Multi-Objective Approach with a Distance Metric in Genetic Programming for Job Shop Scheduling

Salama, Shady

(Citation)
International Journal of Automation Technology,16(3):296-308

(Issue Date)
2022-05-05

(Resource Type)
journal article

(Version)
Version of Record

(URL)
https://hdl.handle.net/20.500.14094/0100485287

https://doi.org/10.20965/ijat.2022.p0296

Paper:

Multi-Objective Approach with a Distance Metric in Genetic Programming for Job Shop Scheduling

Shady Salama†, Toshiya Kaihara, Nobutada Fujii, and Daisuke Kokuryo

Graduate School of System Informatics, Kobe University
1-1 Rokkodai-cho, Nada-ku, Kobe, Hyogo 657-8501, Japan
†Corresponding author, E-mail: shady.salama@kaede.cs.kobe-u.ac.jp
[Received November 15, 2021; accepted March 8, 2022]

The goal of the Fourth Industrial Revolution is to develop smart factories that ensure flexibility and adaptability in complex production environments, without human intervention. Smart factories are based on three main pillars: integration through digitalization, employment of flexible structures, and the use of artificial intelligence (AI) methods. Genetic programming (GP) is one of the most promising AI approaches used in the automated design of production-scheduling rules. However, promoting diversity and controlling the bloating effect are major challenges to the success of GP algorithms in developing production-scheduling rules that deliver high-quality solutions. Therefore, we introduced a multi-objective technique to increase the diversity among GP individuals while considering the program length as an objective to avoid the bloating effect. The proposed approach employs a new diversity metric to measure the distance between GP individuals and the best rule in the current generation. Subsequently, the non-dominated sorting genetic algorithm II (NSGA-II) was used to select individuals based on three objectives: solution quality, similarity value, and program length. To assess the effectiveness of the proposed approach, we compare the two versions with three GP methods in the literature in terms of automatically generating dispatching rules on 10 benchmark instances of the job-shop scheduling problem. The experimental results show that the proposed distance measure enhances the phenotypic diversity of individuals, resulting in improved fitness values without the need for additional fitness assessments. In addition, the integration of NSGA-II with the GP algorithm facilitates the evolution of superior job shop dispatching rules with high diversity and shorter lengths under the makespan and mean tardiness objectives.

Keywords: genetic programming, diversity, bloat control, evolutionary multi-objective optimization, job shop scheduling

1. Introduction

Owing to advances in information and communication technologies, the German government proposed the term "Industry 4.0" to reduce waste, increase flexibility and self-configuration, and facilitate data acquisition and analysis in manufacturing facilities [1]. A smart factory represents the ultimate application of technologies emerging from Industry 4.0. Smart factories integrate physical and cyber technologies to realize short product life cycles and extreme mass customization in a cost-efficient manner [2]. To achieve such high automation levels, artificial intelligence (AI) methods are expected to be widely adopted to perform analysis and make decisions without human intervention. This has given rise to growing interest in developing AI methods to solve optimization problems that occur in manufacturing systems [3]. The job shop scheduling problem (JSSP) is one of the most popular combinatorial optimization models and has attracted the attention of many scholars owing to its wide applicability in practice [4]. In the last few decades, a large number of approaches have been proposed for solving JSSPs, including exact and approximate methods. However, because JSSPs have proven to be NP-hard [5], exact methods are impractical for solving real-world instances within reasonable computational time. Consequently, scheduling heuristics have been introduced to deliver near-optimal solutions in a reasonable computational time [6]. Dispatching rules have been widely used in JSSPs because of their computational efficiency, ability to make real-time scheduling decisions, and ease of implementation. Many rules are manually designed to handle various job shop configurations and objectives [7].

Because dispatching rules are developed to obtain problem-specific solutions, the manual design of sophisticated heuristics is not a trivial task in that it requires considerable time, experience, and coding effort. Recently, owing to advances in computational power and AI algorithms, various methods have been developed to automate the design of dispatching rules [8]. These methods, referred to as "*hyper-heuristics*," represent high-level iterative techniques to select or generate low-level heuristics to solve hard computational problems [9]. The motivation was to reduce the time required to generate superior heuristics. This leads to an increase in the level of general-

ity of the search methods by discovering the right method for each problem instance rather than directly solving the problem. Regarding the automated design of production scheduling rules, genetic programming (GP) is one of the most prevalent techniques [10].

Evolving dispatching rules using the GP variable length representation offers many advantages over other AI techniques. The GP algorithm explores both the structure and the corresponding parameters of a heuristic without assuming any particular distribution or domain knowledge [11]. In addition, the generated rules can be partially interpreted compared with other methods, such as neural networks. Finally, several evolutionary multi-objective methods can be combined with the GP algorithm to optimize conflicting objectives [10]. However, two crucial properties that influence the ability of the GP approach to generate high-quality rules are premature convergence and the unjustified growth of rules [12].

Similar to other evolutionary algorithms, population diversity plays an important role in preventing premature convergence and stagnation towards local optima [13]. Several techniques have been proposed to enhance the population diversity. These methods utilize non-standard selection, mutation, or replacement strategies to increase or control diversity levels [14]. In addition, the geographical distribution of individuals was commonly adopted and reported in the literature, including neighborhoods [15] and islands [16]. The most common diversity preservation mechanism in GP is the niching approach, which encourages the exploration of uncongested regions of the search space by penalizing individuals in crowded areas. The distance between GP individuals is determined by the individuals' structure (genotype) or performance (phenotype) [17]. The distance metrics in the literature have certain limitations that make them unsuitable for use in generating diverse rules for job shop scheduling problems. First, the evolution of GP rules is computationally expensive, limiting the use of phenotypic distance measures. Second, genotypic distance metrics consider the position of a node while neglecting the interaction effect between the node and its parents. Finally, many distance measures assume that all nodes have the same weight, and this is not the case in scheduling rules where nodes near the root have more impact than nodes that are far away. Therefore, in this study, a new distance metric is introduced to address these limitations.

A well-established phenomenon in GP and other variable-length genome methods is the tendency of evolved programs to grow rapidly over time without a significant return in fitness. This phenomenon is referred to as "bloat" [18]. This slows down the search process by wasting computing resources in evaluating large individuals and reduces the likelihood that genetic operators will alter important parts of the evolved programs [19]. The most common way to control bloating is to impose size limits on the evolved programs. The size constraint can be expressed using the maximum allowable tree depth or maximum number of tree nodes [20]. Other methods include code editing, modifying genetic operators, parsimony pressure, and the removal of oversized individuals [18, 21]. In addition, the size of the GP individuals is indirectly (tie-breaker) optimized in parallel with fitness and diversity, as proposed in [21–23].

In view of the above issues, this work aims to reduce the size of the GP-generated rules without negatively affecting the performance. To satisfy these two conflicting goals, a multi-objective GP approach is proposed to simultaneously optimize the quality of the solution, diversity value, and size of the generated rule. The research objectives were as follows.

1. Develop a computationally efficient distance metric for promoting diversity in the JSSPs.

2. Propose a new GP framework to control the bloat effect by simultaneously optimizing the fitness, diversity, and size of the rules.

The remainder of this paper is organized as follows. Section 2 provides a summary of related research. Section 3 describes the proposed approach. The experimental details are presented in Section 4. The results are analyzed in Section 5. Finally, Section 6 presents the conclusions and future work.

2. Related Work

2.1. Evolving Dispatching Rules Using GP

The use of the GP algorithm to evolve dispatching rules was first proposed in 2000 by Miyashita [24]. Since then, the field of automated design of dispatching rules (DRs) using GP has been very active because of the rapid growth of computational capabilities and the global shift towards more automation in production facilities [10]. The literature on the automated design of production scheduling heuristics was comprehensively reviewed with a focus on design choices such as learning methods, attributes, representation, and fitness evaluation [9]. In the same context, Nguyen et al. [10] proposed a unified framework for the automated design of DRs with GP. They presented a detailed discussion of the key components and practical issues that must be investigated before developing a GP system to generate production-scheduling rules. Additionally, GP has been employed to evolve DRs in a wide range of production scheduling problems, such as for single machine [25], parallel machines [26], job shops [27], and flexible job shops [4]. The common conclusion of these studies is that the GP algorithm is a promising approach for generating high-quality DRs that can outperform existing rules without any human intervention. In addition, the performance of GP based on a tree structure was compared with that of a neural network and linear representation [28] for designing DRs. The GP algorithm obtained the best solution quality, followed by the neural network, whereas the linear representation yielded the worst results.

2.2. Promoting Diversity

The fitness landscape of hard optimization problems, such as JSSPs, may contain more than one optimum. Therefore, it is crucial to maintain high levels of diversity among GP individuals to prevent premature convergence towards one of these optima. Although a mutation genetic operator helps achieve this goal to a certain extent, its effect is limited because the GP population usually settles towards a few high-ranking individuals [29]. Therefore, maintaining a sufficient level of diversity among individuals, especially in the early generations, is critical for efficiently exploring the search space and escaping local optima. To enhance diversity among GP individuals, two major challenges must be considered: measuring diversity and maintaining it across evolving generations [30].

From a topological point of view, if two trees are close to each other or identical, one can be easily converted to the other with a few applications of genetic operators [31]. Accordingly, to enhance the population diversity, the similarity (distance) between individuals must be evaluated, and individuals with low similarity values must survive to the next generation. Different distance metrics have been proposed in the literature, such as the number of different structural individuals (genotypes) [22], fitness values (phenotypes) [30], edit distances [23], and composite measures [32]. Burke et al. [14] examined the relationship between several measures of diversity in genetic programming and fitness values. In addition, a survey and analysis of different semantic methods used to increase the phenotypic diversity between GP individuals was presented [33]. The distance metrics in these literature reviews were analyzed using standard GP methods, such as artificial ants, even-5-parity, and symbolic regression. To the best of our knowledge, research that addressed the applicability of distance measures in promoting diversity among GP individuals in the job-shop scheduling domain has not yet been reported.

Multiple diversity mechanisms are used to reduce the risk of high-performance individuals taking over the entire population before effectively exploring the fitness landscape [34]. Diversity mechanisms can be classified into two groups: niching and non-niching. The most common niching methods are fitness sharing, clearing, and crowding. Non-niching techniques include methods such as the removal of genotype or phenotype duplicates, incest prevention, and island models [35]. The main intuition behind fitness-sharing methods is to devalue each individual's fitness at a rate that is proportional to the number of identical individuals in the population. Rather than penalizing fitness values, some scholars have suggested using multi-objective methods, where diversity is included as an explicit objective to be optimized along with fitness in the GP algorithm. A multi-objective method called age-fitness Pareto optimization was proposed [12] to promote population diversity by simultaneously optimizing the age and fitness of individuals. Similarly, Burks and Punch [22] introduced a genetic diversity technique and employed the Pareto tournament selection method to obtain a set of non-dominated GP individuals. Similarity among individuals was measured using tree fragments, and the dominance relationship between individuals was determined using both the solution quality and diversity values.

2.3. Controlling the Bloat Effect

Genetic programming is adversely affected by the bloat problem, which represents the uncontrolled growth of the size of individuals without tangible improvement in their fitness values. Many bloat control methods have been reported in the literature to mitigate the redundant growth of GP individuals, and ten bloat control techniques were compared [18]. In terms of using GP to develop scheduling rules, the bloating effect not only greatly increases the computational time, but also evolving rules tend to be too complex for decision-makers to understand, which limits their use in real-world applications [9]. Therefore, many scholars have suggested the use of simplification techniques to reduce the complexity and improve the readability of GP rules. These methods are used to remove redundant parts of evolved heuristics using manual [4] or online learning [36] techniques, thereby making heuristics more compact and easier to interpret. Similarly, feature selection methods were introduced [37] and [38] to indirectly control the growth in GP rules by selecting a subset of the most important features and eliminating noncritical features, resulting in a reduction in the size of the generated rules. The size of the individuals was also considered as a tiebreaker in other studies, such as [22] and [23], when two GP individuals deliver similar performance. The authors reported that the solution quality of their results was higher than that of the standard GP algorithm, with a significant reduction in the average size of the programs.

Accordingly, two shortcomings were noted in the bloat control methods presented in the literature. First, most bloat methods are analyzed to generate functions for the standard GP problems that question their adaptability for JSSP with a large number of job shop terminals. Second, the current bloat methods used in the JSSP domain indirectly consider the size of individuals (maximum tree depth or feature selection) to avoid sacrificing the performance. Assigning the same weight to the size and fitness of a GP rule allows low-performance rules with smaller sizes to persist across generations, which may lead to early stagnation in GP algorithm performance. Consequently, the objective of this study was to develop a multi-objective GP approach to simultaneously optimize the solution quality, diversity value, and size of the scheduling rules. The intuition behind this is that decreasing the rule length limits the search ability of the GP algorithm; thus, we seek to compensate for this by enhancing the population diversity to cover different parts of the search space.

3. Proposed Approach

This section describes the proposed multi-objective GP approach for evolving scheduling rules in JSSPs. First,

we introduce the distance metric developed to measure similarity values. The overall framework of the algorithm is presented.

3.1. Distance Metric

Because the purpose of developing this new distance metric is to promote diversity among dispatching rules, we carried out a behavior analysis for the rules that were evolved using GP. The preliminary runs enabled four observations to be made. First, the fitness evaluation of rules is computationally expensive, restricting the use of phenotypic distance measures. Second, the structure of GP individuals converges to that of the best rule in a given generation. Third, not only does the location of the nodes greatly affect their performance, but the interaction between the node and its parents also has a significant impact. Fourth, we noticed that not all nodes have the same weight, because nodes near the root had a greater impact on the fitness value of a given rule than distant nodes. These findings were addressed in our distance metric. To reduce the computational requirements, a structural metric is proposed that does not require any additional fitness evaluations. Additionally, instead of evaluating the similarity between every pair of rules in a GP population, the similarity between each rule and the best rule is evaluated, which reduces the similarity calculations from $O(n^2)$ to $O(n)$. Finally, the distance between the two rules is calculated in three steps as follows.

Step 1: The two trees i and j are brought to the same structure (depth) by adding *null* nodes.

Step 2: Weighted edge sets S_i and S_j for the two rules are generated using the following equations:

$$S_i = \{w(e_i) \forall e_i \in E_i\}, S_j = \{w(e_j) \forall e_j \in E_j\},$$

where e is an edge in the set of all edges E in a specific tree, and $w(e)$ is a function that estimates the weight of edge e.

The weight $w(e)$ of an edge e connecting nodes x and y is estimated using the following equation:

$$w(e) = \frac{w_x + w_y}{2}, \quad \ldots \quad (1)$$

where w_x and w_y represent the weights of nodes x and y respectively, which can be evaluated as follows:

$$w_x = k^{(d_{max} - d_x + 1)}, \text{ and } w_y = k^{(d_{max} - d_y + 1)},$$

where d_{max} is the tree depth, d_x and d_y denote the depth of nodes x and y. k is a constant ≥ 1 to indicate that the difference at depth k in the compared trees is k times more important than the difference at depth $k+1$.

Step 3: The similarity value S_{ij} between trees i and j are measured using the following equation:

$$S_{ij} = \frac{\sum_{e \in E_i \cap E_j} (w(e_i) + w(e_j))}{\sum_{e \in E_i \cup E_j} (w(e_i) + w(e_j))}. \quad \ldots \quad (2)$$

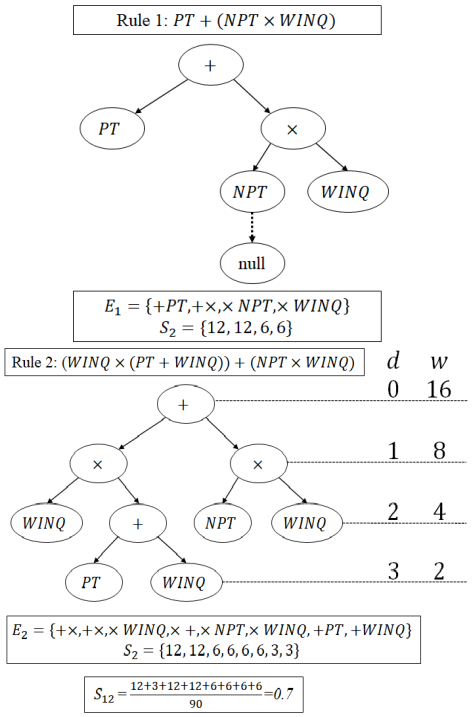

Fig. 1. Example of the proposed similarity measure.

An illustrative example of how the proposed distance metric measures the similarity between two trees is shown in **Fig. 1**. The two rules are presented in both the tree structure and mathematical form. Three terminals are presented: processing time PT, work in the next queue $WINQ$, and next processing time NPT. The depth of the first rule ($d_1 = 2$) is incremented by one using a null node to obtain the same depth as the second rule ($d_2 = 3$). The depth and weight for each node are shown in columns d and w next to the tree structure of rule 2. The weights of the nodes were assigned using $k = 2$. Clearly, the set of common edges includes $+PT$, $+\times$, $\times NPT$, and $\times WINQ$. The similarity value between the two rules is estimated by dividing the sum of the weights of the common edges by the total weight of all the edges in both rules. There is a similarity value of 0.7 between the two rules.

From this example, when $k = 2$, the weight of a node is half the weight of its parent node, that is, increasing the depth by one will reduce the edge weight by half. In addition, the location of an edge (right or left) does not affect its weight because edges at the same depth have the same weight regardless of their position. Moreover, the similarity value ranges from 0 to 1 and can be evaluated even if the two trees have different depths or dissimilar root nodes. Finally, the proposed metric utilizes the information available in the structure of the rules to be compared; thus, additional fitness evaluations are not required.

3.2. Proposed Multi-Objective GP Approach

The aim of the proposed approach is to extend the standard GP algorithm [39] by enhancing diversity and reducing the bloat effect in the evolved rules. Non-

Algorithm 1: Pseudocode of the algorithm.

Inputs: Job shop scheduling problem instance I
Outputs: The Pareto front of non-dominated rules P^*
Initialize population $P \leftarrow \{X_1, X_2, \ldots, X_{popsize}\}$
Set generation $g \leftarrow 0$
while $g < g_{max}$ **do**
  Apply genetic operators to P (generate offspring Q)
  Best rule $X_b \leftarrow 0$
  foreach rule $X_i \in Q$ **do**
    Construct a schedule $\Delta(X_i, I)$
    Calculate the objective value $f(\Delta(X_i, I))$
    if $f(\Delta(X_i, I)) < f(\Delta(X_b, I))$ **do**
      $X_b \leftarrow X_i$
    end if
  end for
  $T \leftarrow P \cup Q$
  foreach rule $X_i \in T$ **do**
    Calculate the similarity value S_{X_i, X_b}
  end for
    $P \leftarrow$ NSGA-II $select(T)$; and $g \leftarrow g + 1$
end while
return the non-dominated individuals $P^* \subseteq P$

dominated Sorting Genetic Algorithm-II (NSGA-II) is one of the most popular Pareto-based multi-objective algorithms used to obtain a set of well-spread Pareto-optimal solutions [40]. In addition, Zhang et al. [41] integrated the strategies of NSGA-II and SPEA2 into GP to solve the dynamic flexible JSSP. The results showed that incorporating NSGA-II with GP generated more effective rules with higher consistency than SPEA2. Therefore, we integrated NSGA-II with the GP algorithm to simultaneously optimize the three objectives at the same time. The objectives considered were the fitness value, similarity value, and rule depth. The pseudocode for the proposed algorithm is presented in **Algorithm 1**.

The algorithm begins by generating a population of dispatching rules using a predefined set of functions and terminals. Evolutionary generation begins by using each of these rules to construct a schedule for a given JSSP instance. The objective values of the rules are then estimated from the constructed schedules, and the best rule in this generation is updated accordingly. Parents and offspring are combined to ensure elitism, which guarantees that the best individuals secure a place in the next generation. Then, the similarity value between each rule and the best-evolved rule is assessed using the proposed distance metric. NSGA-II is used to select rules with high ranks and smaller crowding distances for new population formation. If the required number of generations is not reached, another iterative cycle starts; otherwise, the algorithm terminates and a set of non-dominated scheduling rules is returned.

4. Experimental Setup

This section presents the experimental setup used to compare the proposed approach with the other methods.

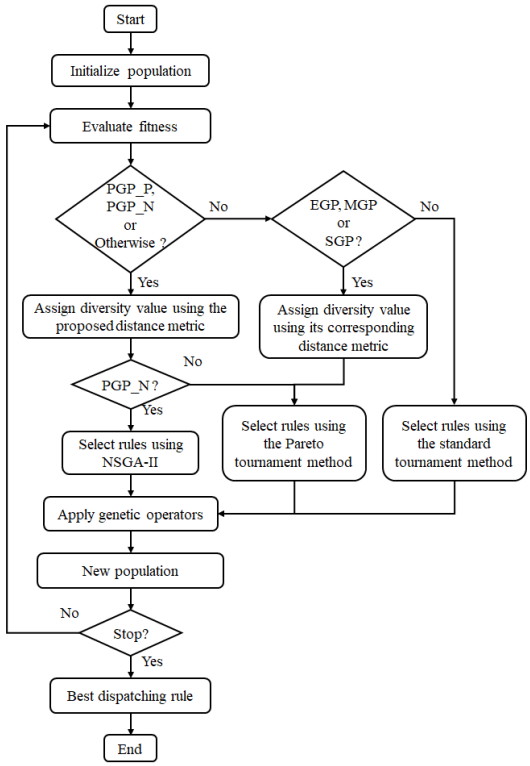

Fig. 2. Flowchart of the five developed algorithms.

In addition, the parameter settings for both the GP algorithms and JSSP instances are shown.

4.1. Comparison Design

To assess the effectiveness of the proposed distance metric and multi-objective GP approach, two algorithms are developed. The first algorithm, referred to as "PGP_P," evaluates the ability of the proposed similarity metric to increase the quality of GP rules by promoting population diversity. The second algorithm, denoted as "PGP_N," follows the same steps presented in **Algorithm 1** and is used to verify whether the proposed approach can optimize the solution quality, diversity, and size of evolved rules simultaneously. The two proposed algorithms were compared with the following three algorithms in the literature.

1. SGP: Standard GP algorithm [39].
2. EGP: SGP with Edit-distance metric [23].
3. MGP: SGP with genetic Marker metric [22].

The SGP algorithm was developed to evaluate the benefits of increasing the population diversity and controlling the bloating effect compared to the standard version of the GP algorithm. In addition, the EGP and MGP algorithms were considered to enable the proposed algorithm to be compared with other GP algorithms that use different distance metrics and bloat control techniques.

A flowchart of the five algorithmic experiments is shown in **Fig. 2**. All algorithms begin by initializing the

same population of scheduling rules using the same random seed. Subsequently, their fitness values were evaluated on a specific JSSP instance using a predefined objective function. In the case of SGP, no diversity metric is used, and the best rules are selected by the standard tournament method. For the EGP and MGP algorithms, the similarity values of the rules were evaluated using the edit distance and genetic marker distance metrics, respectively. The proposed distance metric was used for both PGP_P and PGP_N. For the PGP_N algorithm, the Pareto front rule with the lowest fitness value was used to measure the similarity values. The Pareto tournament method [12] was employed to select the fittest rules for the EGP, MGP, and PGP_P algorithms. The same selection mechanism was used to evaluate the performance of the proposed metric separately. In the case of PGP_N, the best rules were selected using NSGA-II. Genetic operators were then applied to create a new population of scheduling rules. If the stopping criterion is satisfied, the algorithm terminates; otherwise, a new evolutionary iteration begins.

4.2. JSSP and Genetic Programming Settings

Ten JSSP instances (ta61–ta70) based on previously proposed benchmark data [42] were used for comparison purposes. These instances are chosen from the 80 available instances because they are the most complex instances with the largest number of jobs and machines, and the optimal solutions are known for the makespan objective (for validation). The job shop contains 20 machines, and 50 jobs are processed. Each job must visit all machines following a predetermined routing path. The processing times follow a uniform distribution $U[1,99]$. Two objectives were investigated: makespan and mean tardiness. These objectives are chosen because the makespan represents the level of productivity that the system can achieve, whereas the mean tardiness objective assesses its ability to meet customer due dates (customer satisfaction level). Job due dates are estimated using the total work content method [10], where job due date = current time + tightness factor × total processing time, as these dates are not available in the instances reported in the literature. We employed a tightness factor of 1.9 as proposed in a previous report [7].

Regarding the GP parameters used for the five algorithms, a population of 250 rules was created using the ramped-half-and-half method [39], with a maximum depth of 8. **Table 1** lists the set of functions and terminals used. In addition, the underlined terminals are used when optimizing the mean tardiness objective, and are excluded in the case of the makespan objective. The function set comprises basic arithmetic operations and logical functions. The values of the crossover and mutation rates are set to 0.9 and 0.1. Finally, the algorithm terminates after 30 generations are completed. Because of the randomness inherent in the GP algorithm, 20 replications were performed. In addition, the Wilcoxon rank-sum test was used with a significance level of 5%.

Table 1. GP terminal and function sets.

Attribute	Explanation
JR	Job release date
OR	Operation ready time
WR	Work remaining of the job
PT	Operation processing time
RO	Number of remaining operations in a job
WT	Operation waiting time
NPT	Processing time of the next operation
WINQ	Work in the next queue
APR	Average processing time of queued job
DD	Job due date
CT	Machine ready time (current time)
SL	Job slack
Function set	$+$, $-$, $\times$, $/$, *min*, *max*, and *abs*

5. Results

This section discusses the evaluation of the two proposed algorithms with respect to the three approaches. In Section 5.1, parameter k is fine-tuned by analyzing its impact on the performance of the proposed similarity metric. To gain in-depth insights into the behavior of the five algorithms, the quality of the GP individuals was tracked across generations. Additionally, the results obtained by applying the five algorithms to the ten job shop scenarios are presented in Subsections 5.2 and 5.3 for the makespan and mean tardiness objectives, respectively.

5.1. Behavior Analysis of the Proposed Methods

The PGP_P algorithm is implemented using four values of the parameter k, $k = \{1, 2, 3, 4\}$, where $k = 1$ represents the case in which a node takes a weight equal to its depth. In addition, four performance measures were considered:

1. Solution quality: average objective values of rules in a given generation.
2. Mean rule length: the average number of terminals included in GP individuals.
3. Genotypic diversity: the number of unique rules. Two rules are identical if they have the same structure and content [14].
4. Phenotypic diversity: the number of unique fitness values in a population.

The experiments in this subsection were performed in the ta61 instance, as similar results were obtained in the other nine instances. In addition, the aim of this study was to investigate the performance of the algorithms across generations rather than their overall performance, as presented in the next subsections. **Fig. 3** shows the results obtained by implementing the PGP_P algorithm using the four values of the parameter k, where the mean values are

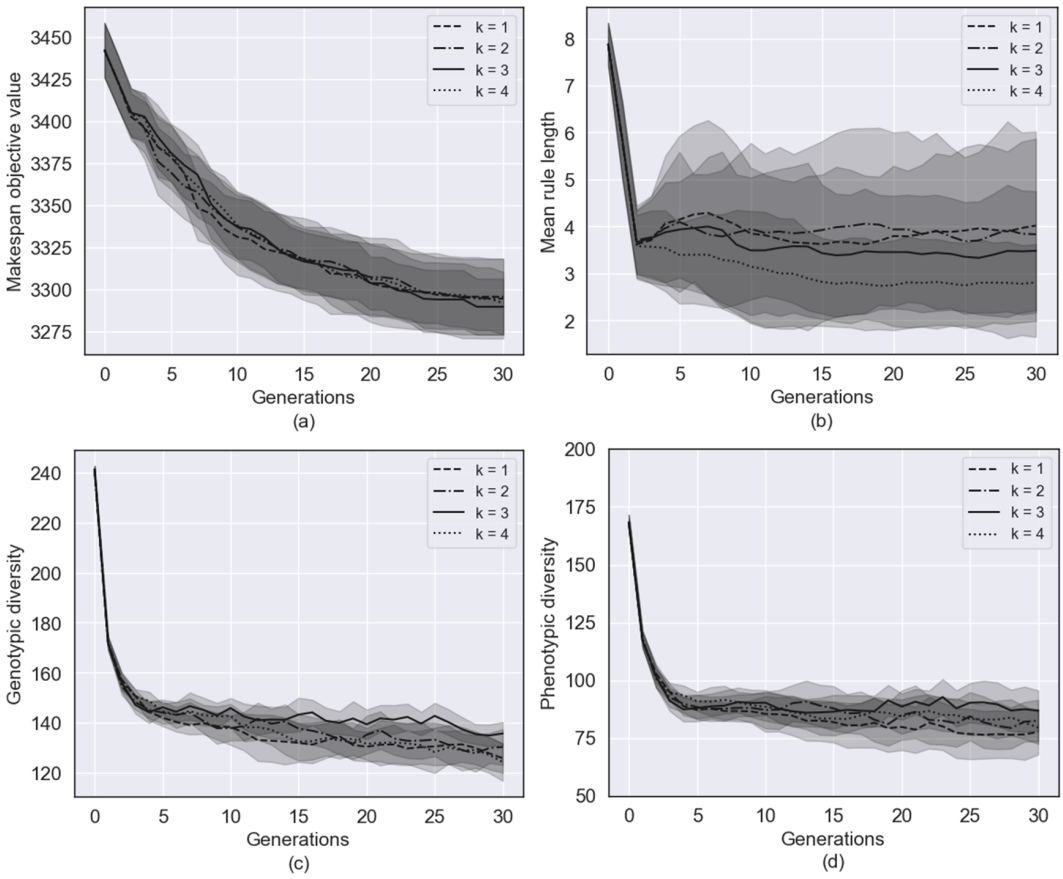

Fig. 3. Effect of changing the value of the parameter k on the four performance measures.

shown as a solid line, and the standard deviations are depicted as a shaded area around it. As shown in **Fig. 3(a)**, changing the value of the parameter k does not affect the solution quality of the evolved rules, as there is no significant difference between the four runs. In contrast, the average rule length is highly sensitive to the k value, as shown in **Fig. 3(b)**. Similar observations are revealed for the other two measures, as shown in **Figs. 3(c)** and **(d)**, where altering the k value had a great impact on the population diversity. The interesting finding here is that even though the population diversity changes according to the value of parameter k, the GP reasoning ability can mitigate this effect on the quality of the generated rules (similar makespan results). Specifically, when the value of k is 4, if the node depth is decreased by one, its weight increases fourfold. Thus, the GP search explores the search space of small rules (higher impact on diversity), resulting in a reduction in the length of new rules, as shown in **Figs. 3(a)** and **(b)**, where $k = 4$ has the smallest average rule length with the same fitness values. The distance metric with $k = 3$ was used in the following GP experiments because it yielded acceptable results for the four performance measures.

In **Fig. 4**, the behavior of the two proposed algorithms is analyzed against that of the three algorithms from the literature on the ta61 instance with respect to the quality of evolved rules, rule length, computational time, and population diversity. Regarding the quality of the evolved rules, the PGP_N algorithm obtained the lowest makespan value with a consistent improvement across generations, as shown in **Fig. 4(a)**. In addition, the exploration ability of PGP_P was superior to that of the EGP and MGP algorithms. It is noteworthy that although the SGP algorithm obtained higher quality makespan results compared to the EGP, MGP, and PGP_P algorithms, these results would be reversed if the computational time was used as a stopping condition instead of the number of generations, as illustrated in **Fig. 4(c)**. The use of the Pareto tournament selection method in the EGP, MGP, and PGP_P algorithms significantly decreased the average rule size by approximately 75% compared with the SGP algorithm, resulting in a reduction in computational time of approximately 56.5%, as shown in **Figs. 4(b)** and **(c)**. Therefore, when comparing the other JSSP instances in the next subsection, the average rule length was used as an indicator of the computational time of the algorithm.

Regarding the diversity of GP individuals, the proposed PGP_N algorithm obtained higher genotypic and phenotypic diversity values across generations compared to other methods, especially in the first ten generations as depicted in **Figs. 4(d)** and **(e)**. In addition, although the SGP algorithm generated high-quality rules, it had the lowest genetic and phenotypic diversity, which explains the exponential growth in the mean rule length as SGP

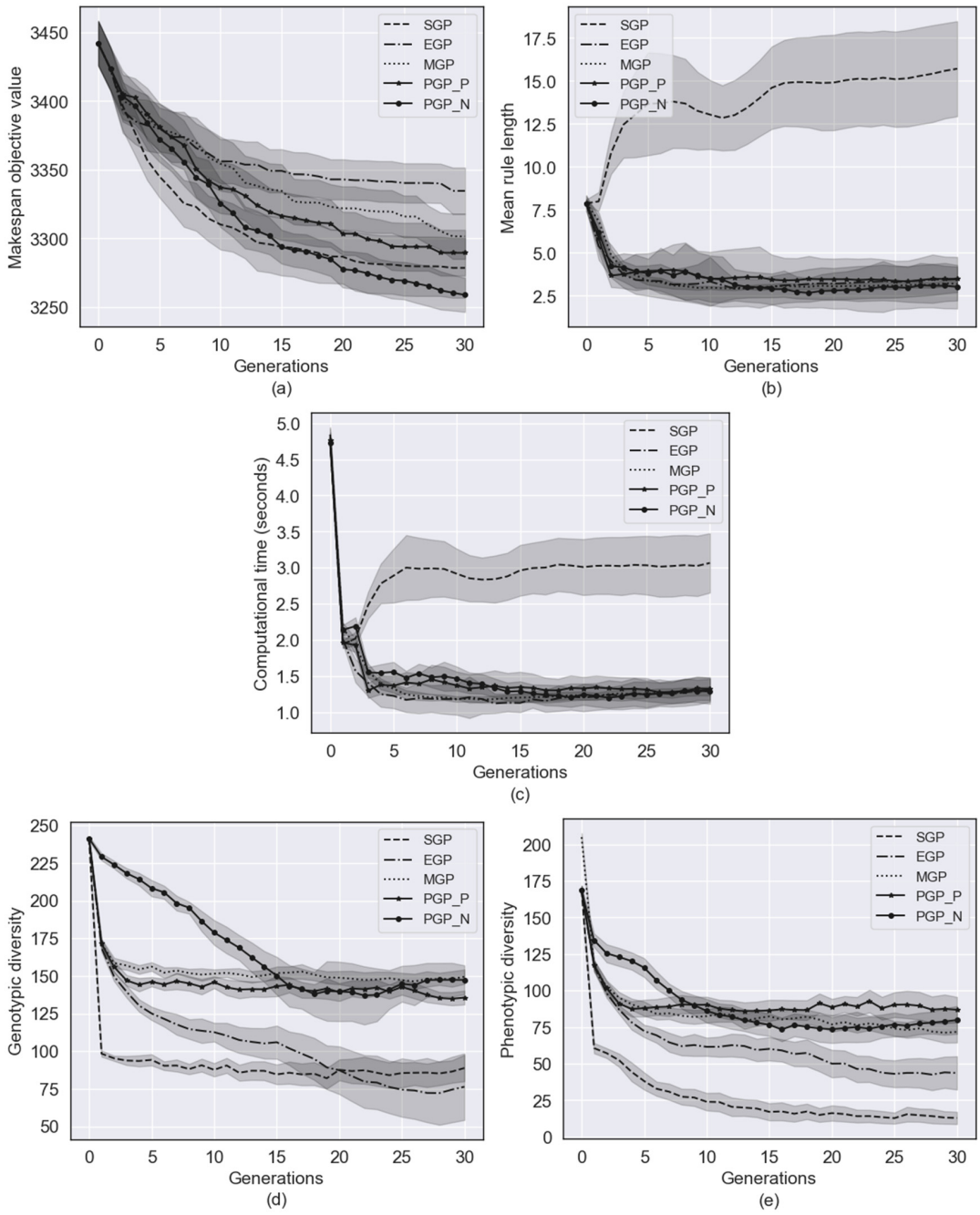

Fig. 4. Performance of the five GP algorithms on the ta61 JSSP instance.

strives to escape from local optima by increasing the rule sizes. Moreover, the use of the proposed distance metric increases the phenotypic diversity of GP rules compared with the literature similarity measures without sacrificing the performance or size of the rules. These results led to two main findings. First, the PGP_N algorithm was able to generate rules with lower makespan values compared to the other methods because of the high diversity levels, especially in the early generations. Second, the use of NSGA-II as a section mechanism enhanced the exploration ability of the GP algorithm compared with the standard or Pareto tournament selection methods used in the other algorithms. Finally, the PGP_N algorithm obtained a mean makespan value of 3315 with a 7.2% deviation from the optimal solution (2868) of this JSSP instance, which proves that GP is a promising approach for the automated design of scheduling rules.

5.2. Makespan Objective

The performances of the five GP algorithms for optimizing the makespan objective were estimated. Ten instances from the literature were used and four performance measures were evaluated. These measures were makespan value (MV), average rule length (RL), genotypic diversity (GD), and phenotypic diversity (PD). The mean and standard deviation of the obtained results are

Table 2. Performance of the five GP algorithms in terms of optimizing the makespan objective on the ten JSSP instances.

Inst.	Perf.	SGP	EGP	MGP	PGP_P	PGP_N
ta61	MV	3078.15±16.4	3152.5±12.5	3129.55±17.2	3102.6±11.6 (-, +, +)	3077.0±13.0 (=, +, +, +)
	RL	14.09±2.6	3.73±1.1	3.38±0.2	3.3±0.8 (+, =, +)	3.19±0.7 (+, +, =, +)
	GD	89.82±6.3	103.33±13.3	155.82±3.4	145.68±5.0 (+, +, -)	180.59±8.3 (+, +, +, +)
	PD	32.08±5.5	62.47±7.7	93.21±5.1	99.1±5.9 (+, +, +)	103.9±6.6 (+, +, +, =)
ta62	MV	3133.15±13.8	3188.75±28.0	3158.2±14.8	3142.3±7.8 (-, +, +)	3135.2±7.5 (=, +, +, =)
	RL	11.61±2.2	3.07±0.6	3.15±0.2	3.08±1.0 (+, =, +)	2.79±1.1 (+, =, =, +)
	GD	96.47±6.7	78.39±13.9	149.74±4.2	137.44±6.0 (+, +, -)	164.35±11.4 (+, +, +, +)
	PD	25.35±4.0	41.48±7.1	74.75±4.8	84.34±6.6 (+, +, +)	79.28±9.7 (+, +, =, -)
ta63	MV	2960.9±10.3	3003.4±17.7	2984.65±8.8	2967.45±11.6 (=, +, +)	2956.45±10.1 (=, +, +, =)
	RL	13.9±2.5	3.48±1.3	3.42±0.2	3.52±1.2 (+, =, -)	3.16±0.9 (+, =, +, +)
	GD	95.28±6.5	101.24±10.7	156.87±3.3	143.83±4.6 (+, +, -)	177.07±9.6 (+, +, +, +)
	PD	25.68±4.6	57.09±6.7	85.27±4.7	88.1±6.7 (+, +, +)	86.99±6.8 (+, +, =, =)
ta64	MV	2857.75±18.2	2912.5±15.0	2885.3±7.2	2861.2±9.5 (=, +, +)	2853.5±7.0 (=, +, +, =)
	RL	13.85±3.0	3.16±0.7	3.11±0.2	3.19±1.4 (+, -, -)	2.9±1.0 (+, =, =, +)
	GD	90.36±6.5	98.81±11.7	156.95±3.1	142.15±5.4 (+, +, -)	178.88±9.6 (+, +, +, +)
	PD	25.95±4.7	42.52±6.5	65.93±4.7	76.51±6.9 (+, +, +)	79.68±8.7 (+, +, +, =)
ta65	MV	2988.9±21.0	3035.6±13.3	3024.85±17.9	2987.65±16.8 (=, +, +)	2994.85±14.1 (=, +, +, =)
	RL	12.99±2.8	3.2±0.9	2.86±0.2	2.92±1.2 (+, =, -)	2.56±1.1 (+, +, =, +)
	GD	86.92±7.8	75.58±15.3	147.65±3.9	132.8±13.7 (+, +, -)	140.17±17.0 (+, +, -, =)
	PD	26.51±4.7	36.62±7.2	56.46±4.7	77.37±8.6 (+, +, +)	65.22±9.6 (+, +, +, -)
ta66	MV	3042.15±11.4	3099.25±10.4	3075.85±9.2	3056.95±7.7 (-, +, +)	3050.0±5.8 (=, +, +, =)
	RL	14.08±3.4	3.58±0.9	3.44±0.2	3.31±1.0 (+, =, +)	3.22±1.2 (+, +, +, +)
	GD	88.61±6.7	102.36±12.8	155.44±3.7	143.47±4.9 (+, +, -)	173.65±9.1 (+, +, =, +)
	PD	28.48±4.4	60.68±8.2	94.49±4.8	95.21±5.3 (+, +, +)	95.21±6.9 (+, +, =, =)
ta67	MV	3024.95±19.8	3108.25±18.5	3084.15±16.2	3040.95±10.1 (=, +, +)	3039.6±11.6 (=, +, +, =)
	RL	13.87±2.2	3.35±0.7	3.51±0.2	3.68±1.3 (+, -, -)	3.49±1.2 (+, -, +, +)
	GD	91.13±5.5	109.15±9.6	157.55±3.1	143.85±4.7 (+, +, -)	180.33±7.0 (+, +, +, +)
	PD	32.65±5.7	66.22±6.1	96.73±4.5	96.25±5.7 (+, +, =)	104.62±6.2 (+, +, =, =)
ta68	MV	2922.95±11.9	2985.15±15.8	2946.65±8.9	2936.35±14.6 (=, +, +)	2922.1±10.9 (=, +, +, =)
	RL	11.57±2.0	3.24±0.7	3.4±0.2	3.21±0.8 (+, +, =)	2.76±0.6 (+, +, +, +)
	GD	94.16±7.3	91.48±13.9	153.12±3.5	143.29±5.0 (+, +, -)	166.0±12.3 (+, +, =, +)
	PD	25.19±4.7	48.27±8.1	83.4±4.4	85.65±7.1 (+, +, +)	79.62±7.9 (+, +, =, =)
ta69	MV	3211.35±6.6	3275.55±14.0	3268.9±11.7	3234.3±12.3 (-, +, +)	3221.15±9.0 (-, +, +, =)
	RL	13.92±3.5	4.62±2.0	3.28±0.2	3.23±1.1 (+, +, +)	3.11±0.8 (+, +, +, +)
	GD	91.14±5.6	111.56±10.2	155.88±3.5	140.8±7.4 (+, +, -)	183.13±8.1 (+, +, +, +)
	PD	28.12±4.8	54.83±6.7	74.94±5.3	87.23±6.0 (+, +, +)	94.44±6.8 (+, +, +, +)
ta70	MV	3278.55±21.8	3334.65±16.9	3301.55±16.5	3289.55±16.7 (=, +, =)	3258.85±12.4 (=, +, +, +)
	RL	13.82±2.4	3.51±0.8	3.46±0.2	3.59±1.3 (+, -, -)	3.29±1.1 (+, =, =, +)
	GD	93.14±6.0	106.65±11.6	154.47±3.9	146.4±4.7 (+, +, -)	169.31±10.2 (+, +, =, +)
	PD	28.55±5.0	63.56±7.5	86.05±4.9	92.52±6.5 (+, +, +)	90.38±7.5 (+, +, =, -)
Summary	MV	(0/4, 0/1)	(10/0, 10/0)	(9/0, 10/0)	2/0	None
	RL	(8/1, 8/1)	(3/2, 6/1)	(4/4, 6/0)	9/0	
	GD	(10/0, 10/0)	(10/0, 10/0)	(0/10, 6/1)	9/0	
	PD	(10/0, 10/0)	(10/0, 10/0)	(9/0, 4/0)	1/3	

listed in **Table 2**. The statistical results achieved by comparing PGP_P with the three algorithms are represented by the tuple next to the PGP_P results (PGP_P versus SGP, PGP_P versus EGP, and PGP_P versus MGP). The symbols "+," "−," and "=" indicate that the corresponding result is significantly better, worse than, or similar to its counterpart, respectively. In addition, the performance of PGP_N is compared with that of the three literature methods, as well as the PGP_P algorithm represented by the fourth element in the tuples next to its results. The last row of the table summarizes the results counting the number of times a certain method loses (significantly worse) or wins (significantly better) against the PGP_P and PGP_N algorithms (lose/win).

Regarding the quality of the evolved rules, the PGP_N algorithm delivered similar performance to the SGP with only one loss, whereas the PGP_P produced inferior results in four instances. In contrast, the proposed algorithms significantly outperformed the SGP algorithm in terms of the average size of the rules and diversity objectives. The rules evolved using the proposed methods were able to achieve smaller makespan values compared with the EGP and MGP algorithms with wins in almost all instances. In addition, the population diversity of the PGP_P and PGP_N rules was significantly larger than that of the EGP rules in the ten instances. In contrast, although the phenotypic diversity of the PGP_P algorithm was greater than that of the MGP algorithm in the nine scenarios, the genotypic diversity was significantly worse in all instances. These results are consistent with those in the literature, as good fitness values were reported to be correlated with high phenotypic diversity. The integration

Table 3. Performance of the five GP algorithms in terms of optimizing the mean tardiness objective on the ten JSSP instances.

Inst.	Perf.	SGP	EGP	MGP	PGP_P	PGP_N
ta61	MT	492.14±5.0	514.54±5.4	507.98±5.5	498.37±4.8 (=, +, +)	494.0±3.4 (=, +, +, =)
	RL	14.56±3.6	4.62±1.4	4.1±0.2	3.71±0.7 (+, +, +)	3.59±0.8 (+, +, +, +)
	GD	94.25±5.5	113.1±12.5	155.83±3.1	142.63±5.0 (+, +, -)	168.69±9.6 (+, +, =, +)
	PD	25.27±3.6	74.33±9.6	97.28±4.6	103.32±6.4 (+, +, =)	100.04±5.5 (+, +, =, -)
ta62	MT	463.19±7.6	485.07±4.9	478.94±4.3	470.15±4.3 (=, +, +)	466.72±4.2 (=, +, +, =)
	RL	12.21±1.8	4.08±1.0	4.17±0.2	4.11±1.0 (+, -, +)	3.68±0.7 (+, +, +, +)
	GD	89.34±6.4	117.38±10.4	158.15±3.3	145.08±5.0 (+, +, -)	182.13±8.1 (+, +, +, +)
	PD	27.96±4.4	88.5±8.0	112.89±4.9	121.24±6.9 (+, +, +)	128.05±7.3 (+, +, =, =)
ta63	MT	461.61±4.0	486.55±5.5	475.17±3.1	468.75±4.7 (-, +, +)	466.37±4.1 (-, +, +, =)
	RL	13.57±2.3	4.11±1.4	4.18±0.2	3.9±0.9 (+, =, =)	3.99±1.0 (+, =, =, =)
	GD	91.78±7.5	106.42±11.4	157.02±3.5	143.86±5.0 (+, +, -)	179.58±8.8 (+, +, +, +)
	PD	26.65±4.0	79.03±8.7	105.7±4.7	116.22±6.3 (+, +, +)	125.12±7.6 (+, +, =, =)
ta64	MT	455.9±7.3	479.64±7.0	467.02±6.5	457.33±6.0 (=, +, +)	447.49±7.0 (=, +, +, +)
	RL	14.13±3.3	3.48±0.6	4.11±0.2	3.83±1.0 (+, -, =)	3.88±1.1 (+, -, =, =)
	GD	95.07±6.6	99.75±11.5	157.64±3.1	145.97±4.8 (+, +, -)	175.15±8.7 (+, +, +, +)
	PD	24.27±4.2	68.86±7.2	94.83±5.1	115.77±6.6 (+, +, +)	112.81±6.1 (+, +, +, -)
ta65	MT	443.85±5.9	459.51±5.9	454.9±4.8	449.91±3.7 (-, +, =)	446.21±4.9 (=, +, +, =)
	RL	13.34±2.1	3.8±0.7	4.09±0.2	3.67±0.8 (+, +, +)	3.51±0.5 (+, +, +, +)
	GD	92.07±6.1	114.4±9.5	157.6±3.1	144.45±4.7 (+, +, -)	179.33±9.8 (+, +, +, +)
	PD	25.05±3.5	81.4±7.6	102.43±5.3	116.4±6.0 (+, +, +)	123.65±7.7 (+, +, =, =)
ta66	MT	470.19±7.4	499.12±6.2	490.54±4.9	476.14±6.0 (=, +, +)	475.51±6.0 (=, +, +, =)
	RL	13.97±2.8	3.73±1.0	3.91±0.2	3.53±1.0 (+, =, +)	3.16±0.8 (+, +, +, +)
	GD	90.77±7.1	96.24±10.0	153.67±3.6	141.62±5.9 (+, +, -)	143.73±13.2 (+, +, =, =)
	PD	24.28±3.5	62.96±7.0	87.22±4.4	110.23±6.6 (+, +, +)	90.11±7.4 (+, +, =, -)
ta67	MT	488.62±6.6	515.9±7.1	502.96±3.9	494.27±3.3 (=, +, +)	491.38±4.3 (=, +, +, =)
	RL	15.24±4.1	3.73±0.7	4.19±0.2	4.15±1.3 (+, -, +)	3.72±1.0 (+, =, +, +)
	GD	90.96±6.8	107.05±9.8	155.82±3.5	144.88±4.9 (+, +, -)	168.02±9.0 (+, +, =, +)
	PD	27.3±4.0	79.58±7.7	97.8±5.4	119.75±6.3 (+, +, +)	114.54±7.1 (+, +, =, -)
ta68	MT	476.26±7.6	496.38±6.5	491.58±4.5	481.2±6.6 (=, +, +)	475.53±5.8 (=, +, +, =)
	RL	14.24±3.1	3.85±0.8	4.08±0.2	3.82±1.1 (+, +, =)	4.0±1.1 (+, -, =, =)
	GD	91.7±6.9	103.03±12.6	155.09±4.0	144.81±4.5 (+, +, -)	173.99±8.6 (+, +, +, +)
	PD	28.31±3.9	72.94±8.9	99.39±6.0	120.84±6.9 (+, +, +)	123.42±7.3 (+, +, +, =)
ta69	MT	523.12±8.2	546.72±5.6	538.03±5.6	527.85±7.4 (=, +, +)	517.62±4.3 (=, +, +, +)
	RL	11.62±1.7	3.4±0.5	3.98±0.2	3.56±0.8 (+, -, +)	3.46±0.9 (+, -, +, +)
	GD	92.87±5.6	96.1±10.3	155.46±3.1	142.01±5.0 (+, +, -)	163.64±11.9 (+, +, =, +)
	PD	23.76±3.0	65.77±6.5	87.13±6.4	101.75±7.5 (+, +, +)	99.83±5.7 (+, +, =, -)
ta70	MT	505.36±6.0	526.77±5.5	521.24±3.4	514.11±6.3 (-, +, +)	508.56±5.1 (=, +, +, =)
	RL	12.92±2.1	4.11±0.9	4.15±0.2	3.79±0.9 (+, =, =)	3.39±0.6 (+, +, +, +)
	GD	98.77±7.2	110.45±13.1	158.48±3.0	141.31±5.2 (+, +, -)	181.76±8.7 (+, +, +, +)
	PD	23.54±3.8	71.94±9.5	99.15±5.1	107.57±7.7 (+, +, +)	113.43±7.3 (+, +, =, =)
Summary	MT	(0/3, 0/1)	(10/0, 10/0)	(9/0, 10/0)	2/0	None
	RL	(10/0, 10/0)	(3/4, 5/3)	(6/0, 7/0)	7/0	
	GD	(10/0, 10/0)	(10/0, 10/0)	(0/10, 6/0)	9/1	
	PD	(10/0, 10/0)	(10/0, 10/0)	(9/0, 2/0)	0/5	

of NSGA-II and the proposed distance metric greatly increased the genotypic diversity and reduced the average length of rules compared with the PGP_P algorithm.

The following conclusions were drawn. The edit distance method used in the EGP algorithm is clearly unsuitable for measuring the similarity between GP rules in JSSPs because it produced the worst results for the GD and PD measures. In contrast, the genetic marker diversity measure used in the MGP algorithm achieved high levels of genotypic and phenotypic diversity among the GP rules. Finally, the rules evolved using the PGP_N algorithm have the lowest makespan values, highest diversity, and smallest average rule length, which demonstrates the usefulness of integrating NSGA-II with the GP algorithm while using the proposed distance metric.

5.3. Mean Tardiness Objective

The five algorithms were evaluated in terms of their ability to optimize the mean tardiness (MT) objective, which evaluates the ability of the job shop to meet customer due dates. The results are presented in **Table 3**. The performance of the PGP_P and PGP_N algorithms was superior to that of the SGP algorithm with respect to the RL, GD, and PD objectives. Regarding the MT objective, both SGP and PGP_N had relatively similar performances, except for one instance in which SGP outperformed PGP_N. Compared with the EGP algorithm, the proposed algorithms achieved results that are of a significantly higher quality for the MT, GD, and PD objectives in all considered instances. The conclusion regarding the average length of the evolved rules is not definitive because

the PGP_P algorithm delivered superior performance in three instances and inferior performance in four instances compared to the EGP algorithm.

The results in **Table 3** indicate that the proposed algorithms were able to create rules with higher solution quality and smaller sizes than the MGP algorithm. Specifically, PGP_P and PGP_N have significantly lower mean tardiness values than the MGP algorithm in nine and ten instances, respectively. In terms of the phenotypic diversity of the evolved rules, both PGP_P and PGP_N outperformed MGP in nine and two instances, respectively, without any loss. In contrast, the PGP_P algorithm yielded poor GD results compared with the MGP results for the ten scenarios. In addition, it is important to note that high genotypic diversity does not necessarily imply high phenotypic diversity because rules with different structures and content may obtain the same fitness values. The main reason for this phenomenon is the presence of redundant operations, for example, even though the $WINQ + PT$ rule and the $WINQ + ((PT \times NPT)/NPT))$ rule have different structures and terminals, they will return the same results. Finally, the rules generated using the proposed multi-objective approach had higher solution quality in two instances, shorter lengths in seven instances, and higher genotypic diversity in nine instances compared with the PGP_P algorithm.

Although the results obtained using the mean tardiness objective were consistent with those obtained using the makespan objective, two additional observations were noted. First, in most scenarios, the average rule length in the case of the mean tardiness objective is greater than the average length of the rules generated for the makespan objective. Second, the PD measure in the SGP algorithm did not change significantly between the two objectives; in spite of this, the other algorithms had higher PD levels in most instances in the case of the mean tardiness objective compared with the makespan objective. Increasing the number of specified terminals related to job due dates by three in the case of the mean tardiness objective might be the reason for the increase in the average rule length and phenotypic diversity. When the number of terminals increases, the heuristic search space increases exponentially; thus, the GP algorithm increases the length of the generated rules to improve the exploration ability. In addition, as the heuristic search space and average length of rules increase, distance metrics become more efficient because the possibility of variability between the structure and fitness value of evolving rules increases.

6. Conclusions

We propose a distance metric to promote diversity among the scheduling heuristics evolved using a genetic programming algorithm. In addition, to mitigate the bloating effect, the proposed metric was integrated with NSGA-II to optimize the solution quality, diversity value, and rule length simultaneously. Two algorithms, PGP_P and PGP_N, were developed to assess the effectiveness of the proposed distance metric and multi-objective GP approaches. Two objective functions were addressed: the makespan and mean tardiness. For each objective, four performance measures were evaluated: the objective value, genotypic diversity, phenotypic diversity, and average length of the evolved rules.

The performance of the two proposed algorithms was compared with that of three algorithms from the literature for ten benchmark job shop scheduling problem instances. The experimental results demonstrated the effectiveness of our proposed methods in generating a phenotypically diverse population of scheduling rules with smaller sizes and higher solution quality compared to other methods. Although one of the existing diversity metrics (MGP) obtained higher quality genotypic diversity results compared to the PGP_P algorithm, the evolved rules using PGP_P significantly outperformed the MGP rules in terms of the other three objectives. Finally, the rules generated using the PGP_N algorithm produced superior results compared to the literature approaches for the studied objectives.

In future research, we plan to employ the proposed approaches to address dynamic events in job shops such as machine breakdowns and job arrivals. In addition, we aim to assess the developed algorithms for other job shop settings, such as flexible job shops or unrelated machines environments.

References:

[1] H. Lasi, P. Fettke, H.-G. Kemper, T. Feld, and M. Hoffmann, "Industry 4.0," Bus. Inf. Syst. Eng., Vol.6, No.4, pp. 239-242, doi: 10.1007/s12599-014-0334-4, 2014.

[2] S. Salama and A. B. Eltawil, "A Decision Support System Architecture Based on Simulation Optimization for Cyber-Physical Systems," Procedia Manufacturing, Vol.26, pp. 1147-1158, doi: 10.1016/j.promfg.2018.07.151, 2018.

[3] Z. Shi, Y. Xie, W. Xue, Y. Chen, L. Fu, and X. Xu, "Smart factory in Industry 4.0," Systems Research and Behavioral Science, Vol.37, No.4, pp. 607-617, doi: 10.1002/sres.2704, 2020.

[4] J. C. Tay and N. B. Ho, "Evolving dispatching rules using genetic programming for solving multi-objective flexible job-shop problems," Computers and Industrial Engineering, Vol.54, No.3, pp. 453-473, doi: 10.1016/j.cie.2007.08.008, 2008.

[5] A. Ishigaki and Y. Matsui, "Effective Neighborhood Generation Method in Search Algorithm for Flexible Job Shop Scheduling Problem," Int. J. Automation Technol., Vol.13, No.3, pp. 389-396, doi: 10.20965/ijat.2019.p0389, 2019.

[6] M. L. Pinedo, "Scheduling: Theory, Algorithms, and Systems," 4th ed., Springer-Verlag, 2012.

[7] V. Sels, N. Gheysen, and M. Vanhoucke, "A comparison of priority rules for the job shop scheduling problem under different flow time- and tardiness-related objective functions," Int. J. of Production Research, Vol.50, No.15, pp. 4255-4270, doi: 10.1080/00207543.2011.611539, 2012.

[8] J. H. Drake, A. Kheiri, E. Özcan, and E. K. Burke, "Recent advances in selection hyper-heuristics," European J. of Operational Research, Vol.285, No.2, pp. 405-428, 2020.

[9] J. Branke, S. Nguyen, C. W. Pickardt, and M. Zhang, "Automated Design of Production Scheduling Heuristics: A Review," IEEE Trans. on Evolutionary Computation, Vol.20, No.1, pp. 110-124, doi: 10.1109/TEVC.2015.2429314, 2016.

[10] S. Nguyen, Y. Mei, and M. Zhang, "Genetic programming for production scheduling: a survey with a unified framework," Complex & Intelligent Systems, Vol.3, No.1, pp. 41-66, doi: 10.1007/s40747-017-0036-x, 2017.

[11] S. Salama, T. Kaihara, N. Fujii, and D. Kokuryo, "Automatic Design of Dispatching Rules with Genetic Programming for Dynamic Job Shop Scheduling," Proc. of the IFIP Int. Conf. on Advances in Production Management Systems, Vol.591, pp. 399-407, doi: 10.1007/978-3-030-57993-7_45, 2020.

[12] M. D. Schmidt and H. Lipson, "Age-Fitness Pareto Optimization," Genetic Programming Theory and Practice VIII: 12th Annual Conf. on Genetic and Evolutionary Computation Proc., pp. 129-146, doi: 10.1007/978-1-4419-7747-2_8, 2011.

[13] Y. Sakakura, N. Taniguchi, Y. Hoshino, and K. Kamei, "Maintaining Individual Diversity by Fuzzy c-Means Selection," J. Adv. Comput. Intell. Intell. Inform., Vol.11, No.8, pp. 884-890, 2007.

[14] E. K. Burke, S. Gustafson, and G. Kendall, "Diversity in genetic programming: an analysis of measures and correlation with fitness," IEEE Trans. on Evolutionary Computation, Vol.8, No.1, pp. 47-62, doi: 10.1109/TEVC.2003.819263, 2004.

[15] C. E. Berbague, N. E. Karabadji, H. Seridi, P. Symeonidis, Y. Manolopoulos, and W. Dhifli, "An overlapping clustering approach for precision, diversity and novelty-aware recommendations," Expert Systems with Applications, Vol.177, No.4, 114917, 2021.

[16] D. Whitley, S. Rana, and R. B. Heckendorn, "The Island Model Genetic Algorithm: On Separability, Population Size and Convergence," J. of Computing and Information Technology, Vol.7, pp. 33-47, 1998.

[17] E. K. Burke, S. M. Gustafson, G. Kendall, and N. Krasnogor, "Advanced Population Diversity Measures in Genetic Programming," Parallel Problem Solving from Nature – PPSN VII: 7th Int. Conf. on Parallel Problem Solving from Nature Proc., pp. 341-350, doi: 10.1007/3-540-45712-7_33, 2002.

[18] S. Luke and L. Panait, "A Comparison of Bloat Control Methods for Genetic Programming," Evolutionary Computation, Vol.14, No.3, pp. 309-344, doi: 10.1162/evco.2006.14.3.309, 2006.

[19] N. Mori, B. McKay, N. X. Hoai, D. Essam, and S. Takeuchi, "A New Method for Simplifying Algebraic Expressions in Genetic Programming called Equivalent Decision Simplification," SCIS & ISIS, Vol.2008, pp. 1671-1676, doi: 10.14864/softscis.2008.0.1671.0, 2008.

[20] E. F. Crane and N. F. McPhee, "The Effects of Size and Depth Limits on Tree Based Genetic Programming," T. Yu, R. Riolo, and B. Worzel, (Eds.), "Genetic Programming Theory and Practice III," Springer US, pp. 223-240, doi: 10.1007/0-387-28111-8_15, 2006.

[21] E. Alfaro-Cid, J. J. Merelo, F. F. de Vega, A. I. Esparcia-Alcár, and K. Sharman, "Bloat Control Operators and Diversity in Genetic Programming: A Comparative Study," Evolutionary Computation, Vol.18, No.2, pp. 305-332, doi: 10.1162/evco.2010.18.2.18206, 2010.

[22] A. R. Burks and W. F. Punch, "An Efficient Structural Diversity Technique for Genetic Programming," Proc. of the 2015 Annual Conf. on Genetic and Evolutionary Computation, pp. 991-998, doi: 10.1145/2739480.2754649, 2015.

[23] E. D. de Jong, R. A. Watson, and J. B. Pollack, "Reducing bloat and promoting diversity using multi-objective methods," Proc. of the 3rd Annual Conf. on Genetic and Evolutionary Computation, pp. 11-18, 2001.

[24] K. Miyashita, "Job-shop scheduling with genetic programming," Proc. of the 2nd Annual Conf. on Genetic and Evolutionary Computation, pp. 505-512, 2000.

[25] C. D. Geiger, R. Uzsoy, and H. Aytuğ, "Rapid modeling and discovery of priority dispatching rules: An autonomous learning approach," J. of Scheduling, Vol.9, No.1, pp. 7-34, doi: 10.1007/s10951-006-5591-8, 2006.

[26] M. Durasević, D. Jakobović, and K. Knežević, "Adaptive scheduling on unrelated machines with genetic programming," Applied Soft Computing, Vol.48, pp. 419-430, doi: 10.1016/j.asoc.2016.07.025, 2016.

[27] S. Salama, T. Kaihara, N. Fujii, and D. Kokuryo, "A Proposal on Dispatching Rule Generation Mechanism Using GP for Dynamic Job Shop Scheduling with Machine Breakdowns," Proc. of the Scheduling Symp. 2020, pp. 155-160, 2020.

[28] J. Branke, T. Hildebrandt, and B. Scholz-Reiter, "Hyper-heuristic evolution of dispatching rules: A comparison of rule representations," Evolutionary Computation, Vol.23, No.2, pp. 249-277, doi: 10.1162/EVCO_a_00131, 2015.

[29] A. Ekárt and S. Z. Németh, "A Metric for Genetic Programs and Fitness Sharing," Genetic Programming: European Conf. on Genetic Programming Proc., pp. 259-270, doi: 10.1007/978-3-540-46239-2_19, 2000.

[30] D. Jackson, "Phenotypic Diversity in Initial Genetic Programming Populations," Genetic Programming: European Conf. on Genetic Programming Proc., pp. 98-109, doi: 10.1007/978-3-642-12148-7_9, 2010.

[31] S. Gustafson and L. Vanneschi, "Crossover-Based Tree Distance in Genetic Programming," IEEE Trans. on Evolutionary Computation, Vol.12, No.4, pp. 506-524, doi: 10.1109/TEVC.2008.915993, 2008.

[32] J. Kelly, E. Hemberg, and U.-M. O'Reilly, "Improving Genetic Programming with Novel Exploration – Exploitation Control," Genetic Programming: European Conf. on Genetic Programming Proc., pp. 64-80, doi: 10.1007/978-3-030-16670-0_5, 2019.

[33] L. Vanneschi, M. Castelli, and S. Silva, "A survey of semantic methods in genetic programming," Genet. Program. Evolvable Mach., Vol.15, No.2, pp. 195-214, doi: 10.1007/s10710-013-9210-0, 2014.

[34] B. Sareni and L. Krahenbuhl, "Fitness sharing and niching methods revisited," IEEE Trans. on Evolutionary Computation, Vol.2, No.3, pp. 97-106, doi: 10.1109/4235.735432, 1998.

[35] M. Hughes, "Investigating the Effects Diversity Mechanisms Have on Evolutionary Algorithms in Dynamic Environments," arXiv: 1610.02732, 2021.

[36] P. Wong and M. Zhang, "Algebraic simplification of GP programs during evolution," Proc. of the 8th Annual Conf. on Genetic and Evolutionary Computation, pp. 927-934, doi: 10.1145/1143997.1144156, 2006.

[37] Y. Mei, S. Nguyen, B. Xue, and M. Zhang, "An Efficient Feature Selection Algorithm for Evolving Job Shop Scheduling Rules With Genetic Programming," IEEE Trans. on Emerging Topics in Computational Intelligence, Vol.1, No.5, pp. 339-353, doi: 10.1109/tetci.2017.2743758, 2017.

[38] S. Salama, T. Kaihara, N. Fujii, and D. Kokuryo, "A New Representation and Adaptive Feature Selection for Evolving Compact Dispatching Rules for Dynamic Job Shop Scheduling with Genetic Programming," Advances in Production Management Systems. Artificial Intelligence for Sustainable and Resilient Production Systems, Cham, pp. 646-654, doi: 10.1007/978-3-030-85906-0_70, 2021.

[39] J. R. Koza, "Genetic programming as a means for programming computers by natural selection," Stat. Comput., Vol.4, No.2, pp. 87-112, doi: 10.1007/BF00175355, 1994.

[40] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," IEEE Trans. on Evolutionary Computation, Vol.6, No.2, pp. 182-197, doi: 10.1109/4235.996017, 2002.

[41] F. Zhang, Y. Mei, and M. Zhang, "Evolving Dispatching Rules for Multi-Objective Dynamic Flexible Job Shop Scheduling via Genetic Programming Hyper-Heuristics," Proc. of the 2019 IEEE Congress on Evolutionary Computation (CEC), pp. 1366-1373, doi: 10.1109/CEC.2019.8790112, 2019.

[42] E. Taillard, "Benchmarks for basic scheduling problems," European J. of Operational Research, Vol.64, No.2, pp. 278-285, doi: 10.1016/0377-2217(93)90182-M, 1993.

Name:
Shady Salama

Affiliation:
Graduate School of System Informatics, Kobe University

Address:
1-1 Rokkodai-cho, Nada-ku, Kobe, Hyogo 657-8501, Japan

Brief Biographical History:
2018 Received Master's degree from Egypt-Japan University of Science and Technology
2018-2019 Research Student, Graduate School of System Informatics, Kobe University
2019- Ph.D. Student, Graduate School of System Informatics, Kobe University

Main Works:
• "Evolving Dispatching Rules Using Genetic Programming for Multi-Objective Dynamic Job Shop Scheduling with Machine Breakdowns," Procedia CIRP, Vol.104, pp. 411-416, 2021.
• "A decision support system architecture based on simulation optimization for cyber-physical systems," Procedia Manufacturing, Vol.26, pp. 1147-1158, 2018.

Name:
Toshiya Kaihara

Affiliation:
Professor, Graduate School of System Informatics, Kobe University

Address:
1-1 Rokkodai-cho, Nada-ku, Kobe, Hyogo 657-8501, Japan

Brief Biographical History:
1985- Mitsubishi Electric Corporation
2001-2004 Associate Professor, Kobe University
2004- Professor, Kobe University

Main Works:
• "Value creation in production: Reconsideration from interdisciplinary approaches," CIRP Annals – Manufacturing Technology, Vol.67, No.2, pp. 791-813, 2018.
• "Optimization and Simulation of Collaborative Networks for Sustainable Production and Transportation," IEEE Trans. on Industrial Informatics, Vol.12, No.1, pp. 417-424, 2016.
• "Cloud-Based Additive Manufacturing and Automated Design: A PLM-Enabled Paradigm Shift," Sensors, Vol.15, No.12, pp. 32079-32122, 2015.

Membership in Academic Societies:
• International Academy for Production Engineering (CIRP)
• International Federation for Information Processing (IFIP)
• International Federation of Automatic Control (IFAC)
• Engineering Academy of Japan (EAJ)

Name:
Nobutada Fujii

Affiliation:
Associate Professor, Department of Systems Science, Graduate School of System Informatics, Kobe University

Address:
1-1 Rokkodai-cho, Nada-ku, Kobe, Hyogo 657-8501, Japan

Brief Biographical History:
1998- JSPS Research Fellow, Kobe University
2000- Research Associate, Department of Mechanical Engineering, Kobe University
2002- Research Associate, Research into Artifacts, Center for Engineering (RACE), The University of Tokyo
2005- Invited Associate Professor, Research into Artifacts, Center for Engineering (RACE), The University of Tokyo
2007- Associate Professor, Graduate School of Engineering, Kobe University
2010- Associate Professor, Graduate School of System Informatics, Kobe University

Main Works:
• "An EOQ Model for Reuse and Recycling Considering the Balance of Supply and Demand," Int. J. Automation Technol., Vol.9, No.3, pp. 303-311, 2015.

Membership in Academic Societies:
• Japan Society of Mechanical Engineers (JSME)
• Japan Society for Precision Engineering (JSPE)
• Society for Serviceology (SfS)
• Institute of Systems, Control and Information Engineers (ISCIE)

Name:
Daisuke Kokuryo

Affiliation:
Assistant Professor, Graduate School of System Informatics, Kobe University

Address:
1-1 Rokkodai-cho, Nada-ku, Kobe, Hyogo 657-8501, Japan

Brief Biographical History:
2007 Received Ph.D. from Graduate School of Science and Technology, Kobe University
2008-2014 Postdoctoral Fellow / Researcher, National Institute of Radiological Sciences
2015- Project Assistant Professor / Assistant Professor, Graduate School of System Informatics, Kobe University

Main Works:
• "Value Co-Creative Manufacturing with IoT Based Smart Factory for Mass Customization," Int. J. Automation Technol., Vol.11, No.3, pp. 509-518, 2017.
• "A Proposal of Max-Min Ant System for Keeping Diversity of Search Performance Using Search Histories," IEEJ Trans. on Electronics, Information and Systems, Vol.139, No.12, pp. 1488-1493, 2019.

Membership in Academic Societies:
• Institute of Systems, Control and Information Engineers (ISCIE)
• Institute of Electrical and Electronics Engineers (IEEE)
• International Society for Magnetic Resonance in Medicine (ISMRM)
• Scheduling Society of Japan (SSJ)