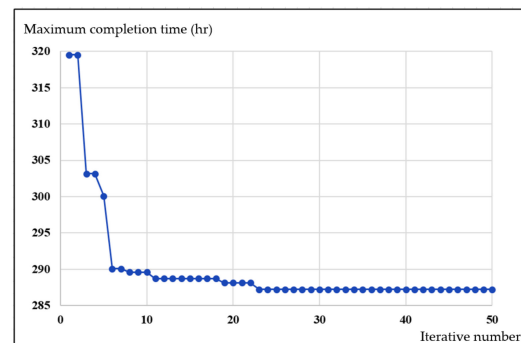
**Figure 5.** Pareto frontier solutions for different population numbers and mating rates.

## (2) Mutation rate

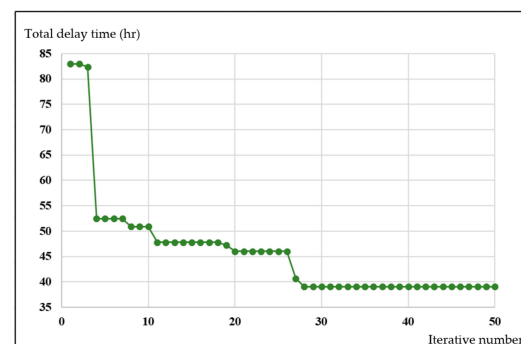
The mutation rate was mostly set between 0.005 and 0.05 [50]. As the population size was small in this study, the mutation rate was set as 0.05. The diversity of feasible solutions could be increased using a mutation in the iterative process of the algorithm.

## (3) Iteration stop condition

The maximum number of iterations was set as the algorithm termination condition in this study. The convergence of the algorithm was judged according to the fitness value's decreasing amplitude. As shown in Figure 6, the makespan became smooth after 25 iterations. As shown in Figure 7, the total tardiness became smooth after 30 iterations. Therefore, the maximum number of iterations of the algorithm was set at 30.



**Figure 6.** Decreasing trend of the maximum completion time in the iterative process of the algorithm.



**Figure 7.** The decreasing trend of the total delay time in the iterative process of the algorithm.

### 4.3. Analysis of the Experimental Results

#### 4.3.1. Experimental Results of the Test Data

For the test data of a small-scale problem, in the combined scenario with 6, 8, and 10 work orders; 2 and 3 machines; and 1 and 2 resource classes, the difference between the results of using Gurobi optimization software to solve the mathematical model and using NSGA-II to solve the problem was tested in this study to confirm the efficiency of the algorithm. The experimental results are shown in Table 4, where  $N \times M \times R$  represents the problem scale of  $N$  work orders,  $M$  machines, and  $R$  resources;  $C_{max}$  represents the makespan; and *Tardiness* represents the total tardiness. The difference in the computing mode was given as

$$\frac{\text{Algorithm target value} - \text{mathematical model target value}}{\text{mathematical model target value}} \times 100\%$$

**Table 4.** Experimental results obtained using small-scale problem test data.

No.	$N \times M \times R$	Mathematical Model			The Proposed Approach			Difference (%) ( $C_{max}$ / <i>Tardiness</i> )
		$C_{max}$	<i>Tardiness</i>	Time (s)	$C_{max}$	<i>Tardiness</i>	Time (s)	
1	$6 \times 2 \times 1$	63.04	39.04	2216	63.04	39.04	88	0.0/0.0
2	$6 \times 2 \times 2$	41.86	17.86	2282	41.86	17.86	95	0.0/0.0
3	$6 \times 3 \times 1$	81.56	0.86	472	81.56	0.86	105	0.0/0.0
4	$6 \times 3 \times 2$	101.44	5.44	1073	101.44	5.44	113	0.0/0.0
5	$8 \times 2 \times 1$	76.78	0.00	826	76.78	0.00	162	0.0/0.0
6	$8 \times 2 \times 2$	38.00	14.00	2206	38.00	14.00	175	0.0/0.0
7	$8 \times 3 \times 1$	77.32	0.22	820	77.32	0.22	128	0.0/0.0
8	$8 \times 3 \times 2$	84.33	0.00	488	84.33	0.00	112	0.0/0.0
9	$10 \times 2 \times 1$	53.54	7.75	2617	53.54	7.75	140	0.0/0.0
10	$10 \times 2 \times 2$	53.50	7.71	2548	53.50	7.71	135	0.0/0.0
11	$10 \times 3 \times 1$	82.72	9.63	3738	82.72	9.63	166	0.0/0.0
12	$10 \times 3 \times 2$	57.75	9.54	3604	58.14	9.63	163	0.7/0.9

Based on the experiments, the optimal solution could be obtained using the mathematical model. The solution of 0 for the total tardiness could also be obtained. This meant that the mathematical model built by this study was feasible and able to handle multi-objective problems. In the non-dominated sorting algorithm, besides the scenario of 10 work orders, 3 machines, and 2 resources, the two computing modes in the other scenarios could obtain optimal solutions that were equivalent to the mathematical model. The algorithm's execution time was shorter than that of the mathematical model. The effectiveness of the algorithm proposed by this study was observed, and a near-optimal solution could be obtained within a reasonable time.

#### 4.3.2. Experimental Results of Actual Work Orders to Be Scheduled

The problem scale of actual work orders to be scheduled is relatively large. It is unlikely to be solved using a mathematical model. This study only used an algorithm to solve it. The experimental results were obtained after analyzing and comparing the current scheduling method of Company G and NSGA-II. The makespan of Company G's work order to be scheduled was 319.53 h and the total tardiness was 82.99 h. The Gantt chart drawn according to the scheduling result is shown in Figure 8. The blue block represents the setup time before the work order processing. Grey and red blocks represent the work order processing time. The red blocks represent the tardiness of the work order and the blank areas represent the idle state of the machine at the time. In the Gantt chart, each work order's setup time and processing time are marked. S8,71 represents the setup time generated by processing work order 71 immediately after work order 8 was processed. S0,4 represents the setup time generated when work order 4 was the first work order on the machine. p9,71 represents the processing time of assigning work order 71 to machine 9 (i.e., the 9th production line). As shown in the Gantt chart, the machine remained in an idle state for an extended period. After comparing with the scheduling result, the primary cause was the unequal job release times. As the work orders have unequal job release times, when a

machine finished a work order, it might need to keep waiting until the next assigned work order could be processed.

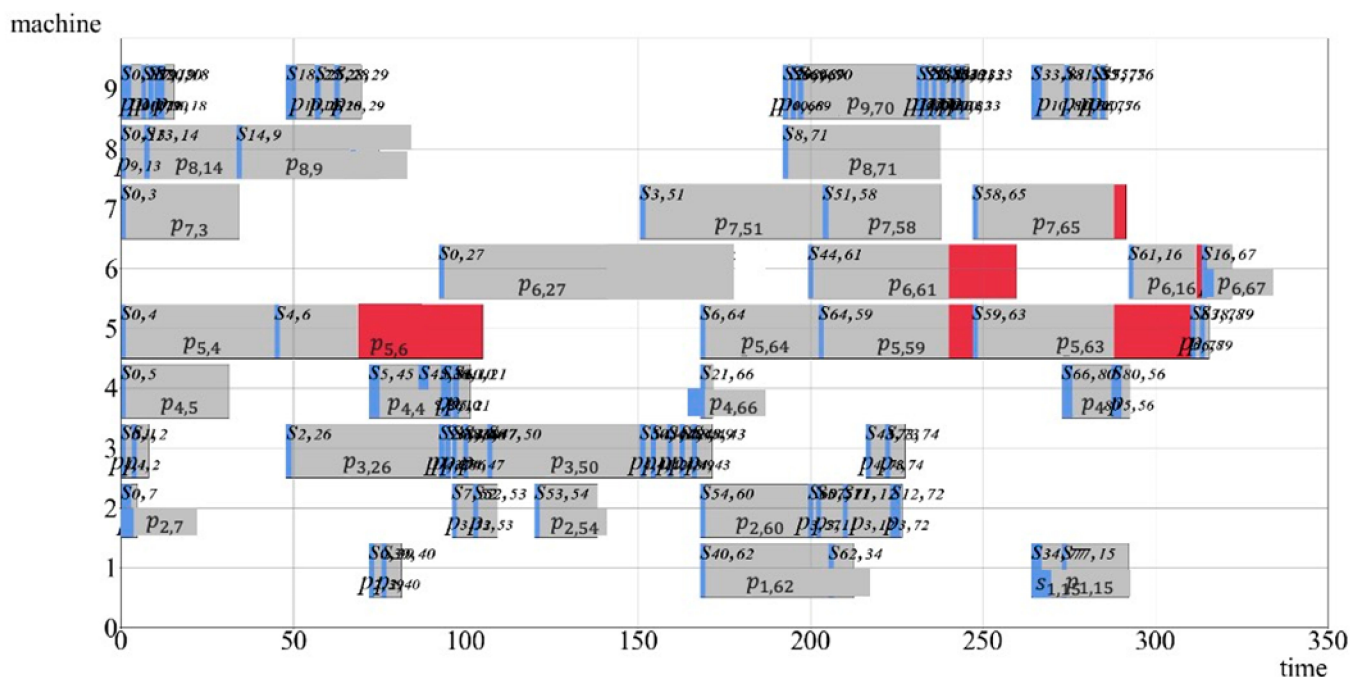


Figure 8. Gantt chart after decoding the current scheduling method.

The Pareto chart drawn according to the final solution after the execution of the proposed non-dominated sorting algorithm used in this study is shown in Figure 9. According to the two objective equations discussed in this study, the horizontal axis represents the makespan and the vertical axis represents the total tardiness. The yellow triangle means the feasible solution corresponding to the point is the Pareto front solution, i.e., the non-dominated solution. The optimal solution set was obtained using the algorithm. The corresponding two target values are shown in Table 5.

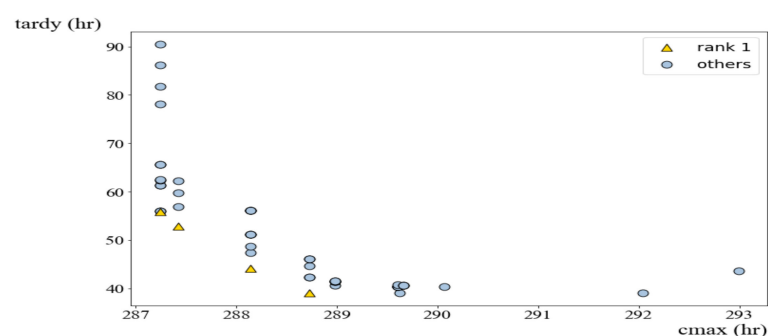


Figure 9. Pareto scatter diagram of the final solution.

Table 5. Target values of non-dominated solutions.

Non-Dominated Solution	Makespan (Hours)	Total Tardiness (Hours)
The best solution 1	287.42	52.80
The best solution 2	288.72	39.05
The best solution 3	288.14	44.16
The best solution 4	287.24	55.86

The Gantt charts drawn according to the scheduling results of four optimal solutions in Table 5 are shown in Figure 10. Compared with Figure 8, the Gantt charts drawn using the aforesaid two algorithms showed that the machine remained idle for a long time. The cause for this was also the unequal job release times, meaning this value significantly influenced the idle state of the machine.

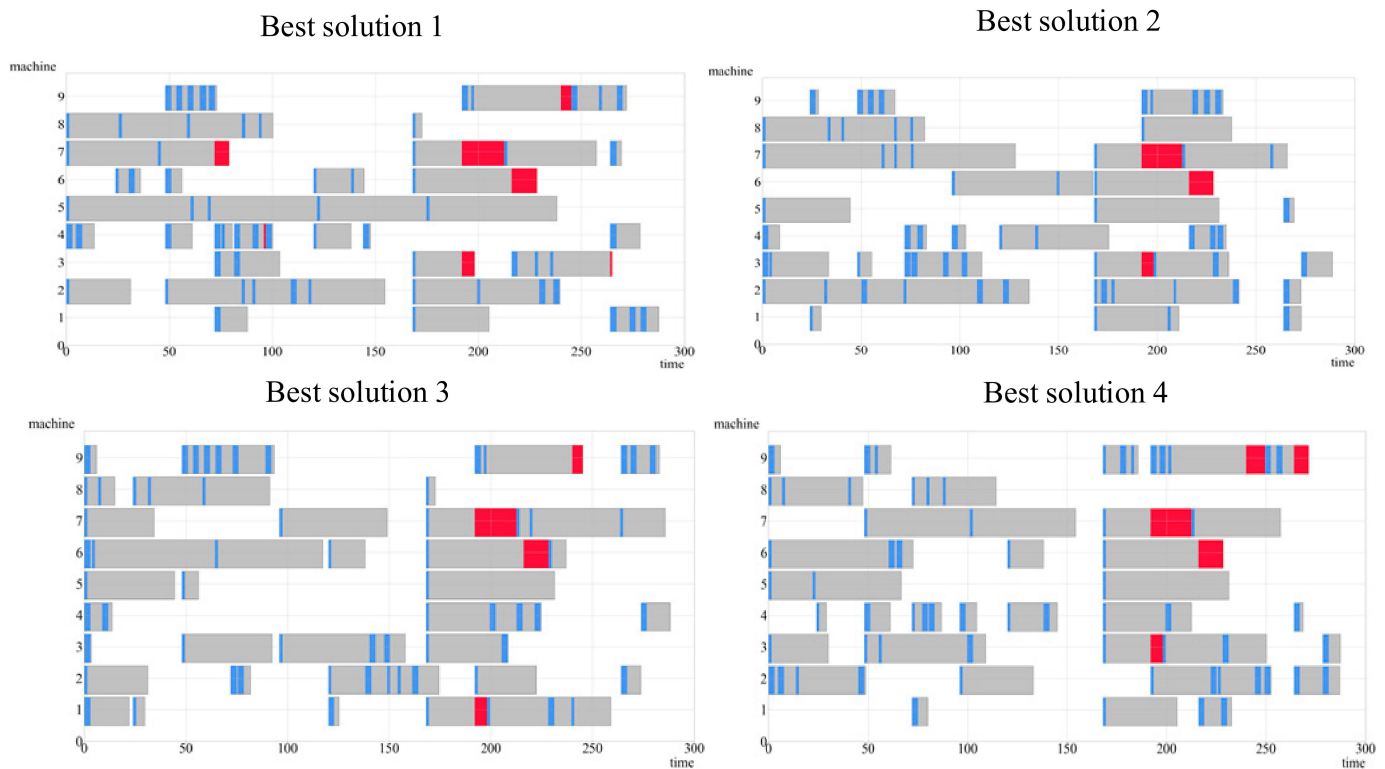


Figure 10. Gantt chart of the optimal decoding by different best solutions.

Finally, according to the solutions obtained using the two considered algorithms, the makespan and total tardiness were compared based on the current scheduling method of Company G and compiled in Table 6. The difference between the computing modes was given as

$$\frac{NSGA-II \text{ target value} - \text{target value of current scheduling method}}{\text{target value of current scheduling method}} \times 100\%.$$

Table 6. Result comparison.

	Makespan		Total Tardiness		Makespan
	Value (Hours)	Improvement (%)	Value (Hours)	Improvement (%)	
Company G's current scheduling method (baseline)	319.53	-	82.99	-	11
NSGA-II (the proposed approach)	287.42	10.0	52.8	36.4	1926
	288.72	9.6	39.05	52.9	
	288.14	9.8	44.16	46.8	
	287.24	10.1	55.86	32.7	

The makespan could be improved by 10.1% with NSGA-II and the total tardiness could be improved by 52.9% at most. As shown in Table 6, not only one solution but four feasible schedules were generated by the proposed NSGA-II approach. The execution time required by the personal computer used in this experiment was around 32 min, which is relatively

long but is still much shorter than the manual scheduling currently used by Company G. The proposed approach could assist Company G to generate a set of good-quality production schedules and could also be applied to re-scheduling whenever necessary.

## 5. Conclusions and Future Research

As the multi-objective scheduling problem has been extensively discussed, performing effective scheduling with multiple performance indicators and multiple production restrictions becomes an important issue. This study discussed the unrelated parallel machine problem with machine eligibility restrictions, sequence-dependent setup times, precedence constraints, unequal job release times, extra shared resource constraints, and taking makespan and total tardiness as the objectives. Such a problem can be found in a real production environment, such as the one discussed in this study, namely, the scheduling of SMT production for PCB manufacturing. Compared with the current approach used by the case company, namely, Company G, the proposed approach using NSGA-II could produce several feasible schedules that satisfied the two predetermined objectives.

Compared with other studies, the advantage of this study was that the precedence constraints and unequal job release times were considered so that the scheduling problem could further meet more of the restrictions found in real production lines. In addition, the common weighted method was substituted by the non-dominated method when discussing this bi-objective scheduling problem. The feasible solutions were clustered using the concept of domination, and the feasible solutions were sorted using the crowded distance so that the optimal solution set expressed the final solution. Therefore, even if the machine allocation or production environment was changed in practice, this scheduling optimization system would still be feasible. Additionally, this study used NSGA-II to solve this problem, and the final scheduling result showed that the makespan could be shortened by 10.1% at most and the total tardiness could be reduced by 52.9% at most. This way, the productivity of production lines can be increased, the company's on-time delivery is enhanced, and manual scheduling time is saved. Production scheduling is always an important activity for the case company and the production planners are reluctant to adjust an existing schedule due to the scale of complexity of this problem. Optimizing under multiple objectives is also very common in practice. This study provided a systematic approach to address this problem by minimizing job delay while increasing machine utilization. This decision support tool could help the production planners to better examine their production schedules while still providing some flexibility on the choices of the one to execute.

The following suggestions and directions are proposed for future research: (1) This study only discussed the most important first SMT process of PCB assembly, but the subsequent processes may also influence the overall scheduling. Therefore, if the other processes can be considered together in the future, the result will be more extensively used in practical problems. (2) Other meta-heuristics or local searches can be combined to enhance the solution efficiency and quality. ML algorithms can be used for solving and the scheduling problem can develop toward intelligentization. (3) In this unrelated parallel machine scheduling problem, this study only solved for optimizing two objectives: makespan and total tardiness. However, there will be more extensive performance indicators to be considered in practice. Therefore, other objectives can be discussed in the future to make the result closer to practical requirements.

**Author Contributions:** Conceptualization, Y.-C.C. and C.-P.Z.; methodology, Y.-C.C., K.-H.C. and C.-P.Z.; validation, Y.-C.C., K.-H.C. and C.-P.Z.; data curation, C.-P.Z.; writing—original draft preparation, Y.-C.C., K.-H.C. and C.-P.Z.; writing—review and editing, Y.-C.C. and K.-H.C. All authors have read and agreed to the published version of the manuscript.

**Funding:** The authors would like to thank the Ministry of Science and Technology, Taiwan, for financially supporting this research under contract nos. MOST 110-2221-E-A49-120 and MOST 110-2410-H-145-001.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

- Mumtaz, J.; Guan, Z.; Yue, L.; Wang, Z.; Ullah, S.; Rauf, M. Multi-level planning and scheduling for parallel PCB assembly lines using hybrid spider monkey optimization approach. *IEEE Access* **2019**, *7*, 18685–18700. [\[CrossRef\]](#)
- Yenisey, M.M.; Yagmahan, B. Multi-objective permutation flow shop scheduling problem: Literature review, classification and current trends. *Omega-Int. J. Manag. Sci.* **2014**, *45*, 119–135. [\[CrossRef\]](#)
- Xu, R.; Chen, H.; Li, X. A bi-objective scheduling problem on batch machines via a Pareto-based ant colony system. *Int. J. Prod. Econ.* **2013**, *145*, 371–386. [\[CrossRef\]](#)
- Natarajan, E.; Kaviarasan, V.; Ang, K.M.; Lim, W.H.; Elango, S.; Tiang, S.S. Production wastage avoidance using modified multi-objective teaching learning based optimization embedded with refined learning scheme. *IEEE Access* **2022**, *10*, 19186–19214. [\[CrossRef\]](#)
- Kato, E.R.R.; de Aguiar Aranha, G.D.; Tsunaki, R.H. A new approach to solve the flexible job shop problem based on a hybrid particle swarm optimization and random-restart hill climbing. *Comput. Ind. Eng.* **2018**, *125*, 178–189. [\[CrossRef\]](#)
- Wu, Y.; Ji, P. A scheduling problem for PCB assembly: A case with multiple lines. *Int. J. Adv. Manuf. Technol.* **2009**, *43*, 1189. [\[CrossRef\]](#)
- Chen, C.Y.; Fathi, M.; Khakifirooz, M.; Wu, K. Hybrid tabu search algorithm for unrelated parallel machine scheduling in semiconductor fabs with setup times, job release, and expired times. *Comput. Ind. Eng.* **2022**, *165*, 107915. [\[CrossRef\]](#)
- Liao, C.J.; Lee, C.H.; Tsai, H.T. Scheduling with multi-attribute set-up times on unrelated parallel machines. *Int. J. Prod. Res.* **2016**, *54*, 4839–4853. [\[CrossRef\]](#)
- Rivera, G.; Porras, R.; Sanchez-Solis, J.P.; Florencia, R.; Garcia, V. Outranking-based multi-objective PSO for scheduling unrelated parallel machines with a freight industry-oriented application. *Eng. Appl. Artif. Intell.* **2022**, *108*, 104556. [\[CrossRef\]](#)
- Bitar, A.; Dauzère-Pérès, S.; Yugma, C.; Roussel, R. A memetic algorithm to solve an unrelated parallel machine scheduling problem with auxiliary resources in semiconductor manufacturing. *J. Sched.* **2016**, *19*, 367–376. [\[CrossRef\]](#)
- Chang, Y.C.; Chang, K.H.; Chang, T.K. Applied column generation-based approach to solve supply chain scheduling problems. *Int. J. Prod. Res.* **2013**, *51*, 4070–4086. [\[CrossRef\]](#)
- Pinedo, M.L. *Scheduling: Theory, Algorithms, and Systems*, 4th ed.; Springer: New York, NY, USA, 2012.
- Garey, M.R.; Johnson, D.S. *Computers and Intractability: A Guide to the Theory of NP-Completeness*; Freeman: San Francisco, CA, USA, 1979.
- Wang, H.; Alidaee, B. Effective heuristic for large-scale unrelated parallel machines scheduling problems. *Omega-Int. J. Manag. Sci.* **2019**, *83*, 261–274. [\[CrossRef\]](#)
- Kaid, H.; Al-Ahmari, A.; Al-Shayea, A.; Nasr, E.A.; Kamrani, A.K.; Mahmoud, H.A. Metaheuristics for optimizing unrelated parallel machines scheduling with unreliable resources to minimize makespan. *Adv. Mech. Eng.* **2022**, *14*, 16878132221097023. [\[CrossRef\]](#)
- Fang, W.; Zhu, H.L.; Mei, Y. Hybrid meta-heuristics for the unrelated parallel machine scheduling problem with setup times. *Knowl.-Based Syst.* **2022**, *241*, 108193. [\[CrossRef\]](#)
- Vallada, E.; Ruiz, R. A genetic algorithm for the unrelated parallel machine scheduling problem with sequence dependent setup times. *Eur. J. Oper. Res.* **2011**, *211*, 612–622. [\[CrossRef\]](#)
- Pakzad-Moghaddam, S.H. A Lévy flight embedded particle swarm optimization for multi-objective parallel-machine scheduling with learning and adapting considerations. *Comput. Ind. Eng.* **2016**, *91*, 109–128. [\[CrossRef\]](#)
- Chang, Y.C.; Chang, K.H.; Kang, T.C. Applied variable neighborhood search-based approach to solve two-stage supply chain scheduling problems. *J. Test. Eval.* **2016**, *44*, 1337–1349. [\[CrossRef\]](#)
- Arnaout, J.P.; Rabadi, G.; Musa, R. A two-stage ant colony optimization algorithm to minimize the makespan on unrelated parallel machines with sequence-dependent setup times. *J. Intell. Manuf.* **2010**, *21*, 693–701. [\[CrossRef\]](#)
- Zitzler, E.; Thiele, L. Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach. *IEEE Trans. Evol. Comput.* **1999**, *3*, 257–271. [\[CrossRef\]](#)
- Shahvari, O.; Logendran, R. An enhanced tabu search algorithm to minimize a bi-criteria objective in batching and scheduling problems on unrelated-parallel machines with desired lower bounds on batch sizes. *Comput. Oper. Res.* **2017**, *77*, 154–176. [\[CrossRef\]](#)
- Bozorgirad, M.A.; Logendran, R. A comparison of local search algorithms with population-based algorithms in hybrid flow shop scheduling problems with realistic characteristics. *Int. J. Adv. Manuf. Technol.* **2016**, *83*, 1135–1151. [\[CrossRef\]](#)
- Zheng, X.L.; Wang, L. A two-stage adaptive fruit fly optimization algorithm for unrelated parallel machine scheduling problem with additional resource constraints. *Expert Syst. Appl.* **2016**, *65*, 28–39. [\[CrossRef\]](#)
- Afzalirad, M.; Rezaeian, J. A realistic variant of bi-objective unrelated parallel machine scheduling problem: NSGA-II and MOACO approaches. *Appl. Soft. Comput.* **2017**, *50*, 109–123. [\[CrossRef\]](#)
- Gao, K.Z.; Suganthan, P.N.; Pan, Q.K.; Chua, T.J.; Cai, T.X.; Chong, C.S. Pareto-based grouping discrete harmony search algorithm for multi-objective flexible job shop scheduling. *Inf. Sci.* **2014**, *289*, 76–90. [\[CrossRef\]](#)
- Zhang, R.; Chang, P.C.; Song, S.; Wu, C. Local search enhanced multi-objective PSO algorithm for scheduling textile production processes with environmental considerations. *Appl. Soft. Comput.* **2017**, *61*, 447–467. [\[CrossRef\]](#)



28. McNaughton, R. Scheduling with deadlines and loss functions. *Manag. Sci.* **1959**, *6*, 1–12. [\[CrossRef\]](#)
29. Yilmaz Eroglu, D.; Ozmutlu, H.C.; Ozmutlu, S. Genetic algorithm with local search for the unrelated parallel machine scheduling problem with sequence-dependent set-up times. *Int. J. Prod. Res.* **2014**, *52*, 5841–5856. [\[CrossRef\]](#)
30. Ezugwu, A.E.; Akutsah, F. An improved firefly algorithm for the unrelated parallel machines scheduling problem with sequence-dependent setup times. *IEEE Access* **2018**, *6*, 54459–54478. [\[CrossRef\]](#)
31. Ewees, A.A.; Al-qaness, M.A.; Abd Elaziz, M. Enhanced salp swarm algorithm based on firefly algorithm for unrelated parallel machine scheduling with setup times. *Appl. Math. Model.* **2021**, *94*, 285–305. [\[CrossRef\]](#)
32. Mir, M.S.S.; Rezaeian, J. A robust hybrid approach based on particle swarm optimization and genetic algorithm to minimize the total machine load on unrelated parallel machines. *Appl. Soft. Comput.* **2016**, *41*, 488–504.
33. Yepes-Borrero, J.C.; Villa, F.; Perea, F.; Caballero-Villalobos, J.P. GRASP algorithm for the unrelated parallel machine scheduling problem with setup times and additional resources. *Expert Syst. Appl.* **2020**, *141*, 112959. [\[CrossRef\]](#)
34. Zheng, X.L.; Wang, L. A collaborative multiobjective fruit fly optimization algorithm for the resource constrained unrelated parallel machine green scheduling problem. *IEEE Trans. Syst. Man Cybern.* **2016**, *48*, 790–800. [\[CrossRef\]](#)
35. Bektur, G.; Saraç, T. A mathematical model and heuristic algorithms for an unrelated parallel machine scheduling problem with sequence-dependent setup times, machine eligibility restrictions and a common server. *Comput. Oper. Res.* **2019**, *103*, 46–63. [\[CrossRef\]](#)
36. Al-Harkan, I.M.; Qamhan, A.A. Optimize unrelated parallel machines scheduling problems with multiple limited additional resources, sequence-dependent setup times and release date constraints. *IEEE Access* **2019**, *7*, 171533–171547. [\[CrossRef\]](#)
37. Fanjul-Peyro, L. Models and an exact method for the Unrelated Parallel Machine scheduling problem with setups and resources. *Expert Syst. Appl.* **2020**, *5*, 100022. [\[CrossRef\]](#)
38. Deb, K.; Pratap, A.; Agarwal, S.; Meyarivan, T.A.M.T. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans. Evol. Comput.* **2002**, *6*, 182–197. [\[CrossRef\]](#)
39. Akbar, M.; Irohara, T. NSGA-II variants for solving a social-conscious dual resource-constrained scheduling problem. *Expert Syst. Appl.* **2020**, *162*, 113754. [\[CrossRef\]](#)
40. Lin, Y.K.; Chang, P.C.; Yeng, L.C.L.; Huang, S.F. Bi-objective optimization for a multistate job-shop production network using NSGA-II and TOPSIS. *J. Manuf. Syst.* **2019**, *52*, 43–54. [\[CrossRef\]](#)
41. Ardakan, M.A.; Rezvani, M.T. Multi-objective optimization of reliability–redundancy allocation problem with cold-standby strategy using NSGA-II. *Reliab. Eng. Syst. Saf.* **2018**, *172*, 225–238. [\[CrossRef\]](#)
42. Tavana, M.; Khalili-Damghani, K.; Di Caprio, D.; Oveisi, Z. An evolutionary computation approach to solving repairable multi-state multi-objective redundancy allocation problems. *Neural Comput. Appl.* **2018**, *30*, 127–139. [\[CrossRef\]](#)
43. Li, K.; Li, D.; Wu, D.Q. Carbon transaction-based location-routing-inventory optimization for cold chain logistics. *Alex. Eng. J.* **2022**, *61*, 7979–7986. [\[CrossRef\]](#)
44. Fan, Y.Q.; Lyu, P.X.; Zhan, D.; Ouyang, K.L.; Tan, X.J.; Li, J. Surrogate model-based multiobjective design optimization for air-cooled battery thermal management systems. *Eng. Appl. Comp. Fluid Mech.* **2022**, *16*, 1031–1047. [\[CrossRef\]](#)
45. Song, Y.L.; Chen, X.; Zhou, J.L.; Du, T.; Xie, F.; Guo, H.F. Research on performance of passive heat supply tower based on the back propagation neural network. *Energy* **2022**, *250*, 123762. [\[CrossRef\]](#)
46. Bederina, H.; Hifi, M. A hybrid multi-objective evolutionary optimization approach for the robust vehicle routing problem. *Appl. Soft. Comput.* **2018**, *71*, 980–993. [\[CrossRef\]](#)
47. Long, J.; Sun, Z.; Pardalos, P.M.; Hong, Y.; Zhang, S.; Li, C. A hybrid multi-objective genetic local search algorithm for the prize-collecting vehicle routing problem. *Inf. Sci.* **2019**, *478*, 40–61. [\[CrossRef\]](#)
48. Jeon, G.; Leep, H.R.; Shim, J.Y. A vehicle routing problem solved by using a hybrid genetic algorithm. *Comput. Ind. Eng.* **2007**, *53*, 680–692. [\[CrossRef\]](#)
49. Grefenstette, J.J. Optimization of control parameters for genetic algorithms. *IEEE Trans. Syst. Man Cybern.* **1986**, *16*, 122–128. [\[CrossRef\]](#)
50. Srinivas, M.; Patnaik, L.M. Adaptive probabilities of crossover and mutation in genetic algorithms. *IEEE Trans. Syst. Man Cybern.* **1994**, *24*, 656–667. [\[CrossRef\]](#)
51. Wang, W.; Nelson, P.C.; Tirpak, T.M. Optimization of high-speed multistation SMT placement machines using evolutionary algorithms. *IEEE Trans. Electron. Packag. Manuf.* **1999**, *22*, 137–146. [\[CrossRef\]](#)