Niching Genetic Programming With Elite Archive for Multi-Objective Dynamic Scheduling

Yong Wang, Jie Zhang, Fangfang Zhang, *Member, IEEE*, Gai-Ge Wang, *Senior Member, IEEE*, and Mengjie Zhang, *Fellow, IEEE*

***Abstract*—Multi-objective dynamic flexible job shop scheduling (MO-DFJSS) is increasingly recognized as a critical challenge in manufacturing and production systems due to the dynamic and complex nature of real-world scenarios. Traditional approaches often fail to adequately address the dynamic and conflicting objectives inherent in these systems. Genetic Programming (GP) as a hyper-heuristic method has shown promise in DFJSS; however, the studies of GP for MO-DFJSS are still with limitations. The current NSGP-II algorithm incorporates the Non-dominated Sorting Algorithm II (NSGA-II) into GP. However, it does not utilize the potential information contained in the Pareto fronts obtained during evolutionary process to help population evolve. However, the Pareto fronts generated in each generation may contain valuable information that can help the population evolve more effectively. For example, the distribution of non-dominated individuals evolves over generations may help for finding promising solutions. This paper proposes an elite individual guided dynamic space optimization strategy with an archive. This strategy treats the non-dominated individuals in each generation as elite individuals and saves them in an archive, and then uses information about the distribution of individuals in the archive to dynamically guide the GP search. This allows the GP population of each generation to explore in a more promising space to improve the performance. Meanwhile, to improve the interpretability of the learned rules in the Pareto fronts, we have also proposed a novel niching strategy to impact the tree size of the learned scheduling rules during evolutionary process. The results show that the proposed algorithm exhibits superior performance in solving MO-DFJSS. The proposed algorithm learns diverse and well distributed Pareto fronts, and also improves the interpretability of the Pareto fronts generated. These features position our approach within emerging directions in computational intelligence by introducing a behavior-driven niching mechanism that enables self-adaptive diversity control and dynamic knowledge retention in multi-objective evolutionary learning.**

***Index Terms*—Dynamic flexible job-shop scheduling, genetic programming, multi-objective, niching, archive.**

I. Introduction

JOB shop scheduling (JSS) is an important combinatorial optimization problem that involves many real-world application scenarios, such as order picking in warehouses [1], manufacturing process design [2], grid/cloud management [3]. The main goal of JSS is to determine the order of processing jobs by a set of machines effectively. Dynamic job shop scheduling (DJSS) extends from JSS by incorporating dynamic events, such as the arrival of jobs over time [4]. Dynamic flexible job shop scheduling (DFJSS) [5] further considers the flexibility of machine resources, i.e., each operation can be executed on multiple machines. DFJSS faces two main decisions: machine allocation for assigning each operaiton to a specific machine, and operation sequencing for choosing the next operation to be processed when a machine is idle [6].

In real industrial applications, scheduling problems often involve the optimization of multiple conflicting objectives [7], [8]. So, MO-DFJSS has attracted a lot of attention and many researches have explored various methods to solve this problem. Genetic Programming (GP) is an evolutionary algorithm that automatically evolves computer programs, typically represented as tree structures, by simulating the process of natural selection [9]. It applies genetic operators such as crossover, and mutation to evolve a population of symbolic expressions that can serve as candidate solutions. As a hyper-heuristic method, has been widely used to automatically evolve scheduling heuristics for solving DFJSS problem [10], [11], [12], [13] due to its flexible representation and good performance. Compared with traditional machine learning methods, GP has several distinctive advantages for solving dynamic scheduling problems. First, GP can automatically evolve effective dispatching rules without requiring extensive domain knowledge. Second, the evolved rules can efficiently make real-time decisions. Third, GP generates interpretable heuristic rules in the form of symbolic expressions (tree structures), which enhances transparency and facilitates practical deployment. However, most of the research on GP for DFJSS focuses on single objective [12], [14], [15], and its application in solving MO-DFJSS needs to be further explored. Therefore, it is valuable to investigate how GP can be further improved to solve MO-DFJSS.

To the best of our knowledge, several GP related studies [16], [17] have integrated well-konwn Pareto dominance-based methods [18], [19] and decomposition-based methods [20] into GP. Among them, NSGP-II [16] demonstrated the best performance.

Received 26 August 2025; accepted 12 October 2025. This work was supported by the National Natural Science Foundation of China under Grant 62473350 and in part by the Fundamental Research Funds for the Central Universities. *(Corresponding author: Gai-Ge Wang.)*

Yong Wang, Jie Zhang, and Gai-Ge Wang are with the School of Computer Science and Technology, Ocean University of China, Qingdao 266100, China (e-mail: wangyong@ouc.edu.cn; zhangjie1198@stu.ouc.edu.cn; wgg@ouc.edu.cn).

Fangfang Zhang and Mengjie Zhang are with the Centre for Data Science and Artificial Intelligence & School of Engineering and Computer Science, Victoria University of Wellington, Wellington 6140, New Zealand (e-mail: fangfang.zhang@ecs.vuw.ac.nz; mengjie.zhang@ecs.vuw.ac.nz).

Recommended for acceptance by G. Cabrera-Guerrero.

Digital Object Identifier 10.1109/TETCI.2025.3623315

However, each generation of NSGP-II evolves in the entire heuristic space and fails to make more efficient use of the information of Pareto fronts collected from previous evolutionary generations to guide the search. We note that the Pareto fronts obtained in each generation may contain valuable information about the direction and trend of the optimization. For example, observing the distribution of the non-dominated individuals in each generation can reveal trends in the optimization process. Observing the distribution can also provide clues about which regions are gradually improving and which regions need to be further explored. By utilizing this information, it is expected to guide the evolutionary process more intelligently and improve the overall effectiveness and efficiency of our algorithm. Therefore, we propose a method that dynamically adjusts the search space based on the Pareto fronts information from each generation. Meanwhile, in order to facilitate the calculation and analysis of this information, we borrow phenotypic characterization (PC) [21] to represent the behavior of individuals in this strategy, and calculate the information presented by observing the behavioral distribution of the population.

The overall goal of this paper is to improve the effectiveness of NSGP for MO-DFJSS. Meanwhile, this paper focuses on improving the interpretability of learned scheduling heuristics obtained in the Pareto fronts. The contributions of this paper are as follows.

1) We propose an elite individual guided dynamic space optimization strategy designed to enhance the quality of scheduling heuristics. The strategy uses an elite archive to store the elite individuals, i.e., the non-dominated individuals at each generation and proposes a reward and punishment strategy to update the archive, then dynamically guide the search of NSGP according to the distribution of the elite individuals in the archive. Specifically, this method can dynamically narrow the search space by analyzing the distribution of the individuals within the archive and offers a search method that focuses more on promising areas to effectively exploring complex multi-objective search space. The results show that this strategy is effective in improving the quality of Pareto fronts and maintaining the diversity of the population.
2) We propose a novel niching strategy based on PC and group selection. The strategy performs niching based on the behavior of the individuals and then influences the interpretability of the Pareto fronts by selecting the individuals with the smallest tree size among those that have survived from the group selection phase. The results show that this strategy has a positive impact on controlling the rule size of Pareto fronts and also helps to improve the diversity of the population. In addition, the strategy effectively controls rule size, contributing to a more interpretable solutions, which is valuable in the context of optimization using GP.
3) The experimental results demonstrate that our proposed algorithm outperforms existing methods in terms of scheduling heuristic quality and algorithmic efficiency. The comparative analysis reveals that our approach can find more robust and well-distributed scheduling rules, underscoring its potential to influence future research directions in the application of GP to complex scheduling problems.

The remainder of this paper is structured as follows: Section II provides the background. Section III details the proposed algorithm. Section IV outlines the experimental design. Section V presents the results and analysis. Further discussion is described in Section VI, followed by conclusions in Section VII.

II. Background

A. Multi-Objective Dynamic Flexible Job Shop Scheduling

In JSS, a set of jobs $\mathcal{J} = \{J_1, J_2, \ldots, J_n\}$ needs to be processed by a set of machines $\mathcal{M} = \{M_1, M_2, \ldots, M_n\}$. Each job J_i is represented by its arrival time r_i, due date d_i, weight w_i, and a sequence of operations $[O_{i,1}, O_{i,2}, \ldots, O_{i,p_i}]$ that must be processed one by one [10]. Each operation $O_{i,j}$ can be performed on several machines (i.e., flexible machine resources) [22]. The processing time of operation $O_{i,j}$ on machine M_k is $t_{i,j,k} = \pi_{i,j}/\gamma_k$, where γ_k is the processing speed of the machine and $\pi_{i,j}$ is the workload of the machine. This paper focuses on one dynamic event, the dynamic random arrival of a new job [4], [23], [24], since this is the most common dynamic event in life [16]. Until the new job actually arrives on the shop floor, its information is unknown. The main constraints of the JSS problems are as follows:

1) An operation cannot start until its preceding operations have been completed.
2) Each machine can process only one job at a time.
3) Each operation can be processed by only one of its optional machines.
4) Scheduling is non-preemptive, meaning that once an operation's processing starts, it must be completed without interruption.

In multi-objective scheduling problems, it needs to optimize multiple conflicting objectives simultaneously. In this paper, we consider seven scheduling objectives [25] for MO-DFJSS: max-flowtime ($Fmax$), mean-flowtime ($Fmean$), max-weighted-flowtime ($WFmax$), mean-weighted-flowtime ($WFmean$), max-tardiness ($Tmax$), max-weighted-tardiness ($WTmax$) and mean-weighted-tardiness ($WTmean$). The selected objective functions are designed to reflect both system-centric and customer-centric performance measures. Flowtime-based metrics, such as max-flowtime and mean-flowtime, quantify system utilization and responsiveness, while tardiness-based metrics, such as max-tardiness and mean-weighted-tardiness, emphasize delivery reliability. Including both weighted and unweighted versions ensures that job priorities are also considered. These objectives have been validated and adopted in numerous previous works on dynamic job shop scheduling [10], [11], [12], [13], [14], [15], forming a solid basis for evaluating algorithm performance under realistic multi-objective conditions. The definitions of these objectives are as follows:

1) $Fmax = \max_{i=1}^{n}\{C_i - r_i\}$
2) $Fmean = \frac{1}{n}\sum_{i=1}^{n}\{C_i - r_i\}$
3) $WFmax = \max_{i=1}^{n}\{w_i(C_i - r_i)\}$
4) $WFmean = \frac{1}{n}\sum_{i=1}^{n}\{w_i(C_i - r_i)\}$
5) $Tmax = \max_{i=1}^{n}\{T_i\}$

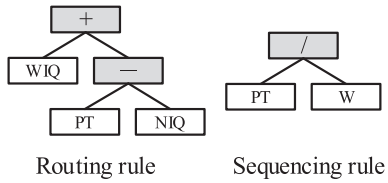


Fig. 1. An example of one individual program for DFJSS.

6) $WTmax = \max_{i=1}^{n}\{w_i T_i\}$
7) $WTmean = \frac{1}{n}\sum_{i=1}^{n}\{w_i T_i\}$

where C_i is the completion time of the job J_i, w_i is the weight of the job J_i, and $T_i = \max\{C_i - d_i, 0\}$ is the tardiness of the job J_i.

All objectives are formulated as minimization objectives, as smaller values of flowtime and tardiness related objectives indicate better scheduling performance.

B. Individual Program Representation

In DFJSS, the decision-making process consists of two core tasks: machine allocation and operation sequencing. The routing rule and sequencing rule have shown effectiveness in generating scheduling rules [6]. In order to explore and learn these two rules simultaneously, the use of a multi-tree representation of GP has been shown to be an effective strategy [26]. Therefore, we use the representation of GP with two trees to express routing and sequencing rules respectively in this paper.

Specifically, Fig. 1 gives an example of an individual in GP to represent the routing and sequencing rule for DFJSS [27]. In particular, the routing rule prioritizes machines based on "WIQ+PT-NIQ", where NIQ represents the number of operations in the queue, WIQ is the workload in the queue of a machine, and PT denotes the processing time of an operation on a specified machine. The sequencing rule assigns a priority value to each ready operation according to "PT/W", where W is the weight of an operation.

C. Related Work

1) Archive for scheduling problem: The archive is a commonly used technique in genetic algorithms, often employed to store non-dominated solutions found during the evolutionary process. Based on MOEA/D, an archive was used to save non-dominated individuals in each subpopulation. Then, computational resources were efficiently allocated by analyzing the contribution of each subpopulation for archive, thereby enhancing the performance of the algorithm [28]. A two-archive strategy that maintains both convergence-oriented and diversity-oriented archives was proposed to balance convergence and diversity simultaneously in constrained multi-objective optimization [29]. A two-stage multi-objective GP approach with archive was introduced to solve the uncertain capacitated arc routing problem [30]. It utilized an external archive to store potentially effective individuals that may have been lost during evolution and reuses them to produce offspring. Similarly, a study that used an archive to retain good individuals during the evolutionary process, involving these individuals in generating new populations was proposed for solving the DFJSS [31]. An archive-based co-evolutionary GP approach that leveraged an archive population to improve the quality of fitness evaluation was proposed for workflow scheduling [32]. Its core idea is also to use the archive to store high-quality individuals found in the population and to continually update the archive using sub-populations to improve its quality. However, the above studies still just use the archive to store quality individuals, without further analyzing the potential information about the individuals stored within the archive, to guide the search for performance improvement of algorithms.

2) Studies for multi-objective scheduling problem: A Q-Learning-based NSGA-II algorithm was introduced in [33] that daptively selects appropriate neighborhood structures for local search by learning from historical search experience for solving MO-DFJSS with multiple dynamic events and limited transportation resources. Two well-known multi-objective optimization frameworks, the NSGA-II [18] and SPEA2 [19] were combined with GP in [16] to address MO-DFJSS, named NSGP-II and SPGP2. Additionally, the multi-objective algorithm based on decomposition was integrated into GP and named MOGP/D in [17]. Among NSGP-II, SPGP2, and MOGP/D, NSGP-II demonstrates superior performance. Furthermore, Semantic Genetic Programming was incorporated into GP to enhance the algorithm's performance [10]. An algorithm that integrated the surrogate technique and brood recombination technique was proposed in [34]. The influence of terminal settings on NSGP-II for solving MO-DFJSS was studied in [35]. There are several other studies focused on topics such as the interpretability [36] or multitask in MO-DFJSS [37].

Overall, it is evident that the application of GP in researching MO-DFJSS is still relatively limited. The study of this field is currently in its infancy, with much space for exploration and improvement. Most of the existing studies focus on specific aspects or combinations of techniques, but there is a lack of comprehensive and in-depth research that fully exploits the potential of GP in solving MO-DFJSS. Our study aims to fill this gap by proposing a novel approach that addresses the existing limitations. By introducing innovative strategies and techniques, we strive to enhance the performance and effectiveness of GP in dealing with MO-DFJSS, thereby making a significant contribution to the advancement of this research area and providing more efficient solutions for real-world multi-objective scheduling problems in dynamic flexible job shop environments.

III. Niching GP With Elite Archive

A. Overall Framework

Fig. 2 illustrates the overall framework of the proposed algorithm. The process begins with the initialisation of a population. Then the population is evaluated, followed by a non-dominated sorting based on their performance in simulations. If the stopping criteria is met, the Pareto fronts are output; if not, the population proceeds through the evolutionary process. During this process, the elite archive must first be updated with the latest population evaluation information. Details of the update method of the elite archive will be given in Section III-B. After that, the population performs niching process according to the proposed niching

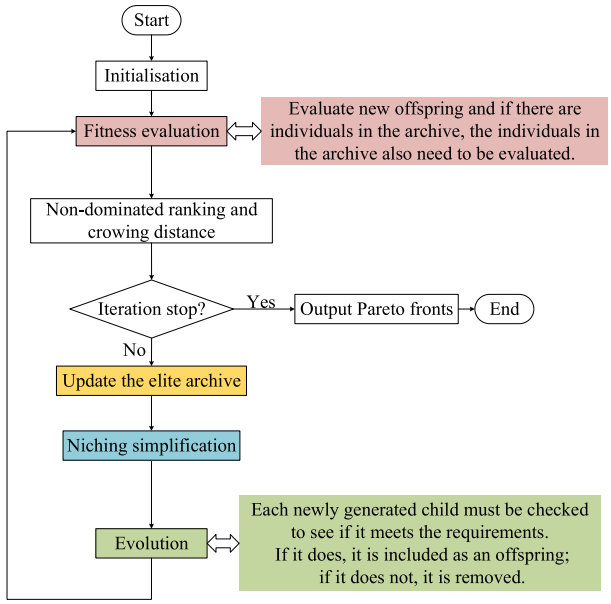


Fig. 2. The flowchart of the proposed algorithm.

strategy and the detail will be described in Section III-C. Then, the population generates offspring based on the elite individual guided dynamic space optimization strategy and the details will be discussed in Section III-D.

B. Elite Archive Strategy

In this paper, we treat the non-dominated individuals as elite individuals. So the external elite archive is used to store the non-dominated individuals, i.e., the rank of individuals is equal to 0, after the completion of each generation of non-dominated sorting. Specifically, at the end of each generation, the elite individuals are added to the archive if it is not a duplicate of any individual in the archive. The criterion for determining if two individuals are duplicates are to check whether the PCs of the two individuals are same. The PC of an individual is a vector constituted the rank of machines or operations, reflecting its decision-making behavior across various scenarios. When two individuals have similar PC, their behavior and effectiveness are likely to be alike [34]. Table I provides an illustrative example of calculating the PC for an individual. For the first decision situation, machines are ranked first by the reference routing rule WIQ (i.e., least work in the queue). Then the routing rule is also used to rank the three machines, with M_3 having the first priority. Finally, set the value of PC_1 as the rank of M_3 ranked by WIQ (i.e., 3). Similarly, PC_2 is 1 and PC_3 is 3. The PC of sequencing rule can be derived in the same way. Finally, the routing and sequencing rules' PCs are combined to form the individual's PC. An example of this combined PC, represented as a 6-D vector (with three routing and three sequencing decision situations), is shown in Fig. 3.

As previously noted, PC is a vector used to indicate the behavior of an individual. Following the guidance in [6], this paper randomly selects 20 routing decision situations and 20 sequencing decision situations, each with 7 candidates (machines or operations). Consequently, the PC of an individual is represented as a 40-D vector. Based on the PC, we use the equation (1) to calculate the distance between two PCs, i.e., **p1**

Algorithm 1: Updating Process of the Archive.

Input: The current population pop, the external elite $archive$
Output: The updated $archive$
for $ind \in archive$ **do**
  if $ind.rank \neq 0$ **then**
    $ind.counter++$;
    if $ind.counter \geq 3$ **then**
      Remove the ind from the $archive$;
    end if
  else
    $ind.counter--$;
  end if
end for
for $ind \in pop$ **do**
  if $ind.rank = 0$ **then**
    Set $duplicate = false$;
    for $ind' \in archive$ **do**
      Calculate $distance$ between ind and ind';
      if $distance = 0$ **then**
        $duplicate = true$;
        break;
      end if
    end for
    if $duplicate = false$ **then**
      Add ind to the $archive$;
    end if
  end if
end for
return $archive$;

TABLE I
AN EXAMPLE OF CALCULATING THE PHENOTYPIC CHARACTERISATION OF A ROUTING RULE

Decision Situations	WIQ	Routing Rule	PC_i
$1(M_1)$	1	3	
$1(M_2)$	2	2	3
$1(M_3)$	**3**	1	
$2(M_1)$	2	2	
$2(M_2)$	3	3	1
$2(M_3)$	**1**	1	
$3(M_1)$	2	2	
$3(M_2)$	1	3	3
$3(M_3)$	**3**	1	

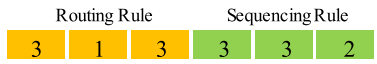


Fig. 3. Example of the phenotypic characterization of a GP individual which is a 6-D vector.

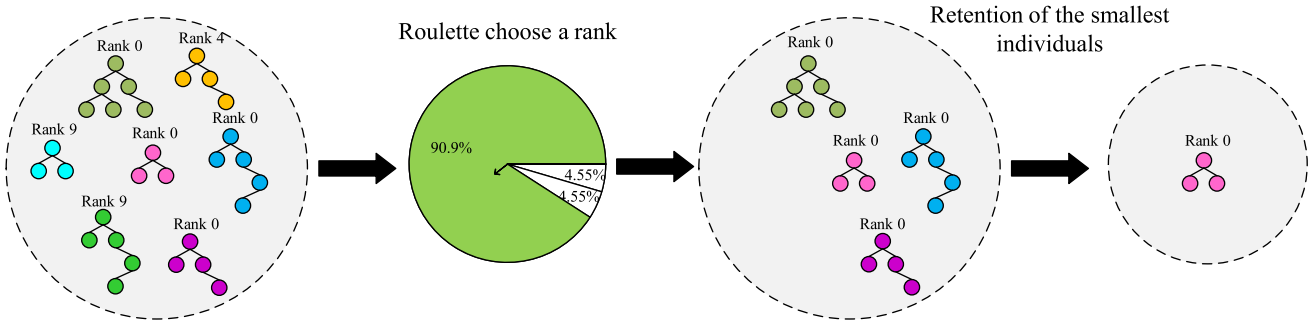


Fig. 4. Specific process of the niching strategy based on PC and group selection.

and **p2**:

$$\mathrm{dis}(\mathbf{p1}, \mathbf{p2}) = \sum_{i=1}^{40} d_{p_{1i}, p_{2i}} \tag{1}$$

where $d_{p_{1i},p_{2i}} = 1$ if $p_{1i} = p_{2i}$, and $d_{p_{1i},p_{2i}} = 0$, otherwise. In the following, we refer to the PCs distance between two individuals as the distance between two individuals. If the distance of two individuals is 0, we treat the two individuals to be duplicate, and will not add the individual into the archive. During the evolution, there will be individuals that perform better in one or two specific instances and worse in the rest of the instances occur in GP where seed rotation is applied, and we define such individuals as mutant individuals. To ensure the elitism of the individuals that are stored within the archive and promptly remove mutant individuals, we propose a reward and punishment strategy for updating the archive. Each individual, upon being added to the archive, is assigned a private counter, which records this times it was punished. After each evaluation, the rank of each individual is checked. If the rank is greater than 0, the counter is added 1; if the value of the counter exceeds 3, remove the individual from the archive; if not, subtract 1 from the counter. The specific archive updating process is shown in Algorithm 1.

C. Niching Strategy Based on PC and Group Selection

The process of the niching strategy is showed in Fig. 4, operates as follows: Firstly, individuals in the population are grouped based on their PCs, with individuals having the same PC forming one group. Each group then chose an individual through probability-weighted selection. Individuals whose rank match the selected individual's rank are retained. To ensure that high-quality individuals have higher probabilities of being selected and thus provide superior genes for the breeding process, the probability-weighted selection assigns greater weights to individuals with smaller rank. The specific algorithm is illustrated in Algorithm 2. Finally, the size of an individual directly impacts the interpretability of the scheduling heuristic. The more nodes there are, the less interpretable the individual becomes. Therefore, it is crucial to control the size of individuals during evolutionary process to ensure Pareto fronts have higher interpretability. Hence, in the final step, we select the individual with the smallest tree size among the retained individuals to represent each group.

Algorithm 2: Probability-Weighted Selection.

Input: The selected group $group$
Output: The representative individual $selectedIndividual$

```
double totalWeight = 0;
for ind ∈ group do
    totalWeight = totalWeight + 1/(ind.rank + 1);
end for
Get a randomValue between [0, totalWeight);
double p = 0;
for ind ∈ group do
    p = p + 1/(ind.rank + 1);
    if randomValue ≤ p then
        selectedIndividual = ind;
        break;
    end if
end for
return selectedIndividual;
```

D. Elite Individual Guided Dynamic Space Optimization Strategy

In GP, the conventional approach often searches the entire heuristic space for optimal solutions. This approach can sometimes lead to inefficiencies and fail to find high-quality solutions. Therefore, we propose a novel strategy that concentrates the search space around the elite individuals found (i.e, the individuals within archive), thus improving the effectiveness of the evolutionary process. The specific process of the elite individual guided dynamic space optimization strategy is shown in Fig. 5. The core idea is to refine the search space dynamically according to the PCs distance between the individuals in the elite archive and newly generated offspring. Specifically, when an offspring ind_o is generated, we first calculate the $distance$ between it and all the individuals in the archive, and get the $minDistance$. Then check if the $minDistance$ satisfies $0 < minDistance < \alpha$. For the first 5 [38] generations, $\alpha = 40$, i.e., the maximum PCs distance is taken. Then, in order to adaptively and dynamically adjust the search space according to the latest archive information for each generation, the value of α is calculated from the information provided by the archive. Specifically, we compute the $distance$ between each individual in the archive and the others to get a list of distances, and

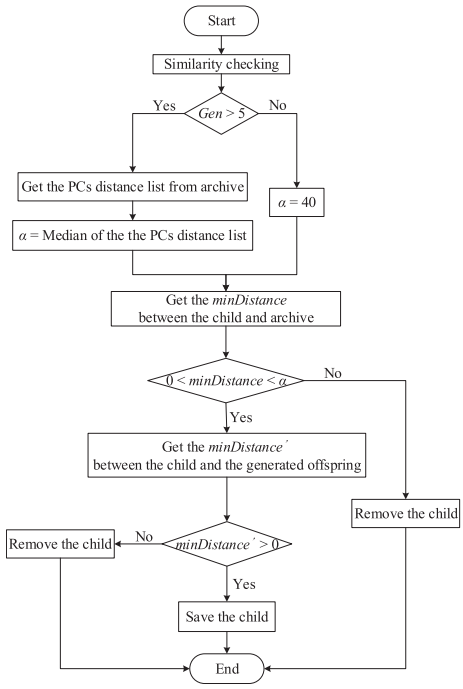


Fig. 5. Specific process of elite individual guided dynamic space optimization strategy.

then sorted the list from smallest to largest. Finally, α takes its median. The median is chosen because it provides a robust central tendency measure, minimizing the influence of outliers or extreme values in the distribution of distances. This ensures that the self-adaptive distance reflects the typical distribution of the archive, maintaining a balanced search space that is neither overly restricted nor too expansive. By dynamically recalculating α as the median at each generation, the search process can adapt to the evolving characteristics of the archive while preserving stability against noisy or skewed data. In the following, we call the distance obtained by archive calculation for each generation the self-adaptive distance.

The reason for not using self-adaptive distance to control the generation of offspring at the first 5 generations is to prevent rapid convergence of the population. During the early stages of the evolutionary process, where most individuals are randomly generated, the population is highly diverse. In this case, a few non-dominated individuals are likely to dominate most others in most cases [39]. This means that the information provided by individuals in the archive has a greater degree of randomness built into it and is not yet reliable enough. Therefore, we propose to set $\alpha = 40$ in the first 5 generations, and then set α to the self-adaptive distance after 5 generations, i.e., the stage that most individuals in the archive are good enough and can specialize in different situations.

Meanwhile, in order to avoid too fast convergence of offspring under the guidance of this strategy, which leads to a decline in the diversity of the population. It is also necessary to compare the similarity between the newly generated individual and the previously generated offspring. An individual will be added to the offspring only if the minimum distance between the current individual and the generated offspring is greater than zero [10].

IV. Experiment Design

A. Simulation Model

In reference to widely used DFJSS instances [16], [40], [41], the scenario involves processing 5000 jobs across ten machines. Jobs arrive according to a Poisson process with a rate of λ. Each job consists of a random number of operations, uniformly distributed between 1 and 10. Each operation is assigned to a random number of candidate machines, also uniformly distributed between 1 and 10. Processing times are drawn from a uniform discrete distribution ranging from 1 to 99. The due date for each job is set to 1.5 times its processing time. Job importance (weights) follows the distribution: 1 (20%), 2 (60%), and 4 (20%) [21]. Each training instance is generated uniquely by assigning a new random seed for the DFJSS simulation [42]. A utilization-level parameter p is employed to simulate job-shop scenarios with varying characteristics [43]. This parameter represents the proportion of time a machine is expected to be busy, adjusted by λ in the Poisson process. It is calculated using equation (2), where μ denotes the average processing time and P_M is the probability of a machine being visited by a job. For instance, P_M is $2/10$ if each job has two operations. A higher value of the utilization level indicates a busier job shop.

$$\lambda = \mu * P_M/p. \tag{2}$$

The initial 1000 jobs are designated as warm-up jobs and are excluded from the objective calculations to establish steady-state performance. Fitness values are calculated based on the subsequent 5000 jobs. The simulation ends upon the completion of the 6000th job.

B. Design of Comparisons

In this study, we conduct three sets of comparison experiments to evaluate the performance of our proposed algorithm. In the following part, the algorithm that uses niching strategy is named NSGP-II-N, the algorithm that uses self-adaptive phenotypic characterization similarity strategy is named NSGP-II-A, and the algorithm that uses both strategies is named NSGP-II-NA. Six pairs of conflicting objectives (Fmax and WTmax, Tmax and WFmax, Fmean and WTmean, Fmax and Fmean, Fmax and WFmean, and Fmean and WFmean) [10], [16] are formed as multi-objective scenarios along with the utilization level (i.e., 0.85 and 0.95) to verify the effectiveness of the algorithms.

1) *Comparison of the overall performance:* In order to evaluate the overall performance of the proposed algorithm on solving MO-DFJSS, NSGP-II and NSGP-II-N are compared with NSGP-II-NA.
2) *Comparison with manual rules:* To further demonstrate the effectiveness of the proposed algorithm, NSGP-II-NA is compared with eight commonly used, manually designed scheduling rules based on four sequencing and two routing heuristics [44].

TABLE II
THE GP TERMINAL AND FUNCTION SET FOR DFJSS

Notation	Description
NIQ	The number of operations in the queue
WIQ	The workload in the queue of a machine
MWT	Waiting time of a machine
PT	Processing time of an operation on a specified machine
NPT	Median processing time for the next operation
OWT	The waiting time of an operation
WKR	Median amount of work remaining for a job
NOR	The number of operations remaining for a job
W	Weight of a job
TIS	Waiting time in system for a job
Function	$+, -, \times, /$, max, min

TABLE III
THE PARAMETER SETTINGS OF GP

Notation	Description
Population size	1000
Number of generations	51
Method for initialising population	ramped-half-and-half
Initial minimum/maximum depth	2 / 6
Maximal depth	8
Crossover rate	0.85
Mutation rate	0.15
Terminal/non-terminal selection rate	10% / 90%
Parent selection	Tournament selection

3) *Performance of the elite individual guided dynamic space optimization strategy:* To demonstrate the effectiveness of the proposed elite individual guided dynamic space optimization strategy in improving the effectiveness of the algorithm, we compare the experimental results of NSGP-II-N and NSGP-II-NA.
4) *Performance of the niching strategy:* The second set of experiments aims to validate the effectiveness of the niching strategy in controlling rule size of Pareto fronts. So, we compute the average run size of Pareto fronts obtained by NSGP-II-A and NSGP-NA to verify the effectiveness of the strategy.

C. Parameter Settings

In the experiments, the terminal and function sets of GP are shown in Table II. The terminal set consists of the features related to machines (e.g., NIQ, WIQ, and MWT), operations (e.g., PT, NPT, and OWT), jobs (e.g., WKR, NOR, W, and TIS). In the function set, the arithmetic operators take two arguments. The "/" operator is protected and returns 1 if divided by zero. The functions "max" and "min" take two arguments and return the maximum and minimum of their arguments, respectively [45]. The GP parameter settings are shown in Table III [10].

V. RESULTS AND DISCUSSION

For every scenario, we conduct 30 independent runs and apply Friedman's test and Wilcoxon ranksum test with a significance level of 0.05 to determine statistical significance. The evaluation metrics used include hypervolume (HV) [46] and inverted generational distance (IGD) [47]. A higher HV value and a lower IGD value signify better performance. Since the actual Pareto fronts are not available for our problem, we construct an approximation by combining the best solutions from all algorithms, which we then use it to compute the IGD.

A. Quality of Learned Scheduling Heuristics

Table IV presents the mean and standard deviation of HV and IGD values and Friedman's test result for both training and test instances. The symbols "↑ ", "↓ ", and "≈" indicate that the two algorithms being compared are statistically better, worse or similar. An algorithm is compared with all algorithms at its left one by one. "Win, Draw, Lose" is the number of scenarios that the proposed algorithm is better, similar, or worse than NSGP-II [37].

According to Table IV, the overall results show that NGSP-II-NA outperforms NSGP-II and NSGP-II-N, exhibiting higher HV values and lower IGD values. This means that NSGP-II-NA is capable of producing better Pareto fronts and offering more diverse solutions in most scenarios. From the comparison results of NSGP-II and NSGP-II-N, it can be seen that in terms of HV and IGD values, the performance of the two is similar. However, according to the results of Friedman's test, NSGP-II-N outperforms NSGP-II in eight out of twelve scenarios, which suggests that the niching strategy can have a positive impact on the performance of the algorithm. In addition, the results of Friedman's test and Wilcoxon ranksum test of NSGP-II-NA are both better than NSGP-II-N. This demonstrates the the superiority of the elite individual guided dynamic space optimization strategy in enhancing the performance of the algorithm.

Based on the results presented in Table IV, it is evident that NSGP-II-N performs similarly to NSGP-II in enhancing algorithmic performance. Therefore, when plotting the violin plots to analyze the distribution of solutions, we focused solely on comparing NSGP-II and NSGP-II-NA. Fig. 6 is a violin plot of the HV values of NSGP-II and NSGP-II-NA for the last generation of Pareto fronts during the test process in twelve scenarios. Fig. 6 shows that the values of HV of Pareto fronts obtained by NSGP-II-NA are overall larger than that of NSGP-II for different scenarios as compared to NSGP-II. This indicates that NSGP-II-NA has bigger HV as compared to NSGP-II. Fig. 7 is a violin plot of the IGD values of NSGP-II and NSGP-II-NA during the test process in twelve scenarios. As can be seen from Fig. 7, the overall IGD values distribution of NSGP-II-NA are smaller than that of NSGP-II, i.e., the solution found by NSGP-II-NA is of higher quality. Consequently, NSGP-II-NA outperforms NSGP-II both from the perspective of the volume of the non-dominated portion space found and from the perspective of the quality of the solutions, reflecting the superior effectiveness of NSGP-II-NA, validating its robustness in various scheduling scenarios.

Furthermore, to provide a more intuitive and interpretable evaluation, we compare NSGP-II-NA with eight commonly used manually designed scheduling rules which are constructed from

TABLE IV
THE MEAN (STANDARD DEVIATION) OF THE HV AND IGD AND FRIEDMAN'S TEST RESULT ON TRAINING INSTANCE AND TEST INSTANCE OF NSGP-II, NSGP-II-N AND NSGP-II-NA BASED ON 30 INDEPENDENT RUNS IN TWELVE MULTI-OBJECTIVE SCENARIOS WHICH ARE REPRESENTED BY THE OBJECTIVE AND UTILISATION LEVEL

		Training					
		HV			IGD		
No.	Scenario	NSGP-II	NSGP-II-N	NSGP-II-NA	NSGP-II	NSGP-II-N	NSGP-II-NA
1	<Fmax-WTmax, 0.85>	0.93(0.03)	0.93(0.02)(≈)	0.93(0.02)(≈)(≈)	0.04(0.02)	0.05(0.01)(≈)	0.05(0.01)(≈)(↑)
2	<Fmax-WTmax, 0.95>	0.93(0.02)	0.93(0.02)(≈)	0.94(0.01)(↑)(↑)	0.04(0.01)	0.04(0.01)(≈)	0.03(0.01)(↑)(↑)
3	<Tmax-WFmax, 0.85>	0.91(0.04)	0.93(0.03)(≈)	0.95(0.02)(↑)(↑)	0.05(0.02)	0.04(0.01)(↑)	0.03(0.01)(↑)(↑)
4	<Tmax-WFmax, 0.95>	0.93(0.02)	0.92(0.02)(≈)	0.94(0.02)(↑)(↑)	0.04(0.01)	0.04(0.01)(≈)	0.03(0.01)(↑)(↑)
5	<Fmean-WTmean, 0.85>	0.87(0.08)	0.90(0.07)(≈)	0.92(0.04)(↑)(≈)	0.08(0.05)	0.06(0.05)(≈)	0.04(0.02)(↑)(≈)
6	<Fmean-WTmean, 0.95>	0.93(0.05)	0.95(0.04)(≈)	0.97(0.02)(↑)(↑)	0.04(0.03)	0.03(0.03)(≈)	0.01(0.01)(↑)(↑)
7	<Fmax-Fmean, 0.85>	0.95(0.02)	0.94(0.02)(≈)	0.95(0.01)(≈)(≈)	0.03(0.01)	0.04(0.02)(≈)	0.02(0.01)(↑)(↑)
8	<Fmax-Fmean, 0.95>	0.97(0.01)	0.98(0.01)(≈)	0.98(0.01)(↑)(↑)	0.01(0.01)	0.01(0.01)(≈)	0.01(0.00)(↑)(↑)
9	<Fmax-WFmean, 0.85>	0.93(0.02)	0.94(0.02)(≈)	0.96(0.02)(↑)(↑)	0.04(0.01)	0.03(0.01)(≈)	0.02(0.01)(↑)(↑)
10	<Fmax-WFmean, 0.95>	0.97(0.01)	0.97(0.01)(≈)	0.98(0.01)(↑)(↑)	0.02(0.01)	0.01(0.00)(↑)	0.01(0.00)(↑)(↑)
11	<Fmean-WFmean, 0.85>	0.90(0.07)	0.91(0.07)(≈)	0.95(0.03)(↑)(↑)	0.06(0.05)	0.06(0.05)(≈)	0.03(0.02)(↑)(↑)
12	<Fmean-WFmean, 0.95>	0.94(0.05)	0.95(0.04)(≈)	0.97(0.03)(↑)(≈)	0.04(0.03)	0.03(0.03)(≈)	0.02(0.02)(≈)(≈)
	Win / Draw / Lose	N / A	0 / 12 / 0	10 / 2 / 0	N / A	2 / 10 / 0	10 / 2 / 0
	Average Rank	2.34	2.04	**1.62**	2.36	1.88	**1.76**
		Test					
1	<Fmax-WTmax, 0.85>	0.88(0.03)	0.89(0.02)(≈)	0.89(0.03)(↑)(↑)	0.05(0.02)	0.04(0.01)(≈)	0.04(0.02)(≈)(≈)
2	<Fmax-WTmax, 0.95>	0.94(0.02)	0.95(0.01)(↑)	0.95(0.01)(↑)(≈)	0.03(0.01)	0.03(0.01)(≈)	0.02(0.01)(↑)(↑)
3	<Tmax-WFmax, 0.85>	0.74(0.02)	0.76(0.05)(↑)	0.77(0.04)(↑)(≈)	0.25(0.03)	0.23(0.05)(≈)	0.23(0.04)(↑)(≈)
4	<Tmax-WFmax, 0.95>	0.94(0.02)	0.95(0.02)(≈)	0.96(0.01)(↑)(↑)	0.03(0.01)	0.03(0.01)(≈)	0.02(0.01)(↑)(↑)
5	<Fmean-WTmean, 0.85>	0.13(0.02)	0.14(0.01)(↑)	0.17(0.16)(↑)(≈)	0.88(0.03)	0.87(0.02)(↑)	0.84(0.16)(↑)(≈)
6	<Fmean-WTmean, 0.95>	0.91(0.07)	0.94(0.05)(↑)	0.95(0.03)(↑)(≈)	0.06(0.04)	0.04(0.03)(↑)	0.03(0.02)(↑)(≈)
7	<Fmax-Fmean, 0.85>	0.81(0.02)	0.81(0.02)(≈)	0.81(0.02)(≈)(≈)	0.03(0.01)	0.03(0.01)(≈)	0.02(0.01)(≈)(≈)
8	<Fmax-Fmean, 0.95>	0.96(0.01)	0.96(0.01)(≈)	0.97(0.01)(↑)(≈)	0.02(0.01)	0.02(0.01)(≈)	0.02(0.01)(↑)(≈)
9	<Fmax-WFmean, 0.85>	0.92(0.03)	0.93(0.03)(≈)	0.95(0.01)(↑)(↑)	0.04(0.02)	0.03(0.02)(↑)	0.02(0.01)(↑)(↑)
10	<Fmax-WFmean, 0.95>	0.68(0.01)	0.69(0.03)(≈)	0.70(0.01)(↑)(↑)	0.03(0.01)	0.02(0.01)(↑)	0.02(0.00)(↑)(↑)
11	<Fmean-WFmean, 0.85>	0.86(0.11)	0.88(0.09)(≈)	0.91(0.05)(↑)(↑)	0.09(0.09)	0.08(0.07)(≈)	0.06(0.03)(↑)(↑)
12	<Fmean-WFmean, 0.95>	0.90(0.08)	0.93(0.04)(≈)	0.94(0.03)(↑)(≈)	0.05(0.05)	0.03(0.03)(≈)	0.02(0.02)(↑)(≈)
	Win / Draw / Lose	N / A	4 / 8 / 0	11 / 1 / 0	N / A	4 / 8 / 0	10 / 2 / 0
	Average Rank	2.27	2.19	**1.54**	2.3	2.07	**1.63**

combinations of sequencing and routing rules that are selected based on their good performance as shown in [48].

The sequencing heuristics include:

- **First-In-First-Out** – prioritizing jobs that arrived earliest (FIFO),
- **Most Operation Number Remaining** – selecting jobs with the most remaining operations (MOPNR),
- **Least Work Remaining** – favoring jobs with the smallest remaining processing workload (LWKR), and
- **Most Work Remaining** – choosing jobs with the largest remaining workload (MWKR).

The routing heuristics include:

- **Shortest Processing Time** – assigning jobs to machines with the least processing time (SPT), and
- **Earliest End Time** – assigning jobs to machines that can complete the job the earliest (**EET**).

The "←" in Table V indicates that the current value is the same as the value in the previous column in the same row. As can be seen from Table V, the algorithm proposed in this paper outperforms all manual rule combinations in all scenarios. This further validates the effectiveness and generalisation capability of the evolved scheduling rules produced by NSGP-II-NA.

B. Quality of Learned Scheduling Heuristics in the Evolutionary Process

Figs. 8 and 9 show the violin plots of the HV and IGD values of NSGP-II and NSGP-II-NA during the training process in twelve multi-objective scenarios. Throughout the evolutionary process, NSGP-II-NA consistently demonstrates superior effectiveness compared to NSGP-II. In terms of HV, NSGP-II-NA achieves higher values, indicating that it effectively finds a diverse set of high-quality solutions across most scenarios. This advantage is evident from the early generations and persists through to the later generations. Similarly, IGD values reveal that NSGP-II-NA consistently finds solutions closer to the approximated true

TABLE V
THE MEAN AND STANDARD DEVIATION OF THE TEST PERFORMANCE OF THE PROPOSED METHODS AND COMPARISON METHODS. (THE STANDARD DEVIATION OF THE MANUAL RULE TEST RESULTS IS 0.00.)

		test				
No.	Scenario	FIFO+SPT	FIFO+EET	MOPNR+SPT	MOPNR+EET	NSGP-II-NA
1	<Fmax-WTmax, 0.85>	32061.92,104627.16	←	19219.30,58758.88	←	1428.61(175.90),2171.12(562.45)(↑)
2	<Fmax-WTmax, 0.95>	110450.92,355709.62	←	53676.62,182673.76	←	2437.14(424.31),4101.54(1146.62)(↑)
3	<Tmax-WFmax, 0.85>	31367.45,3040.73	←	18469.30,2737.19	←	4193.16(514.81),950.89(251.45)(↑)
4	<Tmax-WFmax, 0.95>	109750.42,7055.58	←	52926.62,5971.18	←	5347.33(1267.37),2122.07(531.65)(↑)
5	<Fmean-WTmean, 0.85>	1380.83,2155.93	←	1246.16,1844.16	←	386.11(1.65),78.24(3.65)(↑)
6	<Fmean-WTmean, 0.95>	3214.96,6170.52	←	2721.20,5075.01	←	559.45(7.93),320.14(24.82)(↑)
7	<Fmax-Fmean, 0.85>	32061.92,1380.83	←	19219.30,1246.16	←	1721.36(379.70),393.51(8.97)(↑)
8	<Fmax-Fmean, 0.95>	110450.92,3214.96	←	53676.62,2721.20	←	4754.44(2934.38),586.02(35.75)(↑)
9	<Fmax-WFmean, 0.85>	110450.91,3040.73	←	19219.30,2737.20	←	1869.13(618.41),856.12(25.43)(↑)
10	<Fmax-WFmean, 0.95>	110450.92,7055.58	←	53676.62,5971.18	←	5824.85(3724.75),1207.21(100.16)(↑)
11	<Fmean-WFmean, 0.85>	1380.83,3040.73	←	1246.16,2737.20	←	386.49(2.61),833.30(5.74)(↑)
12	<Fmean-WFmean, 0.95>	3214.96,7055.58	←	2721.20,5971.18	←	560.46(8.93),1140.74(28.81)(↑)
No.	Scenario	LWKR+SPT	LWKR+EET	MWKR+SPT	MWKR+EET	NSGP-II-NA
1	<Fmax-WTmax, 0.85>	22731.56,67203.28	←	←	←	1428.61(175.90),2171.12(562.45)(↑)
2	<Fmax-WTmax, 0.95>	70889.78,233201.89	←	←	←	2437.14(424.31),4101.54(1146.62)(↑)
3	<Tmax-WFmax, 0.85>	22001.06,2548.45	←	←	←	4193.16(514.81),950.89(251.45)(↑)
4	<Tmax-WFmax, 0.95>	70162.28,5249.98	←	←	←	5347.33(1267.37),2122.07(531.65)(↑)
5	<Fmean-WTmean, 0.85>	1161.29,1663.27	←	←	←	386.11(1.65),78.24(3.65)(↑)
6	<Fmean-WTmean, 0.95>	2388.91,4359.52	←	←	←	559.45(7.93),320.14(24.82)(↑)
7	<Fmax-Fmean, 0.85>	22731.56,1161.29	←	←	←	1721.36(379.70),393.51(8.97)(↑)
8	<Fmax-Fmean, 0.95>	70889.78,2388.91	←	←	←	4754.44(2934.38),586.02(35.75)(↑)
9	<Fmax-WFmean, 0.85>	22731.56,2548.45	←	←	←	1869.13(618.41),856.12(25.43)(↑)
10	<Fmax-WFmean, 0.95>	70889.78,5250.00	←	←	←	5824.85(3724.75),1207.21(100.16)(↑)
11	<Fmean-WFmean, 0.85>	1161.29,2548.45	←	←	←	386.49(2.61),833.30(5.74)(↑)
12	<Fmean-WFmean, 0.95>	2388.91,5249.98	←	←	←	560.46(8.93),1140.74(28.81)(↑)

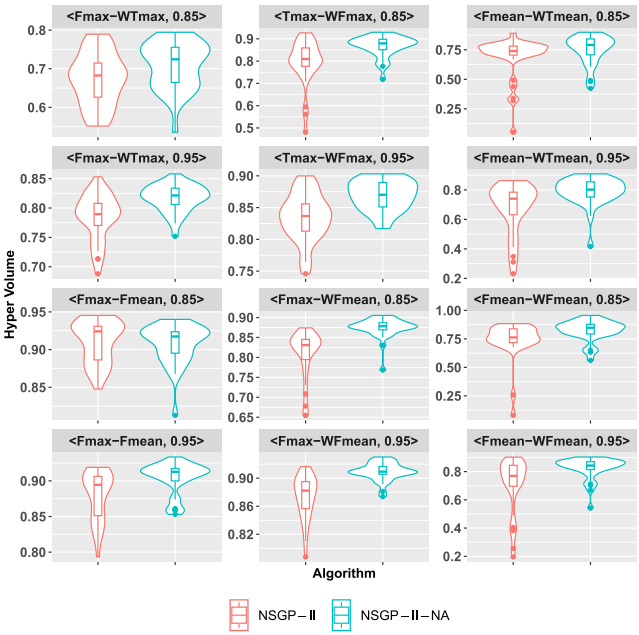


Fig. 6. Violin plots of **HV** values of the 12 multi-objective *test scenarios* for NSGP-II and NSGP-II-NA.

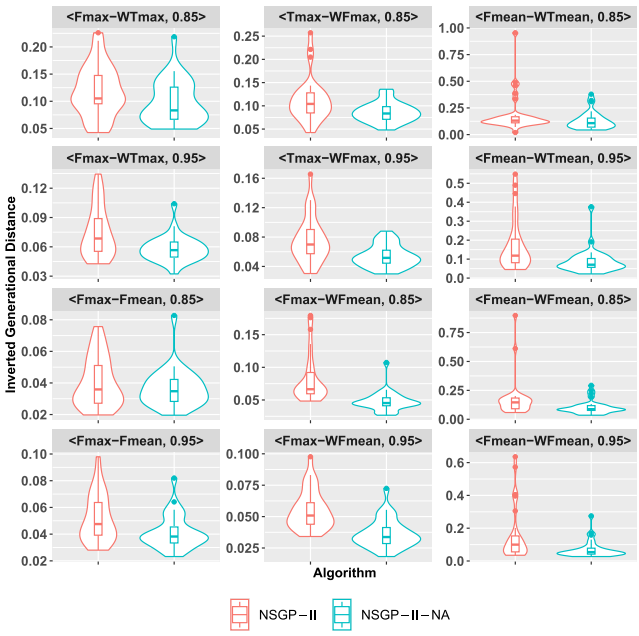


Fig. 7. Violin plots of **IGD** values of the 12 multi-objective **test scenarios** for NSGP-II and NSGP-II-NA.

Pareto fronts compared to NSGP-II. Analyzing the effectiveness changes from the early to later generations during evolution verifies the effectiveness of NSGP-II-NA in learning high-quality scheduling heuristics. This analysis highlights NSGP-II-NA's capability to develop superior scheduling heuristics throughout the evolutionary process.

C. Learned Pareto Fronts

Fig. 10 shows the learned Pareto fronts with the medium HV value of NSGP-II and NSGP-II-NA in twelve multi-objective

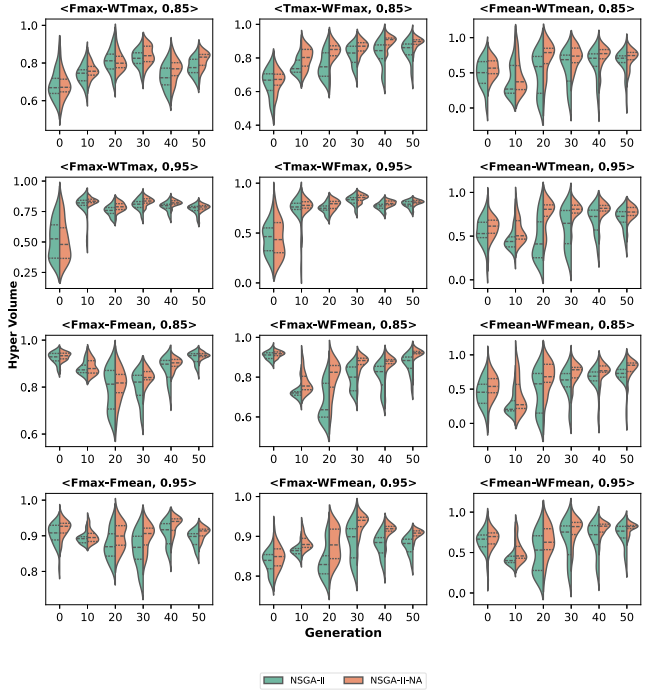


Fig. 8. The violin plots of the HV values of NSGP-II and NSGP-II-NA on **the training instances** at generations 0, 10, 20, 30, 40, and 50, in twelve multi-objective scenarios.

scenarios. We can see that the Pareto fronts found by NSGP-II are dominated by the ones found by NSGP-II-NA in all scenarios, i.e., NSGP-II-NA can always find better Pareto fronts. This shows the effectiveness of NSGP-II-NA in learning better Pareto fronts.

D. Rule Size

To validate the effectiveness of the niching strategy in enhancing the interpretability of the Pareto fronts obtained by the proposed algorithm, we conduct a comparison between the rule sizes of the Pareto fronts achieved by NSGP-II-A and NSGP-II-NA. The only difference between NSGP-II-A and NSGP-II-NA is whether the niching strategy is used. Fig. 11 shows a scatterplot of the average rule size of the Pareto fronts plotted against its corresponding average fitness values during each run of the 12 multi-objective scenarios during test process. It can be seen that in most of the scenarios, the rule sizes obtained by NSGP-II-NA are smaller than those obtained by NSGP-II-A for similar fitness values, and in only a few scenarios (i.e., <Fmean-WTmean, 0.85>, <Fmax-Fmean, 0.85> and <Fmax-WFmean, 0.85>), there is no significant difference in the rule sizes obtained by NSGP-II-NA and NSGP-II-A. Therefore, it can be concluded that the niching strategy plays a positive role in improving the interpretability of Pareto fronts obtained by NSGP-II-NA.

E. Population Diversity

This paper uses Entropy to measure the diversity of individuals of GP in DFJSS [49]. The entropy is calculated as

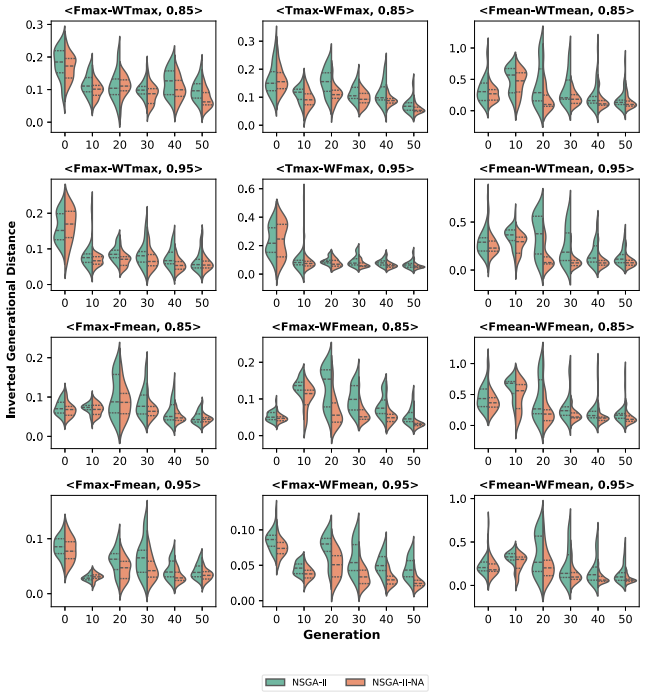


Fig. 9. The violin plots of the IGD values of NSGP-II and NSGP-II-NA on **the training instances** at generations 0, 10, 20, 30, 40, and 50, in twelve multi-objective scenarios.

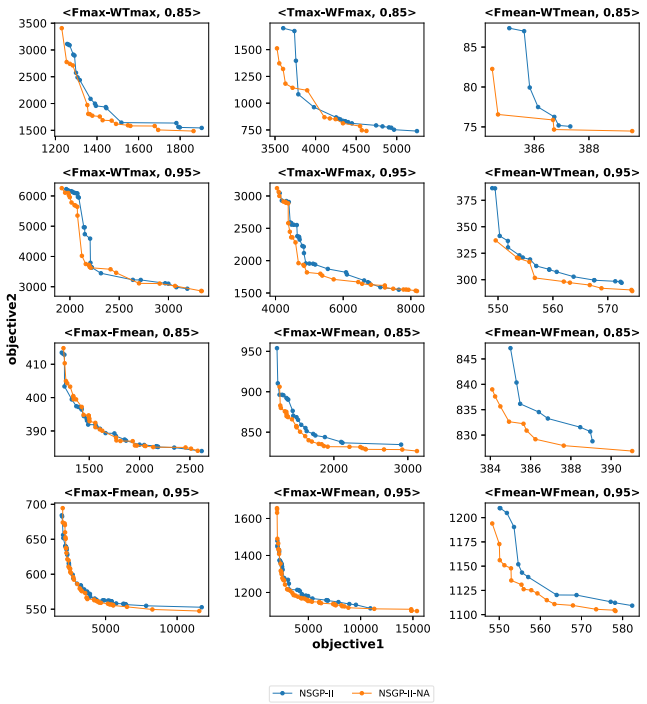


Fig. 10. The learned Pareto fronts of NSGP-II and NSGP-II-NA in twelve multi-objective scenarios.

entropy $= -\sum_{c \in C}(|c|/|\text{inds}|)\log(|c|/|\text{inds}|)$, where C represents the set of clusters derived from the DBScan clustering algorithm. A higher entropy value signifies greater diversity among individuals for a given task.

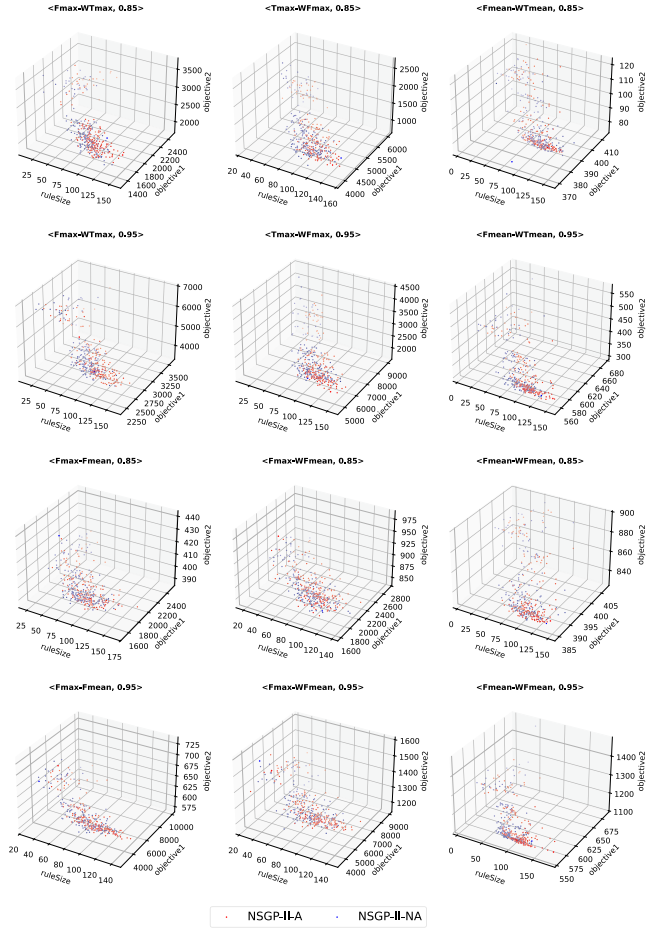


Fig. 11. The average rule size of the Pareto fronts of NSGP-II-A and NSGP-II-NA on the test instances at generations 0, 10, 20, 30, 40, and 50, in twelve multi-objective scenarios for each run.

In order to verify the effectiveness of each of our proposed strategies in enhancing population diversity separately, we compared and analyzed four algorithms, NSGP-II, NSGP-II-A, NSGP-II-N and NSGP-II-NA. Fig. 12 shows the curves of entropy values of NSGP-II, NSGP-II-A, NSGP-II-N, and NSGP-II-NA over 30 independent runs in twelve multi-objective scenarios. The results show that NSGP-II-NA has the largest diversity, followed by NSGP-A and NSGP-II-N, and NSGP-II has the relatively smallest diversity. All of them, except basic compared NSGP-II, are effective in maintaining population diversity throughout the evolutionary process. NSGP-II-A only uses the elite individual guided dynamic space optimization strategy described above, in order to avoid generating offspring individuals that are too similar, we require that the PCs of individuals in the offspring are different from each other. The result shows that it's a powerful way to improve population diversity. Meanwhile, NSGP-II-N only uses the niching strategy, we group individuals based on their PCs, and only one of the individuals in each group will be selected as a representative to participate in the breeding process. By observing the diversity curve of NSGP-II-N, it can be proved that NSGP-II-N is effective in maintaining population diversity throughout the evolutionary process. Finally, the diversity curve of NSGP-II-NA shows that

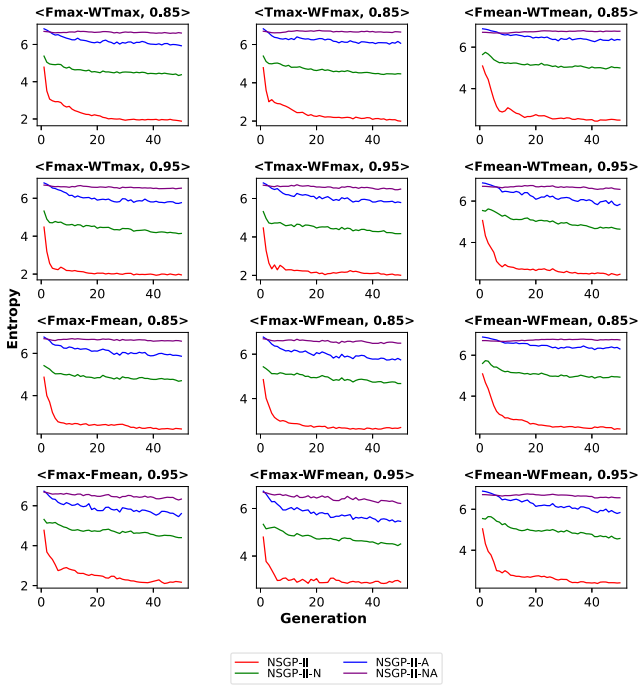


Fig. 12. The curves of entropy values of NSGP-II, NSGP-II-A, NSGP-II-N, and NSGP-II-NA over 30 independent runs in twelve multi-objective scenarios.

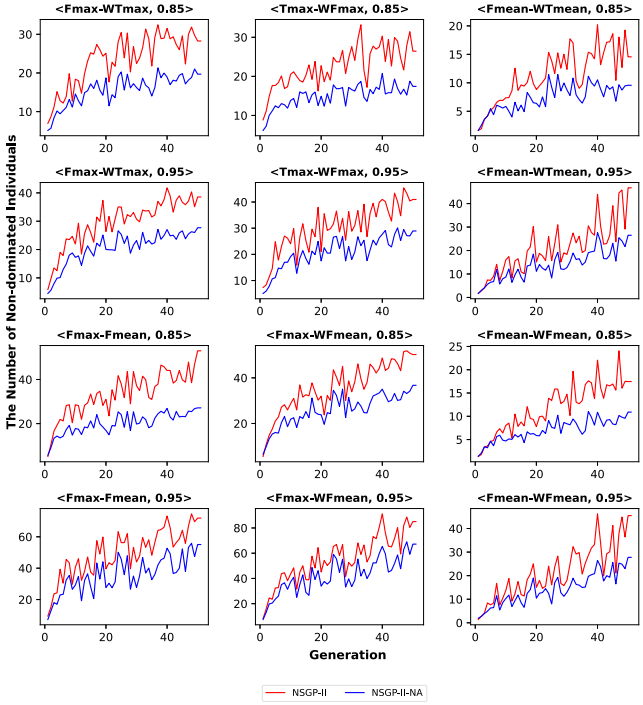


Fig. 13. The curves of the average number of non-dominated individuals of NSGP-II and NSGP-II-NA along with generations over 30 independent runs in twelve multi-objective scenarios.

the combination of the two strategies makes the population diversity reach a better level.

VI. Further Analysis

A. Number of Non-Dominated Individuals

Fig. 13 illustrates the curves depicting the average number of non-dominated individuals of NSGP-II and NSGP-II-NA across 30 independent runs in twelve multi-objective scenarios. The results demonstrate that both NSGP-II and NSGP-II-NA can progressively learn an increasing number of nondominated solutions over successive generations. This trend underscores the efficiency of employing multi-objective GP for deriving scheduling heuristics in MO-DFJSS.

Notably, NSGP-II-NA produces fewer non-dominated solutions compared to NSGP-II. However, considering the effectiveness of NSGP-II-NA, if an algorithm with fewer non-dominated individuals achieves a significantly better HV, it suggests that the algorithm is effectively capturing the most critical trade-offs. While this approach may lead to reduced diversity in the solution set, it also reflects a focus on high-quality solutions that align with specific problem priorities.

B. Population Size After Niching Process

In order to analyze the trend of population size after the niching process, we calculate and analyze the average population size of each generation after the niching process in the NSGP-II-NA algorithm. Fig. 14 shows the average size of the population at each generation in NSGP-II and after the niching process the average size of the population at each generation in NSGP-II-NA.

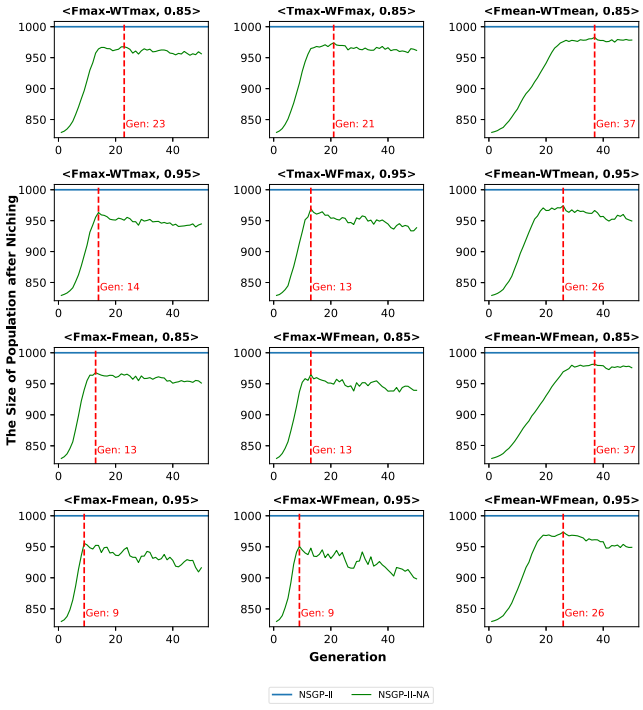


Fig. 14. The average size of population after the niching process of NSGP-II-NA along with generations over 30 independent runs in twelve multi-objective scenarios. The red points represent the peak points of the population size of NSGP-II-NA.

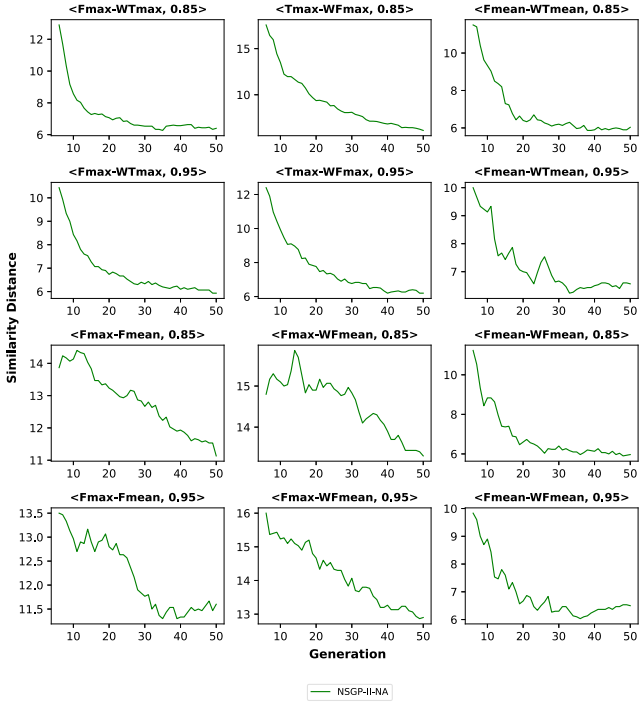


Fig. 15. The average self-adaptive distance of NSGP-II-NA after h generations over 30 independent runs in twelve multi-objective scenarios.

It can be seen that at the beginning of evolution in NSGP-II-NA, the average size of the population gradually increases, i.e., the number of individuals with similar behavior gradually decreases. When the average size of the population approximates the peak of the average size, the population size stabilizes around that value, which indicates that the population is maintained in a stable state with high diversity. These observations highlight the effectiveness of NSGP-II-NA in the evolutionary search process, ensuring thorough exploration of the search space and improving the likelihood of finding optimal solutions by maintaining population diversity and avoiding premature convergence.

C. Self-Adaptive Distance

In order to analyze the trend of the size of the constrained search space during the dynamically guided search by the elite individual guided dynamic space optimization strategy. And the self-adaptive distance can reflect the size of the range to some extent, we computed the average self-adaptive distance obtained in NSGP-II-NA. Fig. 15 shows the average distance per generation after 5 generations, calculated from the individual information in the archive over 30 independent runs in twelve multi-objective scenarios. By examining the data presented in Fig. 15, we can see the similarity distance obtained from the information of individuals within the archive is high at the beginning stage of evolutionary process, indicating significant population diversity. As generations progress, the similarity distance decreases and stabilizes, suggesting that the algorithm has adapted dynamically to the search space and allow the algorithm can search in a smaller space. Combined with the good performance demonstrated in the experimental results of NSGP-II-NA, this implies the switch from a fixed to a median-based α allows the search to focus more around the more valuable regions presented in the archive, improving the efficiency of finding optimal solutions. The consistent pattern across various fitness metrics demonstrates the strategy's robustness and its ability to balance exploration and exploitation. The decreasing similarity distance after the initial generations indicates effective convergence towards high-quality solutions while maintaining sufficient diversity to avoid premature convergence. So, we can conclude that the elite individual guided dynamic space optimization strategy in NSGP-II-NA effectively enhances the evolutionary search process.

TABLE VI
MEAN(STANDARD DEVIATION) OF TRAINING TIME (IN MINUTES) OF NSGP-II AND NSGP-II-NA ACCORDING TO 30 INDEPENDENT RUNS OF TWELVE SCENARIOS

No.	Scenario	NSGP-II	NSGP-II-NA
1	<Fmax-WTmax, 0.85>	110.99(18.74)	117.79(10.07)(≈)
2	<Fmax-WTmax, 0.95>	121.39(18.80)	126.91(9.06)(≈)
3	<Tmax-WFmax, 0.85>	108.41(20.71)	118.00(9.58)(≈)
4	<Tmax-WFmax, 0.95>	116.07(17.59)	122.91(11.54)(≈)
5	<Fmean-WTmea, 0.85>	108.63(19.04)	111.65(14.16)(≈)
6	<Fmean-WTmea, 0.95>	114.60(17.58)	121.47(12.08)(≈)
7	<Fmax-Fmean, 0.85>	139.38(29.73)	111.84(15.05)(≈)
8	<Fmax-Fmean, 0.95>	150.06(36.01)	125.70(15.56)(≈)
9	<Fmax-WFmean, 0.85>	129.38(26.10)	112.00(11.52)(≈)
10	<Fmax-WFmean, 0.95>	152.36(40.14)	129.40(14.63)(≈)
11	<Fmean-WFmean, 0.85>	139.82(32.21)	108.87(10.76)(≈)
12	<Fmean-WFmean, 0.95>	150.85(40.77)	120.32(10.76)(≈)

D. Training Time

Table VI shows the mean and standard deviation of training time (in minutes) of NSGP-II and NSGP-II-NA in twelve scenarios over 30 independent runs. The results indicate no significant differences between them. This suggests that the proposed NSGP-II-NA enhances the effectiveness of learning scheduling heuristics without incurring additional computational costs. In other words, NSGP-II-NA achieves better performance in a similar time, confirming its efficiency.

E. Semantic Analysis of Evolved Policies

Figs. 16 and 17 show the tree structures of one of the best pair of routing rule and the sequencing rule from a scheduling heuristic obtained by NSGP-II-NA in <Fmax-Fmean, 0.95> scenario.

The routing rule is a combination of five terminals (WKR, MWT, NIQ, PT, and WIQ), where WKR is the most frequently used terminal (used 4 times). It is followed by MWT, which is used 2 times. NIQ, PT, and WIQ are used only once. The routing rule (Fig. 16) can be simplified to R_0, as shown in equation (3).

$$R_0 = S_1 * S_2 + S_3 \tag{3}$$

where

$$S_1 = WKR/(WKR + MWT + WIQ) \tag{4}$$

$$S_2 = \max\{MWT, WKR\} \tag{5}$$

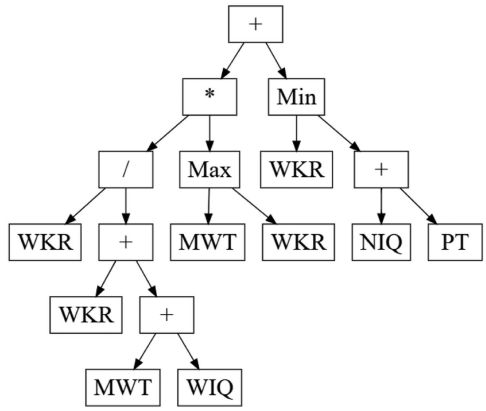


Fig. 16. Example of the best learned *routing rule* for <Fmax-Fmean, 0.95> by NSGP-II-NA in scenario 7.

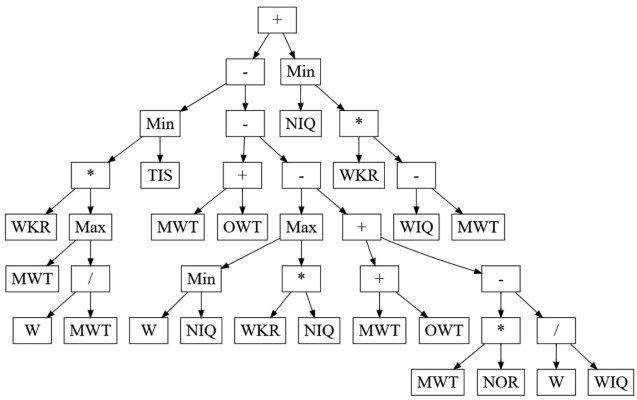


Fig. 17. Example of the best learned *sequencing rule* for <Fmax-Fmean, 0.95> by NSGP-II-NA in scenario 7.

if $S_2 = WKR$ **then**
  if $S_3 = WKR$ **then**
    $R_0 = WKR * (1 + S_1)$
  else
    $R_0 = S_1 * WKR + NIQ + PT$
  end if
else
  if $S_3 = WKR$ **then**
    $R_0 = S_1 * MWT + WKR$
  else
    $R_0 = S_1 * MWT + NIQ + PT$
  end if
end if

$$S_3 = \min\{WKR, NIQ + PT\} \tag{6}$$

To make it clearer, R_0 can be converted to the following IF-ELSE format rule set.

From the four decision expressions for R0 deduced above, we can see that WKR is involved in decision making in three out of the four expressions. It is easy to see that $S_1 \leq 1$ from equation (4). Therefore, in two of the three expressions WKR is the key variable in the decision. Also NIQ and PT play an important decision-making role in two of these expressions.

The sequencing rule is a combination of eight terminals (WKR, TIS, WIQ, MWT, NOR, W, OWT, and NIQ). MWT is the most frequently used terminal, which is used 6 times. It is followed by WKR, W and NIQ, which are used 3 times. WIQ and OWT are used 2 times. TIS and NOR are used only once. The frequency of terminal usage shows that for sequencing rule, MWT plays an important role.

These observations align well with practical expectations in MO-DFJSS. For instance, MWT is a direct indicator of machine utilization and system congestion, which are crucial for efficient scheduling. Similarly, WKR reflects the workload balance across jobs, making it vital for effective decision-making. The frequent use of NIQ and PT further supports this reasoning, as the number of operations in a queue and the processing time are essential for estimating the load and adjusting schedules dynamically. Collectively, these insights demonstrate that the identified rules incorporate key scheduling factors, making them both intuitive and practically relevant.

VII. Conclusion

In this paper, we introduced a novel approach to enhancing the performance of GP in MO-DFJSS by integrating an elite archive and a niching strategy. Our extensive experiments on multiple MO-DFJSS scenarios validated the superiority of NSGP-II-NA over existing approaches, particularly in terms of HV and IGD metrics. Our algorithm, NSGP-II-NA, demonstrated significant improvements in both the diversity and quality of the generated Pareto fronts by leveraging non-dominated individuals to guide the evolutionary search dynamically. The elite archive allowed us to effectively preserve and utilize high quality solutions from previous generations, leading to better convergence and robustness across different scenarios.

The niching strategy based on the behavior of individuals and group selection not only controlled the size of the evolved scheduling rules but also improved the interpretability of the resulting Pareto fronts. This balance between solution quality and rule simplicity is crucial for practical applications where the clarity of scheduling heuristics is essential.

The proposed NSGP-II-NA algorithm sets a strong foundation for future work in the field of evolutionary scheduling. The integration of dynamic search guidance and niching strategies offers promising directions for enhancing the efficiency and effectiveness of GP in solving complex, real-world multiobjective scheduling problems.

Future research will focus on further optimising the niching strategy and exploring additional dynamic events within the job shop environment to expand the applicability of this approach.

References

[1] S. Winkelhaus, E. H. Grosse, and S. Morana, "Towards a conceptualisation of order picking 4.0," *Comput. Ind. Eng.*, vol. 159, Sep. 2021, Art. no. 107511.

[2] B. Illés, P. Tamás, P. Dobos, and R. Skapinyecz, "New challenges for quality assurance of manufacturing processes in industry 4.0," *Solid State Phenomena*, vol. 261, pp. 481–486, Aug. 2017.

[3] Y. P. Dave, A. S. Shelat, D. S. Patel, and R. H. Jhaveri, "Various job scheduling algorithms in cloud computing: A survey," in *Proc. Int. Conf. Inf. Commun. Embed. Syst.*, 2014, pp. 1–5.

[4] F. Zhang, Y. Mei, S. Nguyen, and M. Zhang, "A preliminary approach to evolutionary multitasking for dynamic flexible job shop scheduling via genetic programming," in *Proc. Genet. Evol. Comput. Conf.*, 2020, pp. 107–108.

[5] X.-N. Shen and X. Yao, "Mathematical modeling and multi-objective evolutionary algorithms applied to dynamic flexible job shop scheduling problems," *Inf. Sci.*, vol. 298, pp. 198–224, Mar. 2015.

[6] F. Zhang, Y. Mei, S. Nguyen, K. C. Tan, and M. Zhang, "Instance-rotation-based surrogate in genetic programming with brood recombination for dynamic job-shop scheduling," *IEEE Trans. Evol. Comput.*, vol. 27, no. 5, pp. 1192–1206, Oct. 2023.

[7] D. Yang, X. Zhou, Z. Yang, Q. Jiang, and W. Feng, "Multi-objective optimization model for flexible job shop scheduling problem considering transportation constraints: A comparative study," in *Proc. IEEE. Congr. Evol. Comput.*, 2020, pp. 1–8.

[8] H. Goel and N. Chamoli, "Job scheduling algorithms in cloud computing: A survey," *Int. J. Comput.*, vol. 95, no. 23, pp. 19–22, Jun. 2014.

[9] J. R. Koza and R. Poli, "Genetic programming," in *Search Methodologies*. New York, NY, USA: Springer, 2005, pp. 127–164.

[10] M. Xu, Y. Mei, F. Zhang, and M. Zhang, "A semantic genetic programming approach to evolving heuristics for multi-objective dynamic scheduling," in *Proc. Aust. Joint Conf. Artif. Intell.*, 2024, pp. 403–415.

[11] F. Zhang, Y. Mei, S. Nguyen, and M. Zhang, "Survey on genetic programming and machine learning techniques for heuristic design in job shop scheduling," *IEEE Trans. Evol. Comput.*, vol. 28, no. 1, pp. 147–167, Feb. 2024.

[12] F. Zhang, Y. Mei, S. Nguyen, and M. Zhang, "Collaborative multifidelity-based surrogate models for genetic programming in dynamic flexible job shop scheduling," *IEEE Trans. Cybern.*, vol. 52, no. 8, pp. 8142–8156, Aug. 2022.

[13] M. Xu, Y. Mei, F. Zhang, and M. Zhang, "Genetic programming with diverse partner selection for dynamic flexible job shop scheduling," in *Proc. Genet. Evol. Comput. Conf.*, 2022, pp. 615–618.

[14] M. Xu, Y. Mei, F. Zhang, and M. Zhang, "Niching genetic programming to learn actions for deep reinforcement learning in dynamic flexible scheduling," *IEEE Trans. Evol. Comput.*, early access, May, 01, 2024, doi: 10.1109/TEVC.2024.3395699.

[15] Y. Liu, F. Zhang, Y. Sun, and M. Zhang, "Evolutionary trainer-based deep Q-network for dynamic flexible job shop scheduling," *IEEE Trans. Evol. Comput.*, vol. 29, no. 3, pp. 749–763, Jun. 2025.

[16] F. Zhang, Y. Mei, and M. Zhang, "Evolving dispatching rules for multi-objective dynamic flexible job shop scheduling via genetic programming hyper-heuristics," in *Proc. IEEE Congr. Evol. Comput.*, 2019, pp. 1366–1373.

[17] M. Xu, Y. Mei, F. Zhang, and M. Zhang, "Multi-objective genetic programming based on decomposition on evolving scheduling heuristics for dynamic scheduling," in *Proc. Companion Conf. Genet. Evol. Comput.*, 2023, pp. 427–430.

[18] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.

[19] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the strength pareto evolutionary algorithm," *TIK-Rep.*, vol. 103, Jul. 2001, doi: 10.3929/ethz-a004284029.

[20] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Trans. Evol. Comput.*, vol. 11, no. 6, pp. 712–731, Dec. 2007.

[21] T. Hildebrandt and J. Branke, "On using surrogates with genetic programming," *Evol. Comput.*, vol. 23, no. 3, pp. 343–367, Sep. 2015.

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

[23] F. Zhang, Y. Mei, S. Nguyen, and M. Zhang, "A preliminary approach to evolutionary multitasking for dynamic flexible job shop scheduling via genetic programming," in *Proc. Genet. Evol. Comput. Conf.*, 2020, pp. 107–108.

[24] M. urasević and D. Jakobović, "A survey of dispatching rules for the dynamic unrelated machines environment," *Expert Syst. Appl.*, vol. 113, pp. 555–569, Dec. 2018.

[25] M. L. Pinedo, *Scheduling: Theory, Algorithms, and Systems*. New York, NY, USA: Springer, 2022.

[26] F. Zhang, Y. Mei, and M. Zhang, "Genetic programming with multi-tree representation for dynamic flexible job shop scheduling," in *Proc. Aust. Joint Conf. Artif. Intell.*, 2018, pp. 472–484.

[27] F. Zhang, Y. Mei, S. Nguyen, and M. Zhang, "Correlation coefficient-based recombinative guidance for genetic programming hyperheuristics in dynamic flexible job shop scheduling," *IEEE Trans. Evol. Comput.*, vol. 25, no. 3, pp. 552–566, Jun. 2021.

[28] X. Cai, Y. Li, Z. Fan, and Q. Zhang, "An external archive guided multiobjective evolutionary algorithm based on decomposition for combinatorial optimization," *IEEE Trans. Evol. Comput.*, vol. 19, no. 4, pp. 508–523, Aug. 2015.

[29] K. Li, R. Chen, G. Fu, and X. Yao, "Two-archive evolutionary algorithm for constrained multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 23, no. 2, pp. 303–315, Apr. 2019.

[30] S. Wang, Y. Mei, and M. Zhang, "Two-stage multi-objective genetic programming with archive for uncertain capacitated arc routing problem," in *Proc. Genet. Evol. Comput. Conf.*, 2021, pp. 287–295.

[31] M. Xu, F. Zhang, Y. Mei, and M. Zhang, "Genetic programming with archive for dynamic flexible job shop scheduling," in *Proc. IEEE Congr. Evol. Comput.*, 2021, pp. 2117–2124.

[32] Y. Hong, W.-L. Liu, J. Zhong, P. Liang, J. Guo, and C. Li, "Archive-based cooperative coevolution genetic programming for workflow scheduling," in *IEEE Conf. Artif. Intell.*, 2024, pp. 216–219.

[33] R. Chen, B. Wu, H. Wang, H. Tong, and F. Yan, "A q-learning based NSGA-II for dynamic flexible job shop scheduling with limited transportation resources," *Swarm Evol. Comput.*, vol. 90, Oct. 2024, Art. no. 101658.

[34] F. Zhang, Y. Mei, S. Nguyen, and M. Zhang, "Phenotype based surrogate-assisted multi-objective genetic programming with brood recombination for dynamic flexible job shop scheduling," in *Proc. IEEE Symp. Ser. Comput. Intell.*, 2022, pp. 1218–1225.

[35] F. Zhang, Y. Mei, and M. Zhang, "An investigation of terminal settings on multitask multi-objective dynamic flexible job shop scheduling with genetic programming," in *Proc. Companion Conf. Genet. Evol. Comput.*, 2023, pp. 259–262.

[36] G. Shi, F. Zhang, and Y. Mei, "Interpretability-aware multi-objective genetic programming for scheduling heuristics learning in dynamic flexible job shop scheduling," in *Proc. IEEE Congr. Evol. Comput.*, 2023, pp. 1–8.

[37] F. Zhang, Y. Mei, S. Nguyen, and M. Zhang, "Multitask multiobjective genetic programming for automated scheduling heuristic learning in dynamic flexible job-shop scheduling," *IEEE Trans. Cybern.*, vol. 53, no. 7, pp. 4473–4486, Jul. 2023.

[38] M. Xu, Y. Mei, F. Zhang, and M. Zhang, "Genetic programming with lexicase selection for large-scale dynamic flexible job shop scheduling," *IEEE Trans. Evol. Comput.*, vol. 28, no. 5, pp. 1235–1249, Oct. 2024.

[39] L. Spector, "Assessment of problem modality by differential performance of lexicase selection in genetic programming: A preliminary report," in *Proc. Conf. Companion Genet. Evol. Comput.*, 2012, pp. 401–408.

[40] J. C. Tay and N. B. Ho, "Evolving dispatching rules using genetic programming for solving multi-objective flexible job-shop problems," *Comput. Ind. Eng.*, vol. 54, no. 3, pp. 453–473, Apr. 2008.

[41] S. Nguyen, M. Zhang, D. Alahakoon, and K. C. Tan, "Visualizing the evolution of computer programs for genetic programming [research frontier]," *IEEE Comput. Intell. Mag.*, vol. 13, no. 4, pp. 77–94, Nov. 2018.

[42] S. Nguyen, Y. Mei, and M. Zhang, "Genetic programming for production scheduling: A survey with a unified framework," *Complex. Intell. Syst.*, vol. 3, no. 1, pp. 41–66, Mar. 2017.

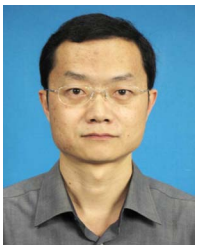
[43] S. Nguyen, M. Zhang, M. Johnston, and K. C. Tan, "Automatic programming via iterated local search for dynamic job shop scheduling," *IEEE Trans. Cybern.*, vol. 45, no. 1, pp. 1–14, Jan. 2015.

[44] K. Lei, P. Guo, Y. Wang, J. Zhang, X. Meng, and L. Qian, "Large-scale dynamic scheduling for flexible job-shop with random arrivals of new jobs by hierarchical reinforcement learning," *IEEE Trans. Ind. Informat.*, vol. 20, no. 1, pp. 1007–1018, Jan. 2024.

[45] F. Zhang, Y. Mei, and M. Zhang, "Surrogate-assisted genetic programming for dynamic flexible job shop scheduling," in *Proc. Aust. Joint Conf. Artif. Intell.*, 2018, pp. 766–772.

[46] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. Da Fonseca, "Performance assessment of multiobjective optimizers: An analysis and review," *IEEE Trans. Evol. Comput.*, vol. 7, no. 2, pp. 117–132, Apr. 2003.

[47] N. Riquelme, C. Von Lucken, and B. Baran, "Performance metrics in multi-objective optimization," in *Proc. Latin Amer. Comput. Conf.*, 2015, pp. 1–11.
[48] H.-H. Doh, J.-M. Yu, J.-S. Kim, D.-H. Lee, and S.-H. N. and, "A priority scheduling approach for flexible job shops with multiple process plans," *Int. J. Prod. Res.*, vol. 51, no. 12, pp. 3748–3764, Jun. 2013.
[49] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. 2nd. Int. Conf. Knowl. Discov. Data Mining*, 1996, pp. 226–231.
[50] W. Li, L. He, and Y. Cao, "Many-objective evolutionary algorithm with reference point-based fuzzy correlation entropy for energy-efficient job shop scheduling with limited workers," *IEEE Trans. Cybern.*, vol. 52, no. 10, pp. 10721–10734, Oct. 2022.

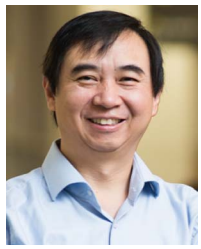


Yong Wang received the Ph.D. degree in computer science and technology from Xi'an Jiaotong University, Xi'an, China, in 2009. He is currently a Professor with the School of Computer Science and Technology, Ocean University of China, Qingdao, China, where he is also the Deputy Director with the School of Computer Science and Technology. His current research interests include software engineering, computational intelligence, and operations research.

Jie Zhang received the B.S. degree in digital media technology from the Wuhan Institute of Technology, Wuhan, China, in 2023. She is currently working toward the M.S. degree with the School of Computer Science and Technology, Ocean University of China, Qingdao, China. Her research focuses on evolutionary computation and scheduling.

Fangfang Zhang (Member, IEEE) received the B.Sc. and M.Sc. degrees from Shenzhen University, Shenzhen, China, in 2014 and 2017, respectively, and the Ph.D. degree in computer science from the Victoria University of Wellington, Wellington, New Zealand, in 2021. She is currently a Lecturer with the Centre for Data Science and Artificial Intelligence & School of Engineering and Computer Science, Victoria University of Wellington. Her research interests include evolutionary computation, hyper-heuristic learning/optimisation, job shop scheduling, surrogate modeling, and multitask learning. She was an Associate Editor for IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, *Expert Systems With Applications*, and *Swarm and Evolutionary Computation*. She holds the position of Vice-Chair of Task Force on Evolutionary Scheduling and Combinatorial Optimisation and Vice-Chair of IEEE New Zealand Central Section. She was the recipient of ACM SIGEVO Dissertation Award, Honorable Mention, and IEEE CIS Outstanding PhD Dissertation Award, for her Ph.D. thesis.

Gai-Ge Wang (Senior Member, IEEE) is currently a Professor with the Ocean University of China, Qingdao, China. His research interests include swarm intelligence, evolutionary computation, and scheduling. He was selected as one of "2021 Highly Cited Researchers" by Clarivate and "2021 and 2020 Highly Cited Chinese Researchers" in computer science and technology by Elsevier. He was the Editors-in-Chief of *International Journal of Artificial Intelligence* and *Soft Computing*, Editorial Board Member of *Journal of Computational Design and Engineering*, *Mathematics*, and *Journal of AIS*.



Mengjie Zhang (Fellow, IEEE) received the Ph.D. degree in computer science from RMIT University, Melbourne, VIC, Australia, in 2000. He is currently a Professor of computer science and Director with the Centre for Data Science and Artificial Intelligence, Victoria University of Wellington, Wellington, New Zealand. He has authored or coauthored more than 800 research papers in refereed international journals and conferences. His current research interests include evolutionary machine learning, genetic programming, image analysis, feature selection and reduction, job shop scheduling, and evolutionary deep learning and transfer learning. He is a fellow of the Royal Society of New Zealand and Engineering New Zealand and IEEE Distinguished Lecturer.