


A Preliminary Study of Indicator-based Genetic Programming for Multi-objective Dynamic Flexible Scheduling

Meng Xu¹ 

Singapore Institute of Manufacturing Technology (SIMTech), Agency for Science, Technology and Research (A*STAR), Singapore (xu_meng@simtech.a-star.edu.sg)

Abstract. Multi-objective dynamic flexible job shop scheduling (MO-DFJSS) presents an intricate task of creating optimal job schedules in a manufacturing environment characterised by uncertainty and flexibility, while simultaneously balancing multiple, often conflicting objectives. Current approaches integrate genetic programming (GP) with Pareto dominance-based and scalarising function-based multi-objective methods to learn Pareto fronts of scheduling heuristics for MO-DFJSS. However, these approaches often rely on approximate performance indicators, which can complicate achieving the final goal. In contrast, indicator-based multi-objective methods offer a more straightforward way by using performance indicators as the assessment criterion. Despite their effectiveness in other domains, no indicator-based multi-objective methods have been combined with GP for MO-DFJSS to date. Addressing this gap, this paper proposes SMS-MOGP, a fusion of GP with the SMS-EMOA, which is a popular indicator-based multi-objective algorithm, to learn scheduling heuristics for MO-DFJSS. Experiment results demonstrate that SMS-MOGP achieves comparable performance to NSGPII and significantly outperforms MOGP/D in solving the MO-DFJSS problems.

Keywords: Heuristic learning · Genetic programming · Dynamic scheduling · Multi-objective.

1 Introduction

Multi-objective dynamic flexible job shop scheduling (MO-DFJSS) is a challenging problem, that involves allocating multiple jobs to various machines, considering factors like new jobs arriving unexpectedly and the need to balance multiple, often conflicting, goals [13]. The dynamic nature requires real-time adjustments to unexpected changes. Scheduling heuristics offer a way to quickly react and make decisions based on the latest information, making them suitable for real-world dynamic environments [12]. However, designing these heuristics manually is difficult, time-consuming, and requires extensive expertise in the specific manufacturing domain [5]. Furthermore, achieving a balance between multiple

objectives and identifying a set of top-tier scheduling heuristics (Pareto front) necessitates specialized algorithms and methods that are not readily adaptable to manual design approaches.

Genetic programming (GP) techniques are extensively used to automatically develop scheduling heuristics for addressing DFJSS problems [14]. GP has also been integrated with traditional multi-objective methods to tackle MO-DFJSS challenges. Classical multi-objective methods can be broadly categorised into three types based on their offspring selection criteria [8]: 1) Pareto dominance-based methods (e.g., strength Pareto evolutionary algorithm 2 (SPEA2) [22]) and non-dominated sorting genetic algorithm II (NSGA-II) [2]; 2) scalarising function-based methods (e.g., multi-objective evolutionary algorithm based on decomposition (MOEA/D) [21]); and 3) indicator-based methods (e.g., Hyper-volume measure-based evolutionary multi-objective algorithm (SMS-EMOA) [1]). Pareto dominance-based methods employ Pareto dominance criteria along with crowding distance (NSGA-II) or clustering techniques (SPEA2) for individual comparison [8]. Scalarising function-based methods use predefined weight vectors to assess fitness and select solutions in various sub-problems (MOEA/D) [8]. In essence, the selection strategies in these multi-objective methods can be considered as an approximate performance indicator of the quality of the solution [8]. Different from these two types, indicator-based methods directly use performance indicators as the assessment criterion [3]. In this case, indicator-based methods are generally more straightforward than those based on Pareto dominance-based or scalarising function-based multi-objective methods. Numerous research studies have reported the superiority of indicator-based multi-objective methods compared to the other two types. Additionally, NSGAII, SPEA2, and MOEA/D have been integrated with GP, resulting in NSGPII [19], SPGP2 [19], and MOGP/D [12], respectively, for solving MO-DFJSS. However, to the best of our knowledge, no indicator-based methods have been investigated with GP for MO-DFJSS. To fill this gap, this paper examines the performance of indicator-based methods in the domain of MO-DFJSS. In indicator-based evolutionary multi-objective algorithms, the HV has been frequently used as an indicator. The use of the HV has clear theoretical support: the HV is a Pareto-compliant performance indicator [6]. In this case, we combine the most popular HV indicator-based method (SMS-EMOA) with GP to solve the MO-DFJSS problems, which is called SMS-MOGP. SMS-MOGP incorporates a selection operator that combines the HV measure with the concept of non-dominated sorting. To be specific, the objectives of this paper are as follows:

- To examine indicator-based methods for MO-DFJSS, this paper proposes combining the popular indicator-based algorithm (SMS-EMOA) with GP, resulting in SMS-MOGP for solving MO-DFJSS problems.
- To reduce the influence of duplicated performance (same phenotypes) by different genotypes in GP, a new comparative criterion is proposed.
- To verify the effectiveness of the proposed method, SMS-MOGP is compared with a Pareto dominance-based method (NSGPII) and a scalarising function-based method (MOGP/D) for MO-DFJSS.

- Further analysis is conducted to examine how the proposed method influences the terminal usage frequency in the learned scheduling heuristics for different MO-DFJSS scenarios.

The remaining sections of this paper are organized as follows: Section 2 provides background information, including a review of MO-DFJSS and a discussion of relevant research. The proposed methodology is outlined in detail in Section 3. Section 4 describes the experimental setup, including the dataset and parameter settings. Section 5 presents the results and experimental findings. Finally, Section 6 summarizes the key conclusions of this paper.

2 Background

2.1 Problem Modelling of MO-DFJSS

MO-DFJSS aims to optimise the scheduling of a set of jobs $\mathcal{J} = \{J_1, J_2, \dots, J_n\}$ on a collection of machines $\mathcal{M} = \{M_1, M_2, \dots, M_m\}$. Every job J_i is characterised by:

- **Sequence of operations** $[O_{i,1}, O_{i,2}, \dots, O_{i,p_i}]$: The ordered list of operations that must be performed sequentially.
- **Arrival time** r_i : The time at which the job arrives in the system.
- **Due date** d_i : The deadline by which the job should be completed.
- **Weight** w_i : The priority or importance of the job.

Each operation $O_{i,j}$ has:

- **Eligible machines** $\mathcal{M}_{i,j} \subseteq \mathcal{M}$: The subset of machines capable of performing the operation.
- **Workload** $\pi_{i,j}$: The amount of work required for the operation.
- **Processing time** $t_{i,j,k}$: The time to process operation $O_{i,j}$ on machine M_k , given by $t_{i,j,k} = \frac{\pi_{i,j}}{\gamma_k}$, where γ_k denotes the processing rate of machine M_k .

Additionally, machines are located at different geographic sites, necessitating a transportation time τ_{k_1,k_2} to transfer a job between machines M_{k_1} and M_{k_2} .

The goal of MO-DFJSS is to optimise multiple conflicting objectives. Common objectives in MO-DFJSS include:

- **Max flowtime**: $F_{\max} = \max_{i=1}^n \{C_i - r_i\}$
- **Max tardiness**: $T_{\max} = \max_{i=1}^n \{T_i\}$
- **Max weighted flowtime**: $WF_{\max} = \max_{i=1}^n \{w_i(C_i - r_i)\}$
- **Max weighted tardiness**: $WT_{\max} = \max_{i=1}^n \{w_i T_i\}$
- **Mean flowtime**: $F_{\text{mean}} = \frac{1}{n} \sum_{i=1}^n (C_i - r_i)$
- **Mean weighted tardiness**: $WT_{\text{mean}} = \frac{1}{n} \sum_{i=1}^n (w_i T_i)$

Where:

- C_i denotes the processing completion time of job J_i .
- $T_i = \max\{C_i - d_i, 0\}$ represents the tardiness of job J_i .

MO-DFJSS takes into account the following constraints:

- **Precedence constraint:** An operation cannot begin until its predecessor operation is completed and the job has been moved to the designated machine.
- **Operation assignment constraint:** Each operation must be performed on one of its eligible machines.
- **Machine capacity constraint:** Each machine can only process one operation at a time.
- **Non-preemptive scheduling constraint:** Operations cannot be interrupted once they start.

In this paper, MO-DFJSS is formulated to find a schedule that optimally balances these objectives while adhering to all constraints. For the purpose of analysis, this paper concentrates on bi-objective scenarios by selecting two of the six objectives mentioned above for each scenario. Further details regarding the selection process of these objectives are provided in Section 4.

2.2 Related Work

GP for MO-DFJSS: Genetic programming (GP), as a hyper-heuristic method, can automatically learn high-quality heuristics and has been successful in various problems such as bin packing [9], vehicle routing [4], and cloud computing [15]. GP has been widely utilised for developing scheduling heuristics for DFJSS, containing routing and sequencing rules [16]. In contrast to conventional optimisation methods, GP operates within a machine learning paradigm, involving both *training* and *test* phases. During the training phase, GP learns a population of scheduling heuristics with fitness evaluations based on a set of training DFJSS instances. For the test phase, the learned heuristics are assessed using a separate set of unseen test instances to determine their performance.

In [19], the integration of GP with two prominent Pareto dominance-based multi-objective methods, SPEA2 [22] and NSGA-II [2], led to the development of SPGP2 and NSGP2 for learning scheduling heuristics for addressing MO-DFJSS problems. The experimental findings indicate that NSGP2 surpasses SPGP2 in both training and test metrics, such as hypervolume (HV) and inverted generational distance (IGD) [7]. Beyond dominance-based approaches, [12] introduces MOGP/D, a novel multi-objective GP method that merges the advantages of MOEA/D [21] with GP. This method effectively learns a well-distributed Pareto front of scheduling heuristics for MO-DFJSS. Subsequent research in [17] enhances the NSGP2 approach to handle MO-DFJSS by integrating surrogate techniques with brood recombination, producing more effective scheduling heuristics than the conventional NSGP2 within the same training timeframe. Additionally, [13] proposes improved NSGP2 methods that consider semantic

diversity and similarity to evolve effective scheduling heuristics for MO-DFJSS. Furthermore, [20] examines the impact of terminal settings on the performance of NSGPII for MO-DFJSS, while other studies delve into interpretability [11] and multitasking [18] within the MO-DFJSS context.

Overall, MO-DFJSS has garnered significant attention, with dominance-based and scalarising function-based methods incorporated into GP proving effective for addressing this complex problem. However, to the best of our knowledge, there has been no exploration of integrating indicator-based methods with GP for solving MO-DFJSS. Indicator-based multi-objective algorithms are widely popular in the multi-objective domain. Investigating the feasibility of integrating indicator-based algorithms with GP to tackle MO-DFJSS could offer valuable insights into whether these methods can outperform dominance-based and scalarising function-based approaches in this domain.

Indicator-based MO Indicator-based multi-objective methods represent a class of optimisation techniques that differ significantly from traditional Pareto dominance-based and scalar function-based approaches. These methods leverage performance indicators as direct criteria for evaluating and guiding the search in multi-objective optimisation problems [3]. In this case, it can provide clear guidance for algorithmic decisions, such as selection and evolutionary operations, based on quantitative indicators of solution quality. The SMS-EMOA is a leading example of indicator-based multi-objective evolutionary algorithms [3]. It establishes a complete ranking of solutions by assessing their contributions to the HV indicator. This paper aims to integrate SMS-EMOA with GP for addressing MO-DFJSS. Algorithm 1 provides the general framework of SMS-EMOA. The algorithm starts by initialising a population P of scheduling heuristics with a specified size N (line 1). In each iteration, a new offspring off_{new} is generated (line 3). This typically involves applying variation operators such as mutation or crossover to selected parents from P . The offspring off_{new} is added to the current population A to form a combined set B (line 4). The combined set B undergoes non-dominated sorting, which partitions B into several non-dominated fronts $[F_1, \dots, F_k]$ (line 5). The algorithm identifies the worst-performing individual q_{worst} in the last front F_k based on the HV indicator (line 6). Then the q_{worst} is deleted from B (line 7). The algorithm terminates when the stopping condition is satisfied.

3 Indicator-based Multi-objective Genetic Programming

3.1 General Framework

SMS-MOGP integrates the popular SMS-EMOA algorithm with GP to learn scheduling heuristics to handle MO-DFJSS problems. Fig. 1 illustrates the overall framework of SMS-MOGP for MO-DFJSS. SMS-MOGP employs classical reproduction, crossover, and mutation to generate offspring for the next generation. In contrast to the original SMS-EMOA, SMS-MOGP explores the heuristic space

Algorithm 1: SMS-EMOA.

Input: Population size: N .
Output: Pareto front: \mathcal{R} .
1 Initialise a random population A of scheduling heuristics with size N ;
2 **while** the stopping condition is not satisfied **do**
3 Generate a new offspring off_{new} ;
4 $B \leftarrow A \cup off_{new}$;
5 $[F_1, \dots, F_k] \leftarrow \text{non-dominated sorting}(B)$;
6 $q_{worst} \leftarrow \arg \min_{q \in F_k} HV(F_k)$;
7 $A \leftarrow B \setminus q_{worst}$;
8 **end**
9 $\mathcal{R} \leftarrow \text{Pareto front}(A)$;
10 **return** \mathcal{R} ;

instead of the solution space, resulting in a Pareto front of scheduling heuristics upon completion of the training process. The performance of these learned scheduling heuristics is then examined on the test set. Additionally, SMS-MOGP does not perform environmental selection; all generated offspring are inherited into the next generation. Compared to NSGP-II, SMS-MOGP utilises a different parent selection strategy.

3.2 Parent Selection Strategy

SMS-MOGP introduces a new parent selection strategy based on three metrics: HV contribution, the number of individuals it is dominated by, and the number of duplicated phenotypes as the individual in the population. Calculating these three metrics is essential.

HV contribution: After fitness evaluation and non-dominated ranking, the Pareto front is obtained. For individuals in the Pareto front, their HV contribution is calculated. In the case of two objectives, the individuals in the Pareto front are sorted in ascending order based on the values of the first objective function f_1 . The sequence is subsequently sorted in descending order based on the f_2 values, as the points are mutually non-dominated. For a sorted Pareto front $\mathcal{R} = \{s_1, \dots, s_{|\mathcal{R}|}\}$, the HV contribution Δ of the boundary points (s_1 and $s_{|\mathcal{R}|}$) is assigned a significantly high value to promote their selection. The HV contribution Δ of the other individuals is calculated as in Eq. (1), where $i = 2, \dots, |\mathcal{R}| - 1$.

$$\Delta(s_i, \mathcal{R}) = |f_1(i+1) - f_1(i)| \times |f_2(i-1) - f_2(i)| \quad (1)$$

The number of dominated by: For individuals not located in the Pareto front, the HV contribution is not required. Instead, the number of individuals by which they are dominated is calculated. This metric facilitates a distinct ranking of dominated solutions, focusing on regions of the solution space that

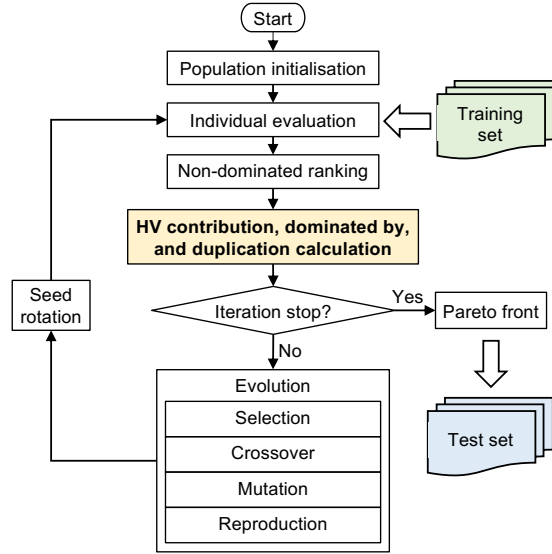


Fig. 1. The flowchart illustrating the SMS-MOGP method for developing scheduling heuristics for MO-DFJSS.

are sparsely populated. Fig. 2 illustrates the process of calculating the number of individuals dominating a given individual, which is the same method used in the original SMS-EMOA [1]. It can be observed that point a_1 is dominated by 4 points, while point a_2 is dominated by 2 points. In this scenario, if points a_1 and a_2 are compared, a_2 will be selected as the better one since it is dominated by fewer points.

The number of duplicated phenotype: Sometimes, two different individuals might have the same number of individuals dominating them. In such cases, the number of duplications is used as a comparable metric. The number of duplications is calculated based on how many individuals share the same phenotype, considering that different genotypes can produce identical phenotypes, a common occurrence in GP.

Selection rule: After obtaining the three values for all individuals in the population, the following rules are applied when selecting a parent:

1. If both individuals are on the Pareto front, prefer the one with a higher HV contribution;
2. If both individuals are not on the Pareto front, prefer the one with a smaller number of being dominated by other individuals;
3. If both individuals are not on the Pareto front and have the same number of being dominated, prefer the one with a smaller duplication number.

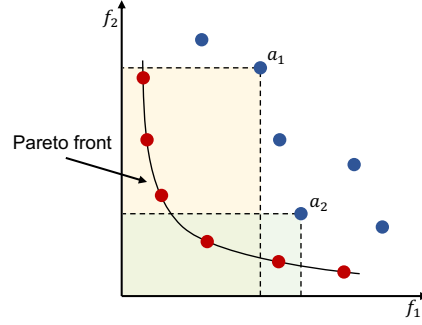


Fig. 2. An visualisation example of calculating the number of dominated by others for given individuals.

Table 1. The six scenarios.

Scenario	Objective 1	Objective 2	Utilisation
$\langle \text{Fmax-WTmax}, 0.85 \rangle$	Fmax	WTmax	0.85
$\langle \text{Fmax-WTmax}, 0.95 \rangle$	Fmax	WTmax	0.95
$\langle \text{WFmax-Tmax}, 0.85 \rangle$	WFmax	Tmax	0.85
$\langle \text{WFmax-Tmax}, 0.95 \rangle$	WFmax	Tmax	0.95
$\langle \text{Fmean-WTmean}, 0.85 \rangle$	Fmean	WTmean	0.85
$\langle \text{Fmean-WTmean}, 0.95 \rangle$	Fmean	WTmean	0.95

4 Experiment Setup

4.1 Dataset

The DFJSS simulation model [13] is employed in this paper for experimentation. Each simulation assumes 6000 jobs (with the first 1000 as warm-up jobs) that must be processed using 10 heterogeneous machines with varied processing rates randomly generated between 10 and 15. Distances between machines and the entry/exit point are uniformly distributed between 35 and 500. Transportation speed is fixed at 5 units. Jobs arrive according to a Poisson process over time, each comprising a random number of operations (uniformly distributed between 2 and 10). Jobs are weighted to reflect their importance: 20% have a weight of 1, 60% a weight of 2, and 20% a weight of 4. Workloads for each operation are uniformly distributed between 100 and 1000. The due date for each job is set by adding 1.5 times its processing time to its arrival time. Different scenarios are simulated by adjusting the utilisation level, where higher levels reflect more congested job shops.

This study explores six distinct scenarios, each involving different pairs of objectives and utilisation levels (such as 0.85 and 0.95), as detailed in Table 1. For every scenario, 50 instances are allocated for training, with another 50 unseen instances set aside for test [13].

Table 2. The terminals used by SMS-MOGP for constructing scheduling heuristics for MO-DFJSS.

Notation	Description
WIQ	Work (total processing time) in the machine’s waiting queue
NIQ	Number of operations in the machine’s waiting queue
MWT	Machine waiting time = current time - machine ready time
OWT	Operation waiting time = current time - operation ready time
PT	Processing time of the operation
NPT	Median processing time for the next operation
rDD	Relative due date = due date - current time
SLACK	Slack time
WKR	Remaining work
NOR	Job remaining number of operations
TIS	Time stay in the system = current time - release time
W	Job weight
TRANT	Transportation time

4.2 Parameter Configurations

Table 2 provides the terminals used in SMS-MOGP for constructing scheduling heuristics for MO-DFJSS. The terminals include machine-related features such as WIQ, NIQ, and MWT; operation-related features like OWT, PT, and NPT; job-related features including rDD, SLACK, WKR, NOR, TIS, and W; and transportation-related features such as TRANT. The functions used include $+$, $-$, \times , $/$, \min , \max , where arithmetic operators require two arguments, and the division operator (“/”) is safeguarded to return 1 when the divisor is 0. Additionally, the functions “min” and “max” accept two arguments, returning the minimum and maximum values, respectively.

With respect to parameter settings for the SMS-MOGP, a population size of 1000 is employed. The Pareto front for scheduling heuristics is obtained after 50 generations. Population initialisation is carried out using the Ramped-half-and-half method. The rates for crossover, mutation, and reproduction are set at 0.80, 0.15, and 0.05, respectively. Tournament selection with a tournament size of 7 is used for parent selection.

5 Results

To assess and compare the algorithms’ performance, we conducted 30 independent runs for each one. We used the Wilcoxon rank-sum test to perform pairwise comparisons among SMS-MOGP, NSGP-II, as well as MOGP/D, applying a significance threshold of 0.05. For the analysis, we use symbols “=”, “ \uparrow ”, and “ \downarrow ” to represent statistical significance, indicating whether the results are comparable to, better than, or worse than those of the other algorithms. For performance evaluation, we employed two well-established metrics: HV [23] and IGD [10]. A higher HV score or a lower IGD score indicates superior performance.

Table 3. The average (standard deviation) of the **HV** metric for 30 independent runs of NSGP-II, MOGP/D, and SMS-MOGP across six scenarios.

Scenarios	NSGP-II	MOGP/D	SMS-MOGP
<Fmax-WTmax, 0.85>	0.85(0.04)(=)	0.81(0.06)(↑)	0.84(0.04)
<Fmax-WTmax, 0.95>	0.81(0.04)(=)	0.77(0.06)(↑)	0.82(0.03)
<WFmax-Tmax, 0.85>	0.78(0.08)(=)	0.80(0.07)(=)	0.78(0.09)
<WFmax-Tmax, 0.95>	0.85(0.03)(=)	0.85(0.06)(=)	0.85(0.05)
<Fmean-WTmean, 0.85>	0.52(0.25)(=)	0.52(0.25)(=)	0.48(0.24)
<Fmean-WTmean, 0.95>	0.63(0.19)(=)	0.64(0.20)(=)	0.60(0.23)
↑ =↓	0 6 0	2 4 0	-

Table 4. The average (standard deviation) of the **IGD** metric for 30 independent runs of NSGP-II, MOGP/D, and SMS-MOGP across six scenarios.

Scenarios	NSGP-II	MOGP/D	SMS-MOGP
<Fmax-WTmax, 0.85>	0.10(0.02)(=)	0.12(0.03)(=)	0.11(0.02)
<Fmax-WTmax, 0.95>	0.11(0.02)(=)	0.14(0.04)(↑)	0.11(0.02)
<WFmax-Tmax, 0.85>	0.11(0.04)(=)	0.11(0.05)(=)	0.10(0.04)
<WFmax-Tmax, 0.95>	0.08(0.02)(=)	0.09(0.05)(=)	0.08(0.02)
<Fmean-WTmean, 0.85>	0.38(0.29)(=)	0.39(0.30)(=)	0.42(0.28)
<Fmean-WTmean, 0.95>	0.25(0.18)(=)	0.25(0.18)(=)	0.29(0.23)
↑ =↓	0 6 0	1 5 0	-

5.1 Test Performance

The average and standard deviation results of HV and IGD from 30 independent runs of different algorithms on test instances across the six scenarios are presented in Tables 3 and 4. The bottom sections of these tables provide a summary of the Wilcoxon test comparisons.

Based on the analysis of Tables 3 and 4, SMS-MOGP exhibits statistically similar HV and IGD test performance compared to NSGP-II across all six scenarios. When compared to MOGP/D, SMS-MOGP demonstrates significantly better HV performance in two scenarios focusing on max flowtime and max weighted tardiness objectives. In the remaining four scenarios, SMS-MOGP shows statistically similar HV performance. Regarding IGD, SMS-MOGP achieves significantly better performance in one scenario involving max flowtime and max weighted tardiness objectives with a utilisation level of 0.95. In the other five scenarios, SMS-MOGP demonstrates statistically similar IGD performance compared to MOGP/D. This comparison suggests that SMS-MOGP generally performs competitively with NSGP-II across all evaluated metrics and scenarios. It shows particular strengths in HV and IGD metrics under specific objective and utilisation conditions compared to MOGP/D.

We further visualise the Pareto fronts by selecting the run with median HV performance among all 30 runs of NSGP-II, MOGP/D, and the proposed SMS-MOGP across six scenarios, as depicted in Fig. 3. It is observed that more non-dominated solutions are found in scenarios focusing on max-objectives com-

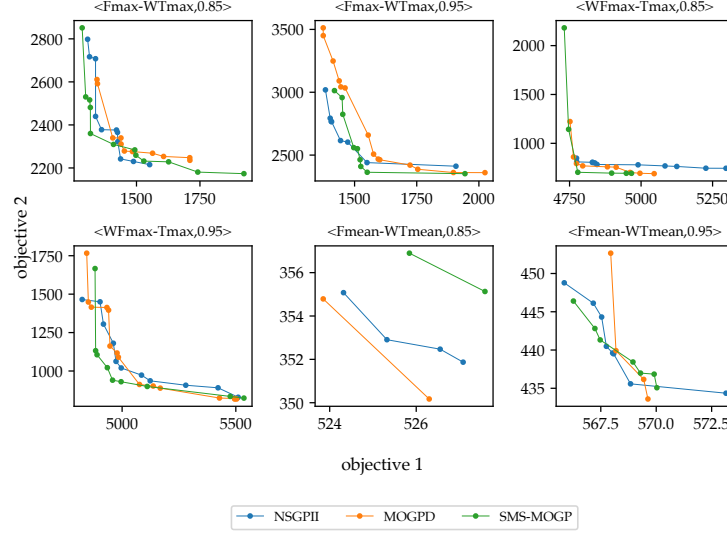


Fig. 3. The visualisation Pareto front of the run with median HV performance of NSGP/II, MOGP/D, and the proposed SMS-MOGP across 6 scenarios.

pared to those focusing on mean-objectives for all three algorithms. This observation suggests that optimising MO-DFJSS with mean-objectives might present greater difficulty. However, determining which algorithm achieves the best spread or dense distribution is challenging due to varying phenomena across different scenarios. For instance, in the scenario $\langle \text{Fmax-WTmax}, 0.85 \rangle$, SMS-MOGP's Pareto front exhibits a more spread out distribution and better extreme points. Conversely, in the scenario $\langle \text{Fmax-WTmax}, 0.95 \rangle$, MOGP/D's Pareto front displays a broader spread and superior extreme points.

5.2 Terminal Analysis

This section analyses the usage frequency of terminals to illustrate how different terminals contribute to effective scheduling heuristics on the Pareto front by SMS-MOGP. Figs. 4 and 5 show the frequency of each terminal's use in the sequencing and routing rules derived from the learned scheduling heuristics in the Pareto front across 30 runs of SMS-MOGP over six scenarios. From these results, we can draw the following observations:

1. Terminals have varying levels of importance in sequencing rules versus routing rules. For example, the terminal WKR is quite significant in sequencing rules but less so in routing rules. Additionally, the frequency of terminal usage in scenarios considering mean-objectives is lower than in those considering max-objectives, both in sequencing and routing rules.

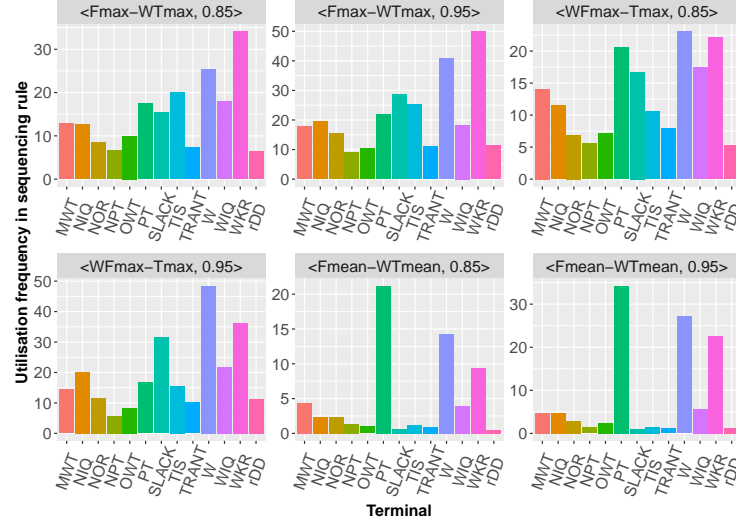


Fig. 4. The curves of average used frequency of terminals in sequencing rules of learned scheduling heuristics in Pareto front of the proposed SMS-MOGP across six scenarios

2. The importance of terminals varies with different objectives within the same rule type. For sequencing rules with max-objectives, the terminals W and WKR are the top two most frequently used. In contrast, for sequencing rules with mean-objectives, W and PT are the most frequently used terminals.
3. Across all six scenarios, the top three most frequently used terminals in routing rules are PT , $TRANT$, and WIQ . However, in scenarios considering max-objectives, $TRANT$ is the most critical terminal, whereas, in scenarios considering mean-objectives, WIQ holds the most importance.

In summary, the terminal PT plays a significant role in both sequencing and routing rules. Terminals $SLACK$, W , and WKR are vital for sequencing rules but are not frequently used in routing rules. Conversely, PT , $TRANT$, and WIQ are key criteria for routing rules.

6 Conclusions

To assess the effectiveness of indicator-based multi-objective techniques in GP for solving MO-DFJSS problems, this study introduces SMS-MOGP, integrating the SMS-EMOA algorithm with GP. SMS-MOGP employs a novel parent selection strategy based on HV contribution, domination count, and duplication number for individual comparison. Experiment results demonstrate that SMS-MOGP achieves comparable HV and IGD performance to NSGPII and significantly outperforms MOGP/D. Further analysis on terminal usage frequency reveals varying levels of importance of different terminals across different scenarios. Specifically, terminal importance differs significantly between scenarios

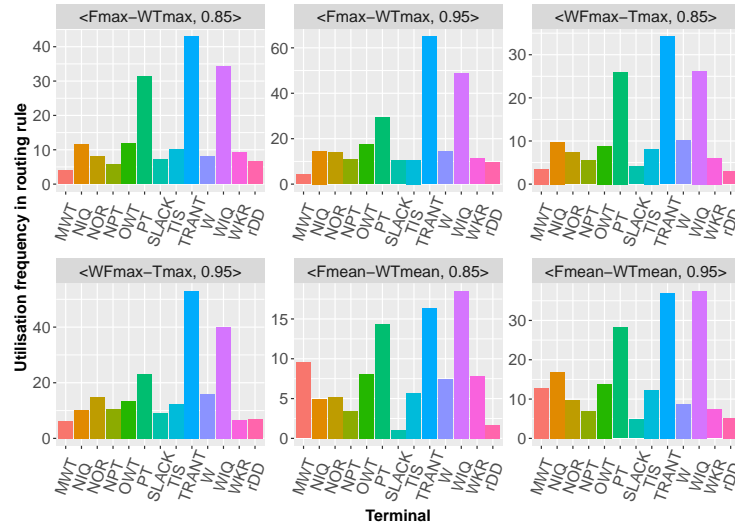


Fig. 5. The curves of average used frequency of terminals in routing rules of learned scheduling heuristics in Pareto front of the proposed SMS-MOGP across six scenarios

considering max objectives and mean objectives, providing valuable guidance for selecting suitable scheduling heuristics based on real-world optimisation objectives.

As an initial exploration, while SMS-MOGP does not surpass NSGP-II, its use of indicator metrics for selection presents a straightforward approach. Future research could enhance effectiveness by exploring multi-indicator-based parent selection strategies. Moreover, initialising high-quality scheduling heuristics by controlling terminal usage frequency for scenarios with different objectives could expedite the evolutionary process and potentially improve the performance of the final learned scheduling heuristics.

References

1. Beume, N., Naujoks, B., Emmerich, M.: Sms-emoa: Multiobjective selection based on dominated hypervolume. *European Journal of Operational Research* **181**(3), 1653–1669 (2007)
2. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation* **6**(2), 182–197 (2002)
3. Falcón-Cardona, J.G., Coello, C.A.C.: Indicator-based multi-objective evolutionary algorithms: A comprehensive survey. *ACM Computing Surveys* **53**(2), 1–35 (2020)
4. Gulić, M., Jakobović, D.: Evolution of vehicle routing problem heuristics with genetic programming. In: *Proceedings of the International Convention on Information and Communication Technology, Electronics and Microelectronics*. pp. 988–992 (2013)

5. Guo, H., Liu, J., Wang, Y., Zhuang, C.: An improved genetic programming hyper-heuristic for the dynamic flexible job shop scheduling problem with reconfigurable manufacturing cells. *Journal of Manufacturing Systems* **74**, 252–263 (2024)
6. Ishibuchi, H., Imada, R., Masuyama, N., Nojima, Y.: Dynamic specification of a reference point for hypervolume calculation in sms-emoa. In: *Proceedings of the IEEE Congress on Evolutionary Computation*. pp. 1–8. IEEE (2018)
7. Ishibuchi, H., Masuda, H., Tanigaki, Y., Nojima, Y.: Modified distance calculation in generational distance and inverted generational distance. In: *Proceedings of the Evolutionary Multi-Criterion Optimization Conference*. pp. 110–125. Springer (2015)
8. Jiang, S., Zhang, J., Ong, Y.S., Zhang, A.N., Tan, P.S.: A simple and fast hypervolume indicator-based multiobjective evolutionary algorithm. *IEEE Transactions on Cybernetics* **45**(10), 2202–2213 (2014)
9. Koza, J.R.: Genetic programming as a means for programming computers by natural selection. *Statistics and Computing* **4**, 87–112 (1994)
10. Liu, H.L., Gu, F., Zhang, Q.: Decomposition of a multiobjective optimization problem into a number of simple multiobjective subproblems. *IEEE Transactions on Evolutionary Computation* **18**(3), 450–455 (2013)
11. Shi, G., Zhang, F., Mei, Y.: Interpretability-aware multi-objective genetic programming for scheduling heuristics learning in dynamic flexible job shop scheduling. In: *Proceedings of the IEEE Congress on Evolutionary Computation*. pp. 1–8. IEEE (2023)
12. Xu, M., Mei, Y., Zhang, F., Zhang, M.: Multi-objective genetic programming based on decomposition on evolving scheduling heuristics for dynamic scheduling. In: *Proceedings of the Genetic and Evolutionary Computation Companion Conference*. pp. 427–430 (2023)
13. Xu, M., Mei, Y., Zhang, F., Zhang, M.: A semantic genetic programming approach to evolving heuristics for multi-objective dynamic scheduling. In: *Proceedings of the Australasian Joint Conference on Artificial Intelligence*. pp. 403–415. Springer (2023)
14. Xu, M., Mei, Y., Zhang, F., Zhang, M.: Niching genetic programming to learn actions for deep reinforcement learning in dynamic flexible scheduling. *IEEE Transactions on Evolutionary Computation* (2024), doi: [10.1109/TEVC.2024.3395699](https://doi.org/10.1109/TEVC.2024.3395699)
15. Xu, M., Mei, Y., Zhu, S., Zhang, B., Xiang, T., Zhang, F., Zhang, M.: Genetic programming for dynamic workflow scheduling in fog computing. *IEEE Transactions on Services Computing* **16**(4), 2657–2671 (2023)
16. Zakaria, Y., Zakaria, Y., BahaaElDin, A., Hadhoud, M.: Niching-based feature selection with multi-tree genetic programming for dynamic flexible job shop scheduling. In: *Proceedings of the International Joint Conference on Computational Intelligence*. pp. 3–27 (2019)
17. Zhang, F., Mei, Y., Nguyen, S., Zhang, M.: Phenotype based surrogate-assisted multi-objective genetic programming with brood recombination for dynamic flexible job shop scheduling. In: *Proceedings of the IEEE Symposium Series on Computational Intelligence*. pp. 1218–1225 (2022)
18. Zhang, F., Mei, Y., Nguyen, S., Zhang, M.: Multitask multiobjective genetic programming for automated scheduling heuristic learning in dynamic flexible job shop scheduling. *IEEE Transactions on Cybernetics* **53**(7), 4473–4486 (2023)
19. Zhang, F., Mei, Y., Zhang, M.: Evolving dispatching rules for multi-objective dynamic flexible job shop scheduling via genetic programming hyper-heuristics. In: *Proceedings of the IEEE Congress on Evolutionary Computation*. pp. 1366–1373 (2019)

20. Zhang, F., Mei, Y., Zhang, M.: An investigation of terminal settings on multitask multi-objective dynamic flexible job shop scheduling with genetic programming. In: Proceedings of the Genetic and Evolutionary Computation Companion Conference. pp. 259–262 (2023)
21. Zhang, Q., Li, H.: Moea/d: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation* **11**(6), 712–731 (2007)
22. Zitzler, E., Laumanns, M., Thiele, L.: Spea2: Improving the strength pareto evolutionary algorithm. TIK report **103** (2001), doi: [10.3929/ethz-a-004284029](https://doi.org/10.3929/ethz-a-004284029)
23. Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C.M., Da Fonseca, V.G.: Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Transactions on Evolutionary Computation* **7**(2), 117–132 (2003)