Received 22 June 2023, accepted 12 July 2023, date of publication 17 July 2023, date of current version 3 August 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3295824

RESEARCH ARTICLE

A Variable Group Parallel Flexible Job Shop Scheduling in a SMEs Manufacturing Platform

YEONJEE CHOI[1], HYUN SUK HWANG[2], AND CHANG SOO KIM[1]
[1]Department of Information Systems, Pukyong National University, Busan 608737, South Korea
[2]Research Center, Hanyoung DS, Inc., Seoul 07600, South Korea

Corresponding author: Chang Soo Kim (cskim@pknu.ac.kr)

This work was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education under Grant 2021R1I1A3052605.

ABSTRACT Manufacturing is a broad field with different types of production processes. Therefore, specific processes can accommodate multiple parallel machines operating simultaneously in some production environments. This assumption is particularly crucial in factory scheduling for industries such as textile, aircraft, and semiconductor manufacturing. The assumption proposed in this paper is differ from FJSP in that they are closer to the real world by allowing different machines to perform the same multiple processes. In this paper, a new approach for solving the flexible job shop scheduling problem has been proposed, which is referred to as the flexible job shop scheduling problem in a parallel machine environment with a variable group. We proposed a hybrid genetic algorithm-based variable neighborhood search algorithm (GA-VNS) to solve this problem, where the multiple parallel machines can operate the same operations simultaneously and are grouped as a variable group. The GA-VNS algorithm combines the global searching ability of GA with the local searching ability of VNS to fully reflect the condition of the parallel machines with variable groups. The objective functions of the algorithm are to minimize the production makespan and to improve facility usage. Multiple simulation experiments are conducted using forty-seven test instances to assess the feasibility and effectiveness of the new approach. This study is expected to reduce processing time and production costs for Small and Medium Enterprises (SMEs) with low cost and high efficiency compared to existing systems, such as the manufacturing execution system, enterprise resource planning, and job shop scheduling system.

INDEX TERMS Genetic algorithms, variable neighborhood search algorithm, flexible job shop scheduling problem, global and local searching algorithm, parallel machines, small and medium sized enterprises, manufacturing platform.

I. INTRODUCTION

The manufacturing industry is embracing multiple shifts due to the multi-product and small-volume production system trend [1]. This requires manufacturing companies to stock up on various raw materials and adopt the latest digital technologies. To keep up with this trend, SMEs need to improve and develop their manufacturing technologies. Hence, the manufacturing industry has adopted information technology systems to keep pace with the changes. Information technology systems are used extensively in manufacturing, and their users and goals are varied. The goals include energy savings, sales growth, automatic systems, flexible manufacturing, real-time monitoring, and optimal scheduling [2], [3]. Building these systems requires the input of human research resources, the introduction of new equipment and cutting-edge technologies, adaptability to changes in existing human resources, and abundant capital investment [4], [5], [6].

However, SMEs are not always able to meet the demands of change. Mittal et al. [6] suggest that while several SMEs have a wide range of opportunities and high economic value creation potential, often fail to realize this potential due to factors such as their low capital, limited use of advanced manufacturing technologies, inflexible corporate culture, and

The associate editor coordinating the review of this manuscript and approving it for publication was Zhiwu Li.

lack of investment in research and development. Therefore, the high barriers to entry for smart manufacturing systems prevent many SMEs from fully exploiting the potential economic benefits of these technologies and reaping the financial benefits.

There are two types of manufacturing software commonly used in manufacturing systems today: enterprise resource planning (ERP) and manufacturing execution system (MES). MES primarily focuses on information systems that improve the quality, accessibility, cost-effectiveness, and operability of manufacturing systems. These systems provide users with a variety of functions. Among these functions, scheduling systems are most commonly used for job shop scheduling in manufacturing [7]. Job shop scheduling reduces delivery time and energy consumption and improves manufacturing efficiency.

The digital revolution in manufacturing systems has transformed the relationship between buyers and producers. Buyers expected providers to share shipping information and provide customized orders. To meet this demand, providers need to stockpile a variety of resources and streamline their production processes. While large enterprises are well-equipped to meet these demands, SMEs are struggling with limited resources in comparison. Therefore, when ordering large quantities of customized products, buyers may need to find two or more SMEs for different products. Furthermore, because SMEs rely on existing buyers, they may have difficulty finding or reaching out to new SMEs, and they may have difficulty assessing their reliability before placing an order [8], [9], [10], [11]. These trends in buyer-producer relationships can impact how new SMEs build new relationships with potential buyers.

In this sense, the main function of the proposed job shop scheduling engine is to schedule production tasks. The job shop scheduling engine aims to efficiently schedule the production process to minimize the delivery time for buyers' orders. By efficiently executing scheduling, the job shop scheduling engine can solve the problem of job shop scheduling and ensure optimal makespan for the production processes [12]. Numerous attempts have been made to solve the job shop scheduling problem (JSSP) in the manufacturing industry. Shao and Kim [13] proposed a self-supervised long-short term memory (SS-LSTM) to solve the JSSP. Solving the NP-hard problem of JSSP yields an optimal schedule, but there are still limitations in practical implementation [14]. JSSP is constrained by the condition that each machine can only process single job at a time. As a solution to this limitation, the flexible job shop scheduling problem (FJSP) was considered, where operation of each job can be processed on multiple machines. Li and Gao [15] proposed a hybrid algorithm (HA) to solve FJSP, which combines genetic algorithm (GA) and tabu search (TS) to enhance both global and local search capabilities. However, the method may not be globally preferable in other cases, as the algorithm relies on predefined conditions.

Hence, various studies have proposed and modified scheduling approaches to solve specific problems based on their requirements. Li et al. [16] proposed a hybrid self-adaptive differential evolution (HSDE) algorithm with heuristic strategies and set job priority and outsourcing operation constraints. Luo et al. [17] proposed an improved memetic algorithm based on the non-dominated sorting genetic algorithm II (NSGA-II) structure to solve the classical distributed flexible job shop scheduling problem, with the aim of achieving minimum makespan, maximum workload of machines, and variable workload of workers. Fattahi et al. [18] applied a novel approach to FJSP and solved the problem by utilizing a simulated annealing (SA) algorithm. The objective of the study is to minimize the makespan and optimize the machine usage efficiency. Tian et al. [19] proposed a novel multi-objective bi-population differential artificial bee colony (BDABC) algorithm to generate the energy-efficient scheduling of multi-variety and small batch flexible job shop operations in aerospace manufacturing facility. The study utilizes mathematical modeling techniques to consider multiple variables. Lin et al. [20] applied a learning-based cuckoo search (LCS) algorithm to minimize makespan in multiple machines and various routing operations for each job environment. The aforementioned new approaches to solving FJSP will lead to more realistic and feasible solutions to real-world scheduling problems.

In this paper, a new approach to FJSP is proposed by utilizing parallel machines with variable group conditions. Variable group (VG) is used for job splitting and machine allocation. In traditional FJSP, each job consists of a set of operations that must be executed on a group of machines, each with a generated processing time. In FJSP with parallel machines and variable groups, each job still consists of a set of operations, but these operations are grouped into a set of variable groups. In a real-world scheduling scenario, a job can consist of multiple operations and can be processed in parallel on multiple machines. In this sense, VG technique specifies operations based on the number of parallel machines available for processing. FJSP with parallel machines and variable groups utilizes GA for the global searching function, and variable neighborhood search (VNS) algorithm for the local search function. GA is powerfully effective in processing global constraints through its encoding and decoding features [21]. The local search performed by VNS includes a machine selection [22] and an iterative best improvement algorithm to enhances its ability to perform local search and to address the complex constraints of the proposed approach.

The main contributions of this paper are as follows.

- To validate the effectiveness and feasibility of the proposed job shop scheduling engine, we present a case study utilizing a real-world manufacturing data-driven example.
- Designing a modified FJSP on parallel machines with variable groups should be feasible in real-world solutions.

- The GA algorithm has powerful efficiency in global searching ability, and VNS is used to achieve local searching ability to make best combination of machines during the searching process.
- The optimal combination of parameters for GA-VNS is determined using an orthogonal experimental approach.

The remainder of this manuscript is organized as follows. Section II reviews the flexible job shop scheduling problem and related work on parallel machine scheduling. Section III describes the architecture and procedures of the job shop scheduling engine. In Section IV, we introduce the FJSP algorithm for parallel machines with variable groups, conduct some numerical experiments, and reports the results. Section V concludes this work and outlines directions for future studies.

II. RELATED RESERACH

A. FLEXIBLE JOB SHOP SCHEDULING PROBLEM

The FJSP is the NP-hard problem and is more complex than traditional JSSP. FJSP has two additional conditions compared to JSSP. First, each operation in the job must be assigned to only one machine among the available machines. Second, unlike JSSP, FJSP allows different tasks to be assigned to the same job. Mathematical programming methods are considered to be accurate methods for solving problems. However, when dealing with a large number of tasks and machines, it can be difficult to solve problems using only mathematical programming methods. Due to the difficulty of solving FJSP with traditional algorithms, various metaheuristic methods have been developed to approach the problem. The most widely used metaheuristic methods for solving FJSPs include simulated annealing (SA), tabu search, genetic algorithm (GA), particle swarm optimization (PSO), and hybrid algorithms [23].

The concept of FJSP was first introduced by Brucker and Schlie [24], who developed a polynomial algorithm to obtain the minimum makespan. Pezzella et al. [25] utilized a GA to solve FJSP and conducted computational experiments to compare its efficiency with tabu search algorithm. Chen et al. [26] proposed a self-learning genetic algorithm (SLGA) to achieve optimal scheduling results. In SLGA, the Q-learning algorithm was shown to be efficient in selecting the optimal parameters of the GA. Singh and Mahapatra [27] proposed quantum-behaved particle swarm optimization (QPSO) to reduce the makespan. QPSO was used to minimize the drawbacks of PSO, which is well known for being trapped in local optima. Yuan et al. [28] proposed a hybrid harmony search (HHS) algorithm to minimize the makespan. The HHS algorithm utilized the global and local search capabilities of harmony search to reduce the search space and evolve the harmony vector. Jamili et al. [29] presented a periodic job shop scheduling problem and propose a particle swarm optimization simulated annealing algorithm (PSO-SA) to reduce the makespan. To address the limitations of individual algorithms, researchers have begun to use hybrid approaches. Therefore, we propose a hybrid approach that combines GA and VNS to improve the global and local search abilities of the algorithm.

B. PARALLEL MACHINE SCHEDULING

Parallel machine scheduling is a very important aspect of job shop scheduling for manufacturing in real-world scenarios. There are three types of machines in parallel machine scheduling problem: identical machines, uniform machines, and unrelated machines. Identical machines have the same processing time due to their similar performance. Asadpour et al. [30] studied on how to solve the identical parallel machine scheduling problem by minimizing the tardiness and total energy consumption. They developed a simulated annealing algorithm and applied a harmony search algorithm for large-scale problems. Cohen et al. [31] also solved parallel machine scheduling under identical machines conditions. The study aimed to propose a better solution and achieve a stable makespan for robust optimization.

The second type of parallel machines, known as uniform machines, are defined as machines with separate performances, but each machine operates at a constant speed. Li et al. [32] proposed a fuzzy simplified swarm optimization algorithm (SSO) to minimize the makespan of uniform parallel machine scheduling. Lin and Ying [33] proposed a particle swarm optimization meta-heuristic and a lower bound of total resource consumption for uniform parallel machines.

The unrelated machines have different processing times for jobs including identical jobs. Fang et al. [34] conducted a study on an unrelated parallel machine scheduling problem with the objective of minimizing the makespan. The study proposed a hybrid meta-heuristic method based on adaptive large neighborhood search with learning automata (LA-ALNS) and tabu search with a dynamic perturbation scheme. Computational results were provided to validate the proposed method. Moser et al. [35] utilized a mathematical model for small instances and simulated annealing to solve the large instances. The objective of the study was to minimize total tardiness and makespan while considering sequence-dependent setup times, due dates, and machine eligibility constraints. Maecker et al. [36] implemented a variable neighborhood search algorithm to solve the unrelated parallel machine scheduling problem in a distributed manufacturing environment, where considering job-machine-dependent delivery times and eligibility constraints. Their study aims to minimize the total makespan.

In the parallel machine scheduling problem, efficient job-machine assignment is essential to find the optimal makespan. This problem utilizes the concept of parallel machines to reflect a real-world scheduling problem. However, existing parallel machine scheduling research differs from this study. In this study, multiple machines with varying velocities are operated on the same operations, which is similar to the parallel machine scenario.

III. VARIABLE GROUP PARALLEL FLEXIBLE JOB SHOP SCHEDULING

The FJSP on parallel machines with variable groups assume manufacturing scenarios for SMEs.

A. A JOB SHOP SCHEDULING USING GA-VNS

The FJSP on parallel machines with variable groups can be described as follows. The problem involves scheduling a set of jobs on a set of machines in order to minimize the makespan or total completion time of all the jobs. This problem is classified as a FJSP with the additional constraint of variable groups. Each product is treated as a single job on the buyer's order. Each job $i = \{1, 2, 3, \ldots, n\}$ can have $j = \{1, 2, 3, \ldots, n_i\}$ operations $O_{i,j}$, and can be executed on $M = \{1, 2, 3, \ldots, m\}$ machines, The processing time $T_{i,j,m}$ for each $M_{i,m}$ varies depending on the $O_{i,j}$ being processed. In addition, each $O_{i,j}$ is assigned a $VG_{i,j,m}$ to group the parallel m that is capable to process the $O_{i,j}$. This paper utilized the notations listed in Table 1. The constraints of the proposed FJSP algorithm are as follows:

1) Jobs are independent and cannot be preempted.

2) No other jobs can be processed simultaneously on any machine for a given job.

3) All jobs can start at time zero.

4) When a job ends, the machine immediately starts the next job, but it can be delayed by other jobs.

5) For the identical operations of the same job, pre-assigned machines can process simultaneously depending on the size of the VG.

6) Machine setup time and product transfer time are not considered.

TABLE 1. Notations for FJSP in parallel machine with variable group.

Parameters	DESCRIPTION
n	number of jobs
m	machine numbers
i	job numbers
j	operation numbers
$O_{i,j}$	j^{th} operation of n_i
$M_{i,m}$	available machine sets for operation $O_{i,j}$
$T_{i,j,m}$	processing time of each operation on each machine
$C_{i,j,m}$	completion time of each operation on each machine
$VG_{i,j,m}$	variable group of parallel machines of each operation
B	number of buyers
P	number of producers

The decision variables and mathematical formulation of proposed algorithm are as follows:

$$minT = max(\sum\nolimits_{i=0}^{n} T_{i,j}) \tag{1}$$

$$(C_{i,j,m} - C_{(i-1),(j-1),m} - (T_{i,j,m})) \geq 0 \tag{2}$$

(1) is objective function for minimization of makespan. (2) ensures that one operation can be performed at a time on any machine.

FJSP on parallel machines with variable groups is based on a hybrid algorithm that combines GA and VNS. Because parallel machines with variable groups generate multiple permutations, this algorithm can be highly effective in searching both the global and local domains simultaneously. Fig. 1 shows the overall procedure of the algorithm, and the pseudocode of GA-VNS is shown in Algorithm 1.

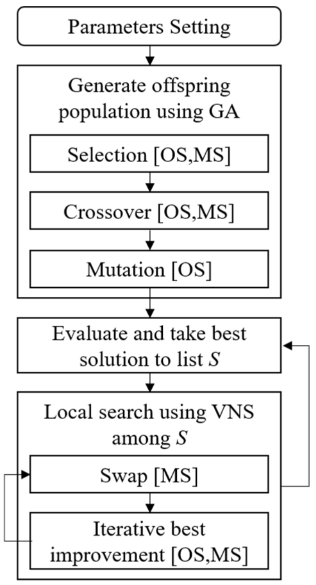
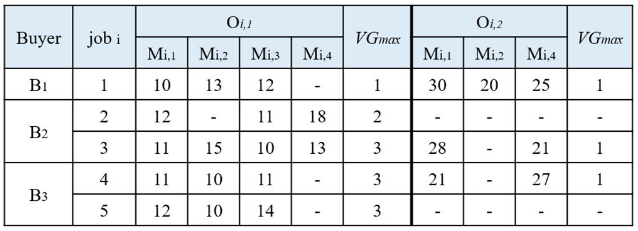

Parameters Setting
Generate offspring population using GA
Selection [OS,MS]
Crossover [OS,MS]
Mutation [OS]
Evaluate and take best solution to list S
Local search using VNS among S
Swap [MS]
Iterative best improvement [OS,MS]

FIGURE 1. An overall procedure of GA-VNS.

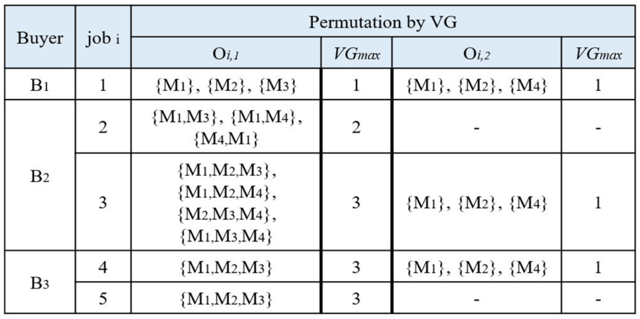


Buyer	job i	$O_{i,1}$				VG_{max}	$O_{i,2}$			VG_{max}
		$M_{i,1}$	$M_{i,2}$	$M_{i,3}$	$M_{i,4}$		$M_{i,1}$	$M_{i,2}$	$M_{i,4}$	
B1	1	10	13	12	-	1	30	20	25	1
B2	2	12	-	11	18	2	-	-	-	-
	3	11	15	10	13	3	28	-	21	1
B3	4	11	10	11	-	3	21	-	27	1
	5	12	10	14	-	3	-	-	-	-

FIGURE 2. Possible datasets with 5 jobs, 2 operations, and VG values for 4 machines.

Fig. 2 shows the FJSP dataset and corresponding VG values for parallel machines with variable groups considering five jobs on four machines. Different product types have different operations. In SMEs manufacturing, the production process is typically divided into multiple operations depending on the product type. Moreover, while the machines used for each operation may vary, their specifications are usually similar, so they can be treated as parallel machines in this paper.

This means that when a buyer orders a product, it is divided into multiple jobs via $O_{i,j}$ operations and allocated to available machines based on the VG setting for each job. If the VG of each job does not exceed the available parallel machines, the VG setting can be set autonomously. The algorithm starts by permuting the machines and operations to fully reflect the VG settings, as shown in Fig. 3. It then encodes the operation string (OS) and machine string (MS) to generate the population of GAs.



Buyer	job i	Permutation by VG			
		$O_{i,1}$	VG_{max}	$O_{i,2}$	VG_{max}
B1	1	{M1}, {M2}, {M3}	1	{M1}, {M2}, {M4}	1
B2	2	{M1,M3}, {M1,M4}, {M4,M1}	2	-	-
	3	{M1,M2,M3}, {M1,M2,M4}, {M2,M3,M4}, {M1,M3,M4}	3	{M1}, {M2}, {M4}	1
B3	4	{M1,M2,M3}	3	{M1}, {M2}, {M4}	1
	5	{M1,M2,M3}	3	-	-

FIGURE 3. Permutation of machine lists for each operation according to VG values.

After generating the OS and MS, the algorithm performs GA operations such as selection, crossover, and mutation. In this paper, a dynamic GA approach is required since the algorithm is based on solving the FJSP. The selection operator adopts elitist selection to reproduce offspring from parents. The elitist selection selects the optimal parent population based on a fitness function, which can be calculated by multiplying a predefined reproduction probability by 0.005. Two crossover operators are then applied for each OS and MS string: precedence operation crossover and job-based crossover. If there are VGs in the population, the job set needs to be partitioned by considering the VG size of each job.

Algorithm 1 GA-VNS Algorithm

```
begin
  generate initial population by encoding procedure;
  gen ← 1
  while (not termination criteria)
    generate offspring population from selection, crossover, and
    mutation operators;
    evaluate;
    take best solutions by decoding procedure;
    update best solutions list using VNS;
    evaluate;
    if best solution < new best solution then
      best solution ← new best solution
    end
    gen ← gen + 1
  end
  output:best solution with minimum makespan
end
```

The MS string used two-point crossover to impart variation to the offspring populations. The two-point crossover randomly selects two points in the MS string and swaps the two parent and offspring strings, i.e., P1 and O1 and P2 and O2. The remaining positions in the two offspring are then filled with two parent strings. If a VG value exists, the process must also take it into account.

The last operator in the GA operation is a mutation that swaps specified positions in the OS string using a swapping mutation. Similar to the two-point crossover, a swapping mutation randomly selects two positions in the parent string and swaps the elements in the selected positions to generate

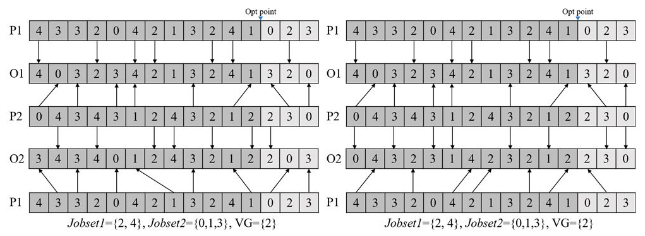

Opt point
P1
O1
P2
O2
P1
Jobset1={2, 4}, Jobset2={0,1,3}, VG={2}
Opt point
P1
O1
P2
O2
P1
Jobset1={2, 4}, Jobset2={0,1,3}, VG={2}

FIGURE 4. (a) Precedence operation crossover (b) job-based crossover of OS string.

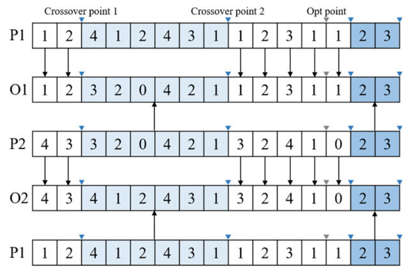


FIGURE 5. MS string two-point crossover according to each VG.

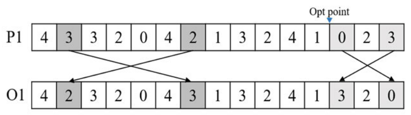


FIGURE 6. OS swapping mutation.

offspring. Once the mutation operation is complete, the decoding procedure begins as the final step in GA.

For crossover, Fig. 4(a) shows a precedence operation crossover and Fig. 4(b) shows a job-based crossover of the OS string. Fig. 5 illustrates a two-point crossover of the MS string when a VG value is present. Fig. 6 shows an OS string swapping mutation. In all operations, the OS and MS strings are divided into operation points due to parallel machines with variable group conditions.

The decoding procedure calculates the processing time of each population so that the algorithm can choose the optimal solution. It takes into account the OS string to interpret the sequence of each job by considering the OS string and matches it with the MS string to obtain the processing time. Meanwhile, there are constraints that need to be considered during the decoding process.

First, regardless of the priority of the job, if the machine is idle, the job should be assigned to time zero. Second, if multiple operations of the same job on parallel machines do not end simultaneously, the next operations of the job cannot be processed by other machines. In this way, the decoding result is shown as follows: [[('2-1', 0, 21), ('0-1', 21, 31)], [('2-1', 0, 10)], [('2-1', 0, 11)], [('1-1', 0, 28)], [('1-1', 0, 13)]]. The decoding result is formatted as lists, with each sub list indicating job with operation number, start processing time, an end processing time. Besides, the large lists are indicating each machine. The makespan of the decoded example

Algorithm 2 VNS Algorithm

```
input: $k_{max}$ = number of neighborhood structure
       $gen_{max}$ = number of iterations
begin
  k ← 1
  while $k < k_{max}$
    gen ← 1
    generate initial population by encoding procedure;
    while $gen < gen_{max}$
      randomly select n in [0, 1];
      if n == 0 then
        apply swap MS string to generate new solution;
      else
        apply iterative best improvement algorithm to MS and OS
        string to generate new solution;
      end
      if best solution < new best solution then
        best solution ← new best solution
        k ← 1
        gen ← gen + 1
        if $gen = gen_{max}$ then
          k ← k + 1
        end
      else
        k ← k + 1, gen ← $gen_{max}$
      end
    end
  end
  output: best solution with minimum makespan
end
```

TABLE 2. Results of parameters comparison.

Trial	PS	CR	MR	NS	CPU(s)	C_{max}	Min-Max
1	50	0.6	0.1	50	0.63	817	0.71
2	50	0.7	0.4	200	0.63	794	0.32
3	50	0.8	0.2	300	0.63	834	1
4	50	0.9	0.3	100	0.63	802	0.45
5	50	0.5	0.5	150	0.63	817	0.71
6	100	0.6	0.2	100	2.58	809	0.57
7	100	0.7	0.3	300	2.57	811	0.61
8	100	0.8	0.1	200	2.57	803	0.47
9	100	0.9	0.4	50	2.56	793	0.30
10	100	0.5	0.5	150	2.54	794	0.32
11	150	0.6	0.1	100	6.54	803	0.47
12	150	0.7	0.3	50	5.78	816	0.69
13	150	0.8	0.2	300	5.74	783	0.13
14	150	0.9	0.5	200	5.87	799	0.40
15	150	0.5	0.4	150	5.77	777	0.03
16	200	0.6	0.3	200	10.21	792	0.28
17	200	0.7	0.2	50	10.19	794	0.32
18	200*	0.8*	0.4*	100*	10.19	775	0
19	200	0.9	0.1	300	10.18	812	0.62
20	200	0.5	0.5	150	10.16	781	0.10
21	300	0.6	0.4	300	23.12	777	0.03
22	300	0.7	0.1	100	23.17	789	0.23
23	300	0.8	0.3	50	23.14	783	0.13
24	300	0.5	0.5	150	23.10	783	0.13
25	300	0.9	0.2	200	22.92	795	0.33

population is determined by the maximum of the processing time value, 31.

VNS is a metaheuristic optimization method that iteratively explores the solution space of a problem by exploring

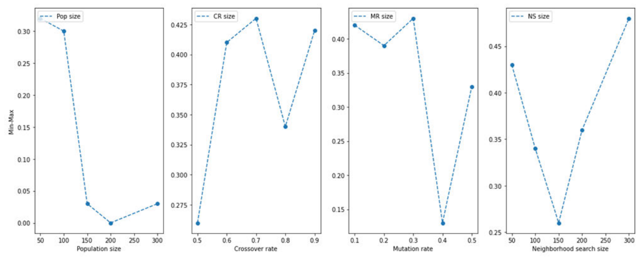


FIGURE 7. The Min-Max normalization value of four parameters in the orthogonal experiment.

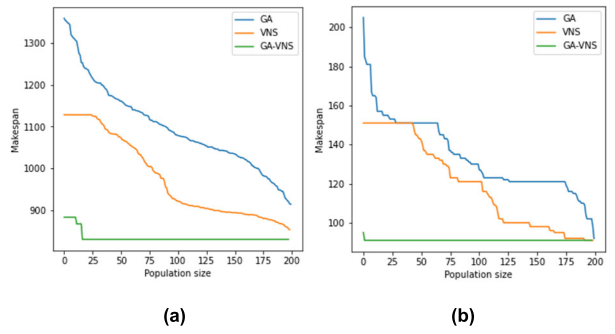


FIGURE 8. (a) VW06 and (b) VW34 dataset minimum makespan result of GA, VNS, and GA-VNS.

the neighborhood of the current solution. The algorithm starts with an initial solution and then iteratively explores the solution space by examining the neighborhood of the current solution. The neighborhood is defined as the set of move operators used to generate new solutions from the current solution.

A key feature of the VNS algorithm is that it can use different neighbors at different stages of the search process. During the local search, the VNS algorithm swaps MS strings to find a better solution. By performing transformations on the MS strings without changing the OS strings, the algorithm can move beyond local optimization and explore other parts of the solution space. In addition, the novel approach of the proposed algorithm is based on multiple parallel machine allocation, so it is robust even if many transitions occur in the MS string during scheduling. The pseudocode of the VNS algorithm is shown in Algorithm 2.

After generating the initial population, the VNS algorithm randomly selects one number from the set {0, 1}. If the number is 0, the swapping operator is applied to the MS string. The swapping operator selects a neighboring MS string and swaps it with the first MS string. If the number is 1, an iterative best improvement algorithm is applied to both the OS and MS strings. This algorithm is based on greedy descent, where the evaluation function calculates the makespan, scores the minimum makespan, and then compares it to other makespans. In this sense, the iterative best improvement algorithm finds the optimal solution at each iteration and exchanges the current string with the new string.

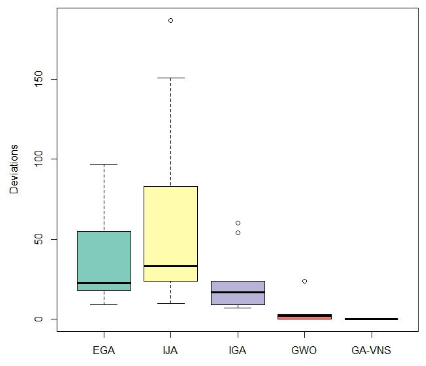


FIGURE 9. Boxplot of the distribution of deviations of the algorithms.

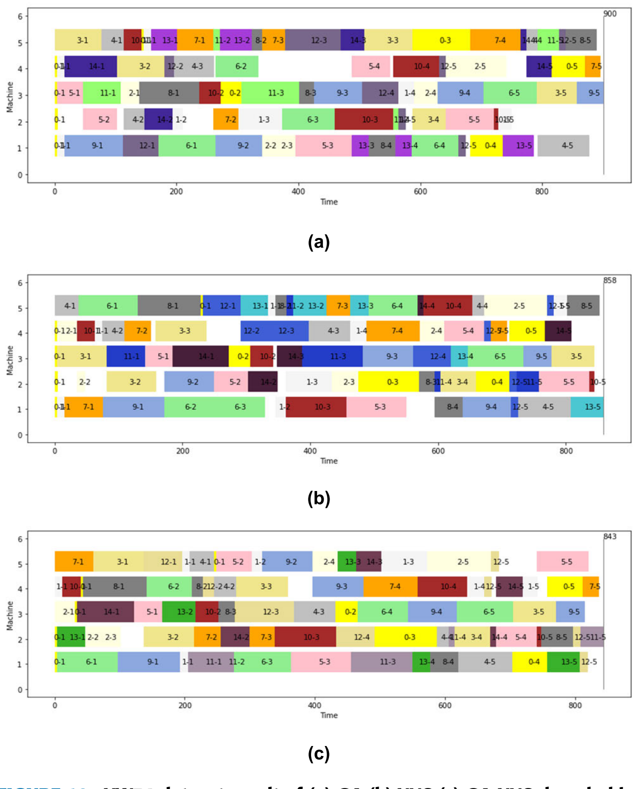


FIGURE 10. VW34 dataset result of (a) GA (b) VNS (c) GA-VNS decoded in gantt chart.

IV. COMPARISON EXPERIMENTS AND DISCUSSION

The proposed algorithm was implemented in a Jupyter notebook and executed on a PC equipped with an Intel(R) Core(TM) i7-10700 CPU@2.90GHz, Windows 10, and 32GB RAM. Since there are no benchmark instances for FJSP on parallel machines with variable groups, we compared our proposed algorithm without the variable group condition to nineteen benchmark instances from Fattahi et al. [37], which include small size flexible job shop scheduling problems (SFJS1:10) and the medium size flexible job shop scheduling problems (MFJS 1:9, 10). GA-VNS is compared to artificial immune algorithm (AIA) [26], hybrid harmony search (HHS) [26], SA [18], and genetic algorithm and tabu search (GA-TS) [15]. The additional ten FJSP benchmarks (MK01:10) [38] were chosen for comparing the proposed algorithm with enhanced genetic algorithm (EGA) [39], improved JAYA algorithm (IJA) [40], improved genetic algorithm (IGA) [41], and grey wolf optimization algorithm (GWO) [42]. In addition, we formulated thirty-nine instances to test the proposed algorithm. Even though the instances were manually formulated, the datasets were based on the benchmark dataset of the large size flexible job shop scheduling problems (LFJS 1:5) from Brandimarte [38], and the VW (1:34) instances are generated randomly by setting the number of jobs, number of operations, number of machines, and processing times, and variable groups.

For the GA algorithm to run, key parameters must be determined. Depending on the parameter settings, the makespan may vary. Therefore, a comparative approach was used to determine the optimal parameter settings for GA and GA-VNS. For the comparison experiment, each parameter was organized into four levels: a population size group (PS) consisting of [50, 100, 150, 200, 300], a crossover rate group (CR) consisting of [0.5, 0.6, 0.7, 0.8, 0.9], a mutation rate group (MR) consisting of [0.1, 0.2, 0.3, 0.4, 0.5], and a neighborhood search size group (NS) consisting of [50, 100, 150, 200, 300]. To validate the parameter combinations, Min-Max normalization was used as shown in Equation (3).

$$\frac{x - x_{min}}{x_{max} - x_{min}} \tag{3}$$

The comparison results depend on the Min-Max normalization value of the makespan, using VW 34 dataset. According to Table 2, the optimal parameter combination is PS = 200, CR = 0.8, MR = 0.4, and NS = 100. However, to consider the value range of individual parameters, the orthogonal experiment was conducted. Fig. 1 shows the Min-Max normalization values of the four parameters in the orthogonal experiment. The average of each parameter was calculated as the Min-Max value. Therefore, the orthogonal experiment allowed the proposed algorithm to identify a reasonable value range for each factor, and the final best parameter combination is PS = 200, CR = 0.5, MR = 0.4, and NS = 150.

Table 3, Table 4, and Table 5 present comparison of the GA-VNS approach with other algorithms. C_{max} refers to the makespan of each instance, which CPU(s) indicate the CPU time per second with the RAM memory usage. Additionally, in Table 5, VG_{max} refers to the maximum size of the VG across all jobs in each instance. In the small-size instances, AIA, HHS, SA, and GA-TS produce the identical makespan results as GA-VNS due to the small scale of jobs and machines. Meanwhile, this trend decreases as the instance size of the increases. GA-VNS achieved the minimum makespan in most of the experiments except for one. Despite remarkable results of the proposed method, the

TABLE 3. Results of VGPFJSP scheduling with SJFS, MJFS, and LJFS.

Instances	Size	AIA [26]		HHS [26]		SA [18]		GA-TS [15]		Proposed		
		C_{max}	CPU(s)	C_{max}	CPU(s)	C_{max}	CPU(s)	C_{max}	CPU(s)	C_{max}	GAP	CPU(s)
SJFS1	2×2	66*	0.03	66*	0.00	66*	0.00	66*	0.00	66*	0%	0.00
SJFS2	2×2	107*	0.03	107*	0.00	107*	0.00	107*	0.00	107*	0%	0.00
SJFS3	3×2	221*	0.04	221*	0.00	221*	0.00	221*	0.00	221*	0%	0.00
SJFS4	3×2	355*	0.04	355*	0.00	363	0.00	355*	0.00	355*	4%	0.00
SJFS5	3×2	119*	0.04	119*	0.00	119*	0.00	119*	0.00	119*	0%	0.00
SJFS6	3×3	320	0.04	320	0.00	330	0.00	320	0.00	287*	23%	0.00
SJFS7	3×5	397	0.04	397	0.00	397	0.00	397	0.00	267*	49%	0.00
SJFS8	3×4	253	0.05	253	0.00	283	0.00	253	0.00	218*	37%	0.00
SJFS9	3×3	210	0.05	210	0.00	295	0.00	210	0.00	190*	53%	0.00
SJFS10	4×5	516	0.05	516	0.00	641	0.00	516	0.00	435*	49%	0.00
MJFS1	5×6	468	9.23	468	0.01	513	0.00	468	0.00	465*	17%	0.00
MJFS2	5×7	448	9.35	446	0.01	437	0.00	446	0.00	423*	11%	0.00
MJFS3	6×7	468	10.06	466*	0.12	641	0.00	466*	0.02	466*	43%	2.17
MJFS4	7×7	554	10.54	554	0.06	552*	0.00	554	0.02	552*	1%	3.72
MJFS5	7×7	527	10.61	514*	0.02	523	0.00	514*	0.02	514*	5%	3.78
MJFS6	8×7	635	22.18	634	0.01	547*	0.00	634	0.01	585	24%	0.00
MJFS7	8×7	879	24.82	879	0.11	576*	0.00	879	0.08	685	51%	5.11
MJFS9	11×8	1088	30.76	1055	0.94	709	0.00	1055	0.48	1047*	52%	8.23
MJFS10	12×8	1067	30.94	1096	0.69	766	0.00	1196	0.59	1051*	53%	8.90

TABLE 4. Results of VGPFJSP scheduling with EGA, IJA, IGA, and GWO.

Instances	Size	EGA [39]	IJA [40]	IGA [41]	GWO [42]	Proposed
		C_{max}	C_{max}	C_{max}	C_{max}	C_{max}
MK01	10×6	74	88	71	66	64*
MK02	10×6	63	52	50	45	42*
MK03	15×8	226	226	226	226	202*
MK04	15×8	119	134	118	104	101*
MK05	15×4	297	302	293	276*	276*
MK06	10×15	163	191	123	109	108*
MK07	20×5	257	244	234	214	211*
MK08	20×10	803	873	803	794*	794*
MK09	20×10	534	611	520	460*	460*
MK10	20×15	443	533	400	348	346*

computation time is longer than other algorithms. In real factory scheduling, the speed issue can be regarded as a disadvantage that needs to be addressed to lower the computation time of GA-VNS.

After conducting the several trials with small-sized instances, Table 3, Table 4, and Table 5 show that GA-VNS only utilizes the local search algorithm when the number of jobs is less than 6, as VNS requires less computation time than GA or the proposed algorithm. Therefore, based on these results, we conclude that GA-VNS effectively solves the FJSP in the parallel machines with variable groups in large-scale problems. SMEs using the algorithm can schedule their entire production while saving an average of 26% in production time compared to other scheduling methods, as demonstrated by the GAP results in Table 3. Fig. 8 (a) and (b) depict the computational experiment results of finding the minimum makespan of GA, VNS, and GA-VNS depending on each population size. Each experiment was conducted by utilizing the VW06 and VW 34 datasets with a max VG size of 3 and 4. The results of EGA, IJA, IGA, GWO, and GA-VNS comparison is shown in Table 4. In the table, MK01:10 instances are in large-scale problems. While the GWO has few minimum makespan results, the GA-VNS outperforms other algorithms. Fig. 9 shows a boxplot of the deviation distribution of the algorithm results. Since the proposed algorithm maintains minimum makespan among other algorithms in the experiments, the proposed algorithm has a consistent deviation result of 0. In addition, to test the global searching and local searching abilities of GA and VNS, we evaluated each algorithm with GA-VNS, as shown in Table 5.

Utilizing parallel machines in FJSP demonstrates efficiency throughout the makespan. Although the GA-VNS has a significant difference in minimum makespan compared to the other algorithms, it takes much time to execute. Nevertheless, the new approach is expected to benefit SMEs in achieving optimal job shop scheduling and effectiveness. Fig. 10 shows the computational experiment results of finding

TABLE 5. Results of VGPFJSP scheduling with VW.

Instances	Size	VG_{max} settings	GA		VNS		Proposed		
			C_{max}	CPU(s)	C_{max}	CPU(s)	C_{max}	C_{max} with vg-parallel machines	CPU(s)
LFJS1	10×5	-	667*	2.44	678	1.85	667*	-	6.21
LFJS2	10×5	-	736*	2.40	777	1.84	736*	-	6.16
LFJS3	10×5	-	607*	2.28	635	1.75	607*	-	5.92
LFJS4	15×5	-	951*	5.47	951*	3.85	951*	-	12.76
LFJS5	15×5	-	958*	6.20	958*	4.32	958*	-	12.96
VW1	2×2	2	56*	0.07	56	0.05	56*	52	0.08
VW2	2×2	2	58*	0.07	58	0.05	58*	56	0.09
VW3	3×2	2	95*	0.16	95	0.11	95*	86	0.40
VW4	3×5	3	33*	0.07	33	0.06	33*	33	0.21
VW5	4×3	2	118	0.16	123	0.12	103*	101	0.42
VW6	6×5	6	102	0.34	92	0.23	91*	91	0.82
VW7	3×5	3	60	0.15	61	0.11	55*	55	0.46
VW8	10×6	10	55	0.78	56	0.56	50*	49	2.02
VW9	10×6	6	45	0.79	47	0.57	40*	38	2.13
VW10	15×8	5	250	3.35	254	2.44	239*	234	8.75
VW11	15×8	2	79	1.40	87	1.00	74*	72	3.65
VW12	15×4	3	1256	4.61	1322	3.74	1195*	1171	12.67
VW13	10×15	5	119	2.51	130	1.76	117*	114	6.46
VW14	20×5	4	217	2.69	218	2.02	203*	201	7.03
VW15	20×10	2	557	7.86	585	5.53	554*	546	19.39
VW16	20×10	3	440	7.24	445	5.24	409*	384	18.64
VW17	20×15	2	348	6.20	363	4.70	342*	335	17.01
VW18	10×5	3	86	1.41	92	1.00	80*	79	3.67
VW19	10×5	3	88	1.39	91	1.00	81*	80	3.67
VW20	10×5	3	89	1.41	89	1.00	81*	76	3.67
VW21	10×5	3	569	2.14	586	1.75	553*	545	5.92
VW22	10×5	4	591	2.91	651	2.23	598*	579	7.49
VW23	15×5	4	849	5.34	880	4.09	843*	826	13.53
VW24	15×5	5	820	4.85	863	3.76	806*	790	12.52
VW25	15×5	5	790*	5.12	867	3.97	790*	789	13.06
VW26	15×5	3	927	5.55	955	4.15	890*	885	13.76
VW27	15×5	4	871	5.42	907	4.16	860*	853	13.67
VW28	20×5	4	1120	9.58	1145	7.05	1116*	1088	23.27
VW29	20×5	4	996	8.28	1031	6.09	968*	952	20.08
VW30	20×5	5	1155	9.41	1130	6.93	1081*	1073	22.89
VW31	20×5	4	1159	9.42	1177	7.12	1117*	827	23.16
VW32	20×5	4	1148	9.63	1188	7.21	1120*	1117	23.56
VW33	10×10	6	830	3.71	919	3.07	796*	821	10.52
VW34	15×5	4	900	4.38	858	3.94	843*	843	12.28

the minimum makespan of GA, VNS, and GA-VNS with the VW34 dataset with a max VG size of 4.

V. CONCLUSION

In this study, we proposed a job shop scheduling engine for SMEs. We have devised a platform to apply the proposed a hybrid genetic algorithm and variable neighborhood search (GA-VNS) to solve the variable group parallel flexible job shop scheduling problem. Efficient and optimal job shop scheduling is essential in the manufacturing industry to reduce production cost and time. To address the real-world scheduling problems, we propose a new approach of FJSP on parallel machines with variable groups, which is a complex optimization problem that aims to process the same job on multiple parallel machines and obtain the minimum manufacturing period.

In this paper, we design and build a GA-VNS to solve FJSP on parallel machines with variable groups. Three experiments were conducted to select the optimal parameters of GA-VNS. To better validate the robustness of the proposed algorithm, we demonstrated that it is effective in minimizing the average makespan through multiple experiments. We used total of twenty-nine benchmark instances to test the efficiency of the proposed algorithm. Although the proposed algorithm outperforms other algorithms, most of the existing FJSP approaches does not consider the parallel machines with variable groups. Therefore, it is hard to compare with the existing approaches with the proposed algorithm. These experiments demonstrated that GA-VNS has excellent performance in solving FJSP on parallel machines with variable groups. In order to discover the suitable combination of parameters for GA-VNS, thirty-nine existing benchmark instances for traditional FJSP instances were modified. In the experiments, we compared the efficiency of global and local search algorithms and validated the effectiveness of a hybrid algorithm.

Overall, the main features of our platform are expected to bring economic and relational benefits to SMEs. In the future, we plan to extend the functionality of our flexible job shop scheduling platform by collecting various manufacturing scheduling cases. Future works are as follows: (1) a new start time variation algorithm is needed to consider the different start times of each operation, and (2) a large scheduling dataset collection is needed to extend the proposed algorithm to other domains.

REFERENCES

[1] M. Löfving, P. Almström, C. Jarebrant, B. Wadman, and M. Widfeldt, "Evaluation of flexible automation for small batch production," *Proc. Manuf.*, vol. 25, pp. 177–184, Jan. 2018, doi: 10.1016/j.promfg.2018.06.072.

[2] S. Mittal, M. A. Khan, and T. Wuest, "Smart manufacturing: Characteristics and technologies," *IFIP Adv. Inf. Commun. Technol.*, vol. 492, pp. 539–548, Mar. 2017, doi: 10.1007/978-3-319-54660-5_48.

[3] P. Weill, "The relationship between investment in information technology and firm performance: A study of the valve manufacturing sector," *Inf. Syst. Res.*, vol. 3, no. 4, pp. 307–333, 1992.

[4] M. G. Mehrabi, A. G. Ulsoy, and Y. Koren, "," *J. Intell. Manuf.*, vol. 11, no. 4, pp. 403–419, 2000, doi: 10.1023/A:1008930403506.

[5] A. Moeuf, R. Pellerin, S. Lamouri, S. Tamayo-Giraldo, and R. Barbaray, "The industrial management of SMEs in the era of industry 4.0," *Int. J. Prod. Res.*, vol. 56, no. 3, pp. 1118–1136, Feb. 2018, doi: 10.1080/00207543.2017.1372647.

[6] S. Mittal, M. A. Khan, D. Romero, and T. Wuest, "A critical review of smart manufacturing & Industry 4.0 maturity models: Implications for small and medium-sized enterprises (SMEs)," *J. Manuf. Syst.*, vol. 49, pp. 194–214, Oct. 2018, doi: 10.1016/j.jmsy.2018.10.005.

[7] B. S. de Ugarte, A. Artiba, and R. Pellerin, "Manufacturing execution system—A literature review," *Prod. Planning Control*, vol. 20, no. 6, pp. 525–539, Sep. 2009, doi: 10.1080/09537280902938613.

[8] L. Raymond and J. St-Pierre, "Customer dependency in manufacturing SMEs: Implications for R&D and performance," *J. Small Bus. Enterprise Develop.*, vol. 11, no. 1, pp. 23–33, Mar. 2004, doi: 10.1108/14626000410519074.

[9] D. Sampaio and J. Bernardino, "Open source CRM systems for SMEs," in *Proc. Int. C* Conf. Comput. Sci. Softw. Eng.*, New York, NY, USA, 2014, pp. 1–4.

[10] H. Reijonen and T. Laukkanen, "Customer relationship oriented marketing practices in SMEs," *Marketing Intell. Planning*, vol. 28, no. 2, pp. 115–136, Mar. 2010, doi: 10.1108/02634501011029646.

[11] N. Tzokas, Y. A. Kim, H. Akbar, and H. Al-Dajani, "Absorptive capacity and performance: The role of customer relationship and technological capabilities in high-tech SMEs," *Ind. Marketing Manage.*, vol. 47, pp. 134–142, May 2015, doi: 10.1016/j.indmarman.2015.02.033.

[12] A. Kouider and B. Bouzouia, "Multi-agent job shop scheduling system based on co-operative approach of idle time minimisation," *Int. J. Prod. Res.*, vol. 50, no. 2, pp. 409–424, Jan. 2012, doi: 10.1080/00207543.2010.539276.

[13] X. Shao and C. S. Kim, "Self-supervised long-short term memory network for solving complex job shop scheduling problem," *KSII Trans. Internet Inf. Syst.*, vol. 15, no. 8, pp. 2993–3010, Aug. 2021, doi: 10.3837/tiis.2021.08.016.

[14] I. A. Chaudhry and A. A. Khan, "A research survey: Review of flexible job shop scheduling techniques," *Int. Trans. Oper. Res.*, vol. 23, no. 3, pp. 551–591, May 2016, doi: 10.1111/itor.12199.

[15] X. Li and L. Gao, "An effective hybrid genetic algorithm and tabu search for flexible job shop scheduling problem," *Int. J. Prod. Econ.*, vol. 174, pp. 93–110, Apr. 2016, doi: 10.1016/j.ijpe.2016.01.016.

[16] H. Li, X. Wang, and J. Peng, "A hybrid differential evolution algorithm for flexible job shop scheduling with outsourcing operations and job priority constraints," *Expert Syst. Appl.*, vol. 201, Sep. 2022, Art. no. 117182, doi: 10.1016/j.eswa.2022.117182.

[17] Q. Luo, Q. Deng, G. Gong, X. Guo, and X. Liu, "A distributed flexible job shop scheduling problem considering worker arrangement using an improved memetic algorithm," *Expert Syst. Appl.*, vol. 207, Nov. 2022, Art. no. 117984, doi: 10.1016/j.eswa.2022.117984.

[18] P. Fattahi, F. Jolai, and J. Arkat, "Flexible job shop scheduling with overlapping in operations," *Appl. Math. Model.*, vol. 33, no. 7, pp. 3076–3087, Jul. 2009, doi: 10.1016/j.apm.2008.10.029.

[19] Z. Tian, X. Jiang, W. Liu, and Z. Li, "Dynamic energy-efficient scheduling of multi-variety and small batch flexible job-shop: A case study for the aerospace industry," *Comput. Ind. Eng.*, vol. 178, Apr. 2023, Art. no. 109111, doi: 10.1016/j.cie.2023.109111.

[20] C. Lin, Z. Cao, and M. Zhou, "Learning-based cuckoo search algorithm to schedule a flexible job shop with sequencing flexibility," *IEEE Trans. Cybern.*, early access, Nov. 10, 2022, doi: 10.1109/TCYB.2022.3210228.

[21] L. Zheng, X. Liu, F. Wu, and Z. Zhang, "A data-driven model assisted hybrid genetic algorithm for a two-dimensional shelf space allocation problem," *Swarm Evol. Comput.*, vol. 77, Mar. 2023, Art. no. 101251, doi: 10.1016/j.swevo.2023.101251.

[22] J. Fan, C. Zhang, W. Shen, and L. Gao, "A matheuristic for flexible job shop scheduling problem with lot-streaming and machine reconfigurations," *Int. J. Prod. Res.*, vol. 61, pp. 1–24, Nov. 2022, doi: 10.1080/00207543.2022.2135629.

[23] M. Yazdani, M. Amiri, and M. Zandieh, "Flexible job-shop scheduling with parallel variable neighborhood search algorithm," *Expert Syst. Appl.*, vol. 37, no. 1, pp. 678–687, Jan. 2010, doi: 10.1016/j.eswa.2009.06.007.

[24] P. Brucker and R. Schlie, "Job-shop scheduling with multi-purpose machines," *Computing*, vol. 45, no. 4, pp. 369–375, Dec. 1990, doi: 10.1007/BF02238804.

[25] F. Pezzella, G. Morganti, and G. Ciaschetti, "A genetic algorithm for the flexible job-shop scheduling problem," *Comput. Oper. Res.*, vol. 35, no. 10, pp. 3202–3212, Oct. 2008, doi: 10.1016/j.cor.2007.02.014.

[26] R. Chen, B. Yang, S. Li, and S. Wang, "A self-learning genetic algorithm based on reinforcement learning for flexible job-shop scheduling problem," *Comput. Ind. Eng.*, vol. 149, Nov. 2020, Art. no. 106778, doi: 10.1016/j.cie.2020.106778.

[27] M. R. Singh and S. S. Mahapatra, "A quantum behaved particle swarm optimization for flexible job shop scheduling," *Comput. Ind. Eng.*, vol. 93, pp. 36–44, Mar. 2016, doi: 10.1016/j.cie.2015.12.004.

[28] Y. Yuan, H. Xu, and J. Yang, "A hybrid harmony search algorithm for the flexible job shop scheduling problem," *Appl. Soft Comput.*, vol. 13, no. 7, pp. 3259–3272, Jul. 2013, doi: 10.1016/j.asoc.2013.02.013.

[29] A. Jamili, M. A. Shafia, and R. Tavakkoli-Moghaddam, "A hybrid algorithm based on particle swarm optimization and simulated annealing for a periodic job shop scheduling problem," *Int. J. Adv. Manuf. Technol.*, vol. 54, nos. 1–4, pp. 309–322, Apr. 2011, doi: 10.1007/s00170-010-2932-8.

[30] M. Asadpour, Z. Hodaei, M. Azami, E. Kehtari, and N. Vesal, "A green model for identical parallel machines scheduling problem considering tardy jobs and job splitting property," *Sustain. Oper. Comput.*, vol. 3, pp. 149–155, Jan. 2022, doi: 10.1016/j.susoc.2022.01.002.

[31] I. Cohen, K. Postek, and S. Shtern, "An adaptive robust optimization model for parallel machine scheduling," *Eur. J. Oper. Res.*, vol. 306, no. 1, pp. 83–104, Apr. 2023, doi: 10.1016/j.ejor.2022.07.018.

[32] K. Li, J. Chen, H. Fu, Z. Jia, and W. Fu, "Uniform parallel machine scheduling with fuzzy processing times under resource consumption constraint," *Appl. Soft Comput.*, vol. 82, Sep. 2019, Art. no. 105585, doi: 10.1016/j.asoc.2019.105585.

[33] S. Lin and K. Ying, "Uniform parallel-machine scheduling for minimizing total resource consumption with a bounded makespan," *IEEE Access*, vol. 5, pp. 15791–15799, 2017, doi: 10.1109/ACCESS.2017.2735538.

[34] W. Fang, H. Zhu, and Y. Mei, "Hybrid meta-heuristics for the unrelated parallel machine scheduling problem with setup times," *Knowl.-Based Syst.*, vol. 241, Apr. 2022, Art. no. 108193, doi: 10.1016/j.knosys.2022.108193.

[35] M. Moser, N. Musliu, A. Schaerf, and F. Winter, "Exact and metaheuristic approaches for unrelated parallel machine scheduling," *J. Scheduling*, vol. 25, no. 5, pp. 507–534, Oct. 2022, doi: 10.1007/s10951-021-00714-6.

[36] S. Maecker, L. Shen, and L. Mönch, "Unrelated parallel machine scheduling with eligibility constraints and delivery times to minimize total weighted tardiness," *Comput. Oper. Res.*, vol. 149, Jan. 2023, Art. no. 105999, doi: 10.1016/j.cor.2022.105999.

[37] P. Fattahi, M. S. Mehrabad, and F. Jolai, "Mathematical modeling and heuristic approaches to flexible job shop scheduling problems," *J. Intell. Manuf.*, vol. 18, no. 3, pp. 331–342, Jul. 2007, doi: 10.1007/s10845-007-0026-8.

[38] P. Brandimarte, "Routing and scheduling in a flexible job shop by Tabu search," *Ann. Oper. Res.*, vol. 41, no. 3, pp. 157–183, Sep. 1993, doi: 10.1007/BF02023073.

[39] M. Dai, D. Tang, A. Giret, and M. A. Salido, "Multi-objective optimization for energy-efficient flexible job shop scheduling problem with transportation constraints," *Robot. Comput.-Integr. Manuf.*, vol. 59, pp. 143–157, Oct. 2019, doi: 10.1016/j.rcim.2019.04.006.

[40] J.-Q. Li, J.-W. Deng, C.-Y. Li, Y.-Y. Han, J. Tian, B. Zhang, and C.-G. Wang, "An improved Jaya algorithm for solving the flexible job shop scheduling problem with transportation and setup times," *Knowl.-Based Syst.*, vol. 200, Jul. 2020, Art. no. 106032, doi: 10.1016/j.knosys.2020.106032.

[41] G. Zhang, Y. Hu, J. Sun, and W. Zhang, "An improved genetic algorithm for the flexible job shop scheduling problem with multiple time constraints," *Swarm Evol. Comput.*, vol. 54, May 2020, Art. no. 100664, doi: 10.1016/j.swevo.2020.100664.

[42] M. Pal, M. L. Mittal, G. Soni, S. S. Chouhan, and M. Kumar, "A multi-agent system for FJSP with setup and transportation times," *Expert Syst. Appl.*, vol. 216, Apr. 2023, Art. no. 119474, doi: 10.1016/j.eswa.2022.119474.

YEONJEE CHOI was born in Busan, South Korea, in 1994. She received the B.S. degree in gender studies from the University of California, Los Angeles, Los Angeles, CA, USA, in 2016. She is currently pursuing the M.S. degree in information systems from Pukyong National University.

From 2019 to 2021, she was an Assistant Research Engineer with the Technical Research Center, New Communication InfoTech Company Ltd. Her research interests include the development of monitoring systems, smart grids, anomaly detection, and deep learning.

HYUN SUK HWANG was born in Busan, South Korea, in 1968. She received the Ph.D. degree in management information system from Pukyong National University, Busan, in 2001.

From 2017 to 2021, she was the Technical Director of the Technical Research Center, New Communication InfoTech Company Ltd. Her research interests include the development of monitoring system in system construction, anomaly detection, and deep learning.

CHANG SOO KIM received the Ph.D. degree from the Department of Computer Engineering, Chung-Ang University, Seoul, South Korea, in 1991. He has been a Professor with the Department of IT Convergence and Application Engineering, Pukyong National University, Busan, South Korea, since 1992. He has been the Vice President of the Korea Multimedia Society, since 2011. His research interests include scheduling of smart factories and big data simulation.