Contemporary Mathematics
http://ojs.wiserpub.com/index.php/CM/

Research Article

Generating Pareto Optimal Solutions for Multi-Objective Optimization Problems Using Goal Programming

Alyaa Hegazy Abdelhamid[1,2*], Ramadan Hamed Mohamed[1], Mahmoud Mostafa Rashwan[1], Aya Rezk Allah Farag[1]

[1]Department of Statistics, Faculty of Economics and Political Science, Cairo University, Cairo, Egypt
[2]Department of Mathematics, Faculty of Basic Sciences, The German University in Cairo (GUC), Cairo, Egypt
Email: alyaahegazy90@feps.edu.eg

Received: 13 March 2023; **Revised:** 18 March 2023; **Accepted:** 5 July 2023

Abstract: Goal programming (GP) is a well-known multi-criteria decision-making tool that is supported by a network of practitioners and researchers who aim to develop its mathematical foundation to cover a wide range of applications. The popularity of GP models stems from their structure, which is based on a satisfying philosophy. This philosophy takes into consideration the preferences of the decision-maker concerning the model parameters. Therefore, the GP model provides the decision-maker with one satisfactory solution that reflects the trade-off between competing objectives. Nevertheless, there is no guarantee regarding the efficiency of this solution. Consequently, this study is designed to improve the quality of decision-making processes by addressing the efficiency issue with the solutions of GP models. The main contribution of this paper is to improve the mathematical framework of the GP model so that it can generate a set of Pareto optimal solutions rather than just one solution. This allows stakeholders to have a complete picture of the feasible space of solutions and select the solution that represents the best compromise according to their preferences. As a result, the proposed methodology is called generational GP. In addition, the study enhances the quality of GP solutions by integrating the notion of the hypervolume subset selection problem with the suggested technique. This, in turn, overcomes the efficiency problem of GP solutions. The performance of the proposed method has been validated through an application to the flow shop scheduling problem. However, our modeling approach is useful for decision-makers in different fields of study. Finally, the merits of the generational GP method are highlighted, with a strong emphasis on potential areas for future research.

***Keywords*:** multi-objective optimization problems, Pareto optimal solutions, generational GP, hypervolume subset selection problem, green permutation flowshop scheduling problem

MSC: 26A33, 65D30, 92D25, 93A30

1. Introduction

Most real-world applications take into account multiple competing objectives. There are two primary categories of methods for solving multi-objective optimization problems (MOPs) [1]. The first category includes evolutionary algorithms. The second category involves a range of classical techniques. These methods are sorted into three different

DOI: https://doi.org/10.37256/cm.5320242674

groups depending on the phase in which decision-makers participate in the decision-making process [2]. These groups are known in the literature as *a posteriori (generation), a priori, and interactive* methods.

The first group involves techniques that generate a set of efficient solutions for the decision-maker. Therefore, the intervention of the stakeholder is limited to the selection process for solutions. The epsilon constraint method is one of the most common techniques in this group. On the contrary, the methods in the second group, i.e., a priori methods, allow decision-makers to express their preferences before starting the solution process. Goal programming (GP) is a popular technique in this group. In GP, model parameters, such as the aspiration levels of goals, are determined based on the preferences of the decision-maker before solving the optimization problem. In interactive methods, stakeholders interact after each step of the solution process to show their preferences [2]. Despite the variety of multi-criteria decision-making (MCDM) tools to solve MOPs, this study focuses on developing the mathematical framework of GP to provide the decision-maker with the solution of the best compromise.

The GP technique is one of the most widely used MCDM approaches because of the simplicity of its mathematical framework [3]. This framework takes into account the simultaneous optimization of competing objectives. Therefore, the GP model provides the decision-maker with a solution that reflects the trade-off between several conflicting objectives. Moreover, the GP approach is based on the satisficing philosophy, which allows the decision-maker to represent his or her preferences concerning the model parameters. Consequently, the quality of the decision-making processes is enhanced through interacting with the stakeholder before solving the GP model.

The mathematical structure of GP models is based on minimizing a distance function that considers the deviations between the objectives and their target values, which are determined by the decision-maker. The first GP model was initially presented by Charnes et al. [4], and it was considered an extension to linear programming models. Furthermore, several variants of GP models were suggested in the literature to represent the decision-maker's preferences. The most common variants are weighted GP (WGP) and lexicographic GP (LGP) [5]. On the one hand, WGP focuses on minimizing the weighted sum of the deviations from all objectives. These weights reflect the relative importance of each objective. On the other hand, the LGP models allow the decision-maker to rank objectives according to their importance. The deviations from the goals at the highest priority level are minimized first. Goals at the next level of importance are optimized while maintaining the minimal values of goals at higher priority levels. Consequently, this variant is based on solving a series of sequential optimization models [6].

Despite the popularity of GP and the variety of its variants to solve different MOPs, there is no guarantee that it provides the decision-maker with an efficient solution. The efficiency issue happens due to the mathematical structure of GP, which is based on the satisfying philosophy [7]. The essence behind this philosophy is to provide the decision-maker with a solution that is as close as possible to the desired target value. This is opposed to the optimization philosophy, which seeks to find an optimal solution. Consequently, several studies were introduced in the literature to handle this problem. The contributions of these studies are classified into two main directions [8]. The first one focused on testing only the efficiency of a GP solution, such as the technique introduced by Cabllero et al. [9]. The second type of study was designed to provide the decision-maker with an efficient solution in the case of the failure of an efficiency test [8, 10, 11].

This study aims at developing the mathematical framework of GP models to avoid introducing a dominant solution to the decision-maker. In addition, the study suggests a GP model that generates a set of efficient solutions rather than just one solution. This set covers the feasible space of potential solutions. This, in turn, allows the stakeholder to have a variety of efficient candidate solutions and select the one that ideally reflects his or her preferences. Consequently, the contribution of this paper is twofold. Firstly, the suggested technique is able to produce a set of GP solutions rather than just one solution. Therefore, the framework of GP is adjusted to be similar to the a posteriori approach. This implies that the proposed generational goal programming (GGP) methodology converts the GP technique to a generation method while maintaining the preferences of the decision-maker. The second contribution of this paper is to improve the quality of GP solutions by overcoming the efficiency issues related to them. This is accomplished by combining the proposed technique with the concept of the hypervolume subset selection problem (HYPssp). This allows stakeholders to have a subset of non-dominated solutions that capture the most diverse and well-distributed solution points in the objective space. To assess the reliability of the proposed method, it has been applied to the green permutation flowshop scheduling problem (GPFSP). This application has practical benefits in the fields of manufacturing and industrial engineering, and it is used as an example to show the usefulness of the suggested technique. Nevertheless, the proposed GGP methodology can be compared to the current optimization methods in the literature in addition to applying it to other applications, such as resource allocations among level crossings [12], the vehicle routing problem [13], the ship scheduling problem [14], the complex building design problem [15], and the vessel scheduling problem [16].

The rest of this paper is arranged as follows: Section 2 introduces the proposed GGP method. Section 3 gives an overview of the GPFSP. Moreover, it presents the mathematical programming models used in this study. In Section 4, data settings and computational results are illustrated to test the validity of the proposed technique. The last section concludes the paper.

2. The proposed GGP methodology

In this study, the suggested GGP technique is introduced in the context of bi-objective minimization problems. However, it can be generalized to any MOP. The purpose of a two-dimensional minimization problem is to find a vector of decision variables in the decision space X that minimizes a vector of conflicting objectives in the objective space, i.e., $f : \mathrm{X} \to \mathbb{R}^2$. Furthermore, this study is interested in points in the objective space that form a subset of $\mathbb{R}^2$. A point $p = (p_1, p_2) \in \mathbb{R}^2$ weakly dominates another point $q = (q_1, q_2) \in \mathbb{R}^2$ (also known as $p \succeq q$) if and only if $p_t \leq q_t$ for $t = 1, 2$ [17]. Therefore, the desired output from any technique used to solve MOP is to obtain non-dominated solutions. These solutions are incomparable and are used to determine a Pareto frontier [18].

The proposed GGP methodology relies on several steps. The first step is based on calculating the minimum and maximum values for each objective. On the one hand, the minimum value of each objective is considered an ideal value since the suggested technique is illustrated in the context of bi-objective minimization problems. Ideal values are obtained by running two single-objective optimization models. On the other hand, a maximum value is regarded as the worst value that an objective function reaches. Maximum values are estimated in this study by adopting the idea of redundant goals in LGP [19, 20].

The issue of goal redundancy occurs in the lexicographic variant of GP when the target values of goals at the highest priority level are equal to their ideal values. This results in redundancy in goals placed at the lowest priority level, which implies that these goals are not achieved [20]. This also implies that the goals at the lowest priority level obtain their worst possible values. Consequently, this concept is adopted in this study to estimate the worst value for each objective. On the one hand, the maximum value of the first objective is obtained by placing it at the second priority level in the LGP model. Due to choosing the ideal values of both goals as their target levels, the model achieves the goal at the first priority level, i.e., the second objective, and produces the worst value of the goal at the second rank, i.e., the first objective. On the other hand, the worst value of the second objective is achieved by running the same LGP model after reversing the previous priority order.

After computing the maximum and minimum values for each objective, the next step aims at constructing a group of intervals for each objective to generate a set of aspiration levels for the LGP model. To create these intervals, the range of each objective is determined as the difference between the maximum and minimum values. The length of each interval is the range divided by the number of intervals (n), which is determined by the decision-maker. Furthermore, this number is equivalent to the number of aspiration levels since one aspiration level is randomly generated from each interval. Therefore, the set w of n points of aspiration levels is obtained from these intervals. The x-coordinate of each point represents an aspiration level for the first objective, while the y-coordinate is an aspiration level for the second objective. Moreover, the intervals are constructed such that these points have an increasing order of the first coordinates and a decreasing order of the second coordinates.

After running the LGP model n times to obtain a set of GP solutions, the last step in the proposed methodology is to select a subset of non-dominated solutions. There are several criteria for choosing this subset. This study follows the approach presented by Bringmann et al. [17] to obtain a subset of incomparable solutions. The authors suggested the HYPssp algorithm, which produces a subset of efficient solutions based on the maximum value of the hypervolume performance metric. This indicator, which was introduced by Zitzler and Thiele [21], is designed to measure the volume of the objective space covered by a set of solutions according to a reference value. This paper follows the study of Hughes [22], which estimated the coordinates of the reference point as the maximum (worst) value for each objective.

There are several reasons behind combining the HYPssp algorithm with the proposed methodology. Firstly, GP does not necessarily produce a Pareto-optimal solution [8]. As mentioned in the previous section, GP may provide the decision-maker with an inefficient solution since its mathematical structure follows the satisfying philosophy rather than the optimizing one. Secondly, and most importantly, rational decision-makers do not choose a dominant solution [8]. Accepting a dominant solution occurs in the traditional variants of GP because other efficient solutions are unavailable to stakeholders due to a lack of knowledge about the feasible region. Consequently, adjusting the structure of GP to generate a set of solutions is considered fundamental for stakeholders who are interested in exploring the feasibility of GP solutions. The HYPssp algorithm ensures that inefficient GP solutions are avoided. Therefore, the decision-maker selects from this subset the most suitable solution for the MOP under consideration.

The proposed GGP methodology can be summarized in the framework of bi-objective minimization problems as follows:

1. Calculate the minimum (ideal) value of each objective using a single-objective optimization model.
2. Compute the maximum (worst) value of each objective by considering the redundancy issue of LGP.
3. Set intervals for each objective, and choose an aspiration level at random from each one.
4. Construct the set w of points for aspiration levels.
5. Use these target values for the goals in the LGP model.
6. Run the HYPssp algorithm to obtain a subset of efficient GP solutions.

The advantages of the suggested technique are highlighted by applying it to the GPFSP. This application has practical benefits in the field of industrial engineering. The next section introduces this application and explains in detail the steps of the GGP methodology.

3. Overview of the GPFSP

This section introduces the GPFSP. In addition, it presents the mathematical programming models used to solve it in the context of the proposed GGP methodology.

3.1 *Introduction to the GPFSP*

The traditional permutation flowshop scheduling problem (PFSP) is a combinatorial optimization problem that takes objectives related to production time into account. The maximum completion time, i.e., makespan, flow time, and tardiness, are the most common production efficiency-related objectives in the literature. The purpose of the PFSP is to identify the optimal order of processing jobs on machines. Additionally, it makes the assumption that jobs will be processed on machines in the same order [23]. Other assumptions include the independence of jobs and the availability of jobs at the start of processing time. Moreover, machines are independent, and their interruption is not allowed during the processing stage. These assumptions are considered in this study. More details about the PFSP and its requirements are available at [24].

The GPFSP is considered an extension of the traditional PFSP. This recent variant considers energy efficiency-related objectives, such as energy consumption and carbon emissions. In this study, makespan and total energy consumption (TEC) are considered the competing objectives, and they are assessed using the proposed methodology. Moreover, the speed of machines is the factor that creates the conflict between these objectives. Processing jobs at a higher speed decreases makespan. However, this increases the energy consumed by machines.

Several optimization techniques were introduced to evaluate the trade-off between production and energy efficiency-related objectives in the context of the speed-scaling strategy. Mansouri et al. [25] used the epsilon constraint method and developed constructive heuristics to study the compromise between makespan and TEC for a two-machine sequence-dependent permutation flowshop scheduling problem. Moreover, the compromise between the total flow time and TEC was assessed in the context of the GPFSP [26]. The authors employed the augmented epsilon constraint technique to generate Pareto-optimal solutions for small-scale problems. To cope with the complexity of this problem, their study proposed multi-objective iterated greedy algorithms and variable block insertion heuristics for large-scale problems. Furthermore, Saber and Ranjbar [27] studied the conflict between the total tardiness and the total carbon emissions for the GPFSP. The authors suggested a mixed integer mathematical programming model and a multi-objective decomposition-based heuristic algorithm to solve this problem. Recently, in the context of task scheduling for parallel systems, Stewart et al. [28] developed a mixed-integer mathematical programming model to minimize makespan and energy consumption by varying the speed of processors. The authors used the epsilon constraint and the weighted sum scalarization methods to solve this problem.

The next subsection introduces the mathematical programming models used to study the GPFSP.

3.2 *Mathematical programming models for the GPFSP*

As mentioned in Section 2, the proposed GGP methodology relies on calculating the minimum and maximum values for each objective. Model 1 below is a single-objective mixed-integer mathematical programming model that is used twice to compute the minimum (ideal) values for makespan and TEC. These ideal values are used as target levels in the LGP model, i.e., Model 2 below. Before introducing these models, Table 1 below presents the notations, parameters, and decision variables used.

Table 1. Indexes, parameters, and variables of the mathematical programming models

Indexes		Parameters		Positive decision variables		Binary decision variables	
i	Index for jobs	N	Number of jobs: $i, j = 1, 2, \ldots, N$	C_{jm}	Completion time of the job in the jth position on m machine	x_{ijms}	1 if job i is in the jth position on machine m at speed s, 0 otherwise
j	Index for positions of jobs	M	Number of machines: $m, k = 1, 2, \ldots, M$	θ_m	Idle time on machine m		
m, k	Indexes for machines	S	Number of speed levels; $s = 1, 2, 3$ for fast, normal and slow speeds, respectively.	$C_{\max}$	Maximum completion time (makespan)		
s	Index for processing speeds	p_{im}	The standard processing time of job i on machine m	TEC	Total energy consumption (KWh)		
		v_s	Processing speed factor				
		γ_s	Conversion factor for processing speed s				
		φ_m	Conversion factor for idle time on machine m				
		π_m	Power of machine m (KWh)				

It is worth mentioning that Model 1 below is similar to the model introduced by Amiri and Behnamian [29]. However, their model is designed to assess the stochastic variant of the GPFSP under scenario analysis. The deterministic version of their model is used in this study.

Model 1. Deterministic mixed-integer mathematical programming model for GPFSP

$$\min z_1 = C_{\max}{}^{*} \tag{1}$$

$$\min z_2 = \text{TEC}^{*} \tag{2}$$

subject to

$$C_{11} \geq \sum_{i=1}^{N} \sum_{s=1}^{S} x_{i11s} * \frac{p_{i1}}{v_s} \tag{3}$$

$$C_{j,m+1} \geq C_{jm} + \sum_{i=1}^{N} \sum_{s=1}^{S} x_{ijms} * \frac{p_{i,m+1}}{v_s}, \forall j \epsilon N, \forall m \in M \tag{4}$$

$$C_{j+1,m} \geq C_{jm} + \sum_{i=1}^{N} \sum_{s=1}^{S} x_{i,j+1,m,s} * \frac{p_{im}}{v_s}, \forall j \epsilon N, \forall m \in M \tag{5}$$

$$\sum_{s=1}^{S} x_{ijms} = \sum_{s=1}^{S} x_{ijks}, \forall i, j \epsilon N, \forall m, k \in M \tag{6}$$

$$\sum_{i=1}^{N} \sum_{s=1}^{S} x_{ijms} = 1, \forall j \epsilon N, \forall m, \in M \tag{7}$$

$$\sum_{j=1}^{N} \sum_{s=1}^{S} x_{ijms} = 1, \forall i \epsilon N, \forall m, \in M \tag{8}$$

$$C_{\max}{}^{*} \geq C_{NM} \tag{9}$$

$$\theta_m = C_{\max}{}^{*} - \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{s=1}^{S} x_{ijms} * \frac{p_{im}}{v_s}, \forall m \in M \tag{10}$$

$$\text{TEC}^{*} = \sum_{m=1}^{M} \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{s=1}^{S} x_{ijms} * \frac{\pi_m * \gamma_s * p_{im}}{60 v_s} + \sum_{m=1}^{M} \frac{\pi_m \varphi_M \theta_m}{60} \tag{11}$$

$$C_{jm} \geq 0, \forall j \epsilon N, \forall m \in M \tag{12}$$

$$x_{ijms} \in \{0,1\}, \forall i, j \epsilon N, \forall m \in M, \forall s \epsilon S \tag{13}$$

The objective functions (1) and (2) aim at minimizing makespan and TEC, respectively. Each objective function is minimized individually to get its ideal value. This implies that Model 1 is used twice. Constraint (3) states that the completion time of the job in the first position on the first machine should equal the processing time on that machine. The inequality is used in the case of machine idle time. Constraint (4) guarantees that a job in sequence position j cannot end its processing on the current machine unless it has already finished its processing on the previous machine. Constraint (5) states that a job in processing on machine m can move to the next position in the sequence after ensuring that the job in the previous position has finished processing on the same machine. In other words, the completion time of any job on any machine is determined by the processing time on that machine in addition to the completion time of its predecessor on the same machine. Constraint (6) states that jobs are processed in the same order, with one speed on each machine. Constraint (7) ensures that each position on each machine has one job with one speed. Furthermore, constraint (8) guarantees that each job is processed at exactly one speed and has one position on each machine. The makespan of the schedule is defined in constraint (9). Idle times on machines are computed using constraint (10). Constraint (11) calculates TEC in kilowatt hours. The non-negativity and binary constraints of the decision variables are defined in constraints (12) and (13), respectively.

It is worth noting that the output of Model 1 is twofold. Firstly, it gives the optimal sequence of processing jobs on machines. Secondly, it assigns the optimal processing speed of each job to each machine. Moreover, the LGP version of Model 1 is introduced in Model 2 below. Model 2 is used to estimate the worst value of each objective.

Model 2. Deterministic LGP model for the GPFSP

Achievement function

$$\min Z = [p_1, p_2] \tag{14}$$

subject to:

Goal constraints

$$C_{\max} + n_1 - p_1 = C_{\max}{}^* \tag{15}$$

$$\text{TEC} + n_2 - p_2 = \text{TEC}^* \tag{16}$$

System constraints (3)-(13)

$$n_1, p_1, n_2, p_2 \geq 0 \tag{17}$$

where n_1 and n_2 (p_1 and p_2) are the negative (positive) deviational variables of the two goals. These deviational variables are defined to be non-negative, as stated in constraint (17). Constraints (15) and (16) specify the two goals and their corresponding target values, i.e., $C_{\max}{}^*$ and TEC*, respectively. These target values are calculated from Model 1 above, and they represent the ideal values for both goals. The achievement function in equation (14) is a vector of positive deviational variables since both objectives have to be minimized. Furthermore, the achievement function places makespan in the first priority order, followed by TEC in the second place. By considering this order, the maximum (worst) value of TEC is computed due to redundancy in the LGP model. The worst value of makespan is estimated by reversing the previous priority order of the two goals.

The proposed GGP methodology is based on generating a set of GP solutions rather than just one solution. These solutions are obtained by varying the aspiration levels for both goals. This is achieved by a random generation from the sets of intervals in constraints (18) and (19) below. On the one hand, the first set of intervals guarantees that the target levels for makespan (g_t^1) are increasing in order. On the other hand, the construction of the second set of intervals ensures obtaining values in a decreasing order for the aspiration levels of TEC (g_t^2).

$$g_t^1 \in \left[\min C_{\max} + (t-1) * \text{length}_{C_{\max}}, \min C_{\max} + t * \text{length}_{C_{\max}}\right), \forall t = 1, 2 \ldots, n \tag{18}$$

$$g_t^2 \in \left[\max \text{TEC} - t * \text{length}_{\text{TEC}}, \max \text{TEC} - (t-1) * \text{length}_{\text{TEC}}\right), \forall t = 1, 2 \ldots, n \tag{19}$$

where min $C_{\max}$ is the minimum (ideal) value of the makespan, which is calculated from Model 1 above. Max TEC is the maximum (worst) value of the second objective, and it is computed from Model 2. Moreover, the length$_{C_{\max}}$ (length$_{\text{TEC}}$) is the difference between the maximum and minimum values of makespan (TEC) divided by the number of intervals (n). This number is determined by the decision-maker, and it is equal to the number of aspiration levels used for Model 3 below.

Model 3. The GGP model for the GPFSP

Achievement function (14)

Subject to:

Goal constraints

$$C_{\max} + n_1 - p_1 = g_t^1, \quad \forall t = 1, 2, \ldots, n \tag{20}$$

$$\text{TEC} + n_2 - p_2 = g_t^2, \quad \forall t = 1, 2, \ldots, n \tag{21}$$

System constraints (3)-(13) and (17).

The aspiration levels for the first (second) goal are $g_t^1 (g_t^2)$. In addition, they are computed from the set of intervals in constraints (18) and (19) above. Consequently, constraints (20) and (21) imply that Model 3 is run n times to obtain a set of n GP solutions. Since there is no guarantee about the efficiency of these solutions, the next step involves applying the HYPssp algorithm [17] to select a subset of size l of these solutions. This subset involves efficient solutions that maximize the hypervolume performance indicator.

The next section illustrates the data settings of the GPFSP and the computational results of the proposed GGP technique.

4. Data settings and computational results

The literature on the flowshop scheduling problem includes several benchmarks to validate the exact and meta-heuristic algorithms. The benchmark of Tailard [30] is the most commonly used. Nevertheless, this study adopts the PFSP benchmark instances of Vallada et al. [31] to assess the performance of the proposed GGP methodology. The reason for this choice is based on the variety of small datasets available in this benchmark, which is consistent with the exactness of the suggested technique. The authors generated 240 large instances and the same number for small- to medium-sized instances. A combination of N jobs and M machines is used to construct their proposed set of benchmark instances. In addition, the number of jobs is between 10 and 60, while the number of machines ranges from five to 20, i.e., $N = \{10, 20, 30, 40, 50, 60\}$, and $M = \{5, 10, 15, 20\}$. 10 instances are created for each combination of the elements of the two sets. The processing times of jobs are calculated using a uniform distribution. This study considers applying the suggested GGP method to the first five instances of the first combination, i.e., the combination of 10 jobs and five machines.

Table 2 below shows the minimum and maximum values calculated for each objective in each instance. On the one hand, the minimum (ideal) value for each objective is computed from Model 1 above. On the other hand, maximum values are obtained by adopting the idea of goal redundancy in LGP, as illustrated in Model 2.

Table 2. Minimum and maximum values of the makespan and total energy consumption

Instance	$C_{\max}$		TEC	
	min	max	min	max
10 × 5-01	579.1667	868.75	1073.8975	2365.4875
10 × 5-02	581.6667	872.5	1119.09	2933.9333
10 × 5-03	606.6667	910	1186.08	2862.3858
10 × 5-04	580.8333	871.25	1154.265	3185.9826
10 × 5-05	594.1667	891.25	1212.705	3428.4615

Table 3 below illustrates the parameters used for the GPFSP. The regular parameters of the PFSP are adopted from the study of Vallada et al. [31]. They include the number of jobs (N), the number of machines (M), and the data for processing time. The study of Mansouri et al. [25] is used as a reference for setting energy parameters.

The uniform and normal distributions are used to generate the aspiration levels for each objective, i.e., g_t^1 and g_t^2 ($\forall t = 1, 2, \ldots, n$) for makespan and TEC, respectively. These distributions are used in this study for the purpose of comparison. On the one hand, the parameters of the uniform distribution are the lower and upper bounds of each interval computed from the set of intervals in constraints (18) and (19). On the other hand, the location parameter of the normal distribution is the average of the lower and upper bounds. The standard deviation is computed as the length of the interval divided by six. The computations of these parameters are based on the study of Moghaddam [32].

Table 3. Summary of the parameters of the GPFSP

Parameter	Level
Number of jobs (N)	10
Number of machines (M)	5
Processing time (p_{im})	Uniform (1, 99)
Machines power (π_m)	60 kilowatts
Processing speed (v_s)	{1.2, 1, 0.8}
Processing conversion factor (γ_s)	{1.5, 1, 0.6}
Idle time energy consumption (φ_m)	0.05

For each of the five instances, Model 3 is used to obtain two sets of GP solutions. This is achieved using normally and uniformly distributed aspiration levels. Due to obtaining inefficient GP solutions under each instance, the HYPssp algorithm [17] is utilized to obtain subsets of non-dominated solutions that give the maximum value of the hypervolume indicator. Columns 3 and 4 of Table 4 below illustrate the normalized hypervolume values calculated from each subset of efficient solutions using both distributions. Observing these values across the five instances concludes that applying the GGP methodology using normally distributed aspiration levels generates subsets of non-dominated solutions that have a relatively larger volume in the objective space. This means that these solutions are more diverse and well-distributed than those generated by a uniform distribution. In addition, the initial size (n) of the set of GP solutions and the size (l) of the subsets of Pareto optimal solutions are provided in columns 1 and 2, respectively. The sizes of the sets of GP solutions are arbitrarily chosen, and the sizes of the corresponding subsets are selected as n / 2.

Tables A1 and A2 in the appendix show detailed computations for the first instance, i.e., the instance 10 × 5-01. Columns 1 and 2 of Table A1 contain the aspiration levels obtained from the uniform distribution and the corresponding GP solutions, respectively. In addition, column 3 involves the aspiration levels generated from the normal distribution, and column 4 presents the corresponding GP solutions. The subsets of efficient solutions are shown in columns 1 and 2 of Table A2. These efficient solutions are produced using the HYPssp algorithm [17]. Due to space limitations, the tables for the rest of the instances are omitted. However, they are available upon request.

Table 4. Results of the hypervolume performance metric

Instance	Hypervolume values			
	Initial size (n) (1)	Subset size (l) (2)	Uniform random goals (3)	Normal random goals (4)
10 × 5-01	80	40	0.5187	0.5226
10 × 5-02	60	30	0.5015	0.5087
10 × 5-03	50	25	0.5132	0.5111
10 × 5-04	56	28	0.5139	0.5148
10 × 5-05	48	24	0.5040	0.5576

It is worth noting that the mathematical programming models introduced in this study are implemented using the GAMS software version 24.1.1, and CPLEX is used as the solver. The Java code, written by Bringmann et al. [17], is used to run the HYPssp algorithm. The test instances were run on a laptop with a 1.8 GHz Intel Core i5 8th-generation processor and 8 GB of RAM.

5. Conclusion

This paper presents a new GP approach to improving the quality of decision-making processes. The proposed GGP method aims to overcome the efficiency issue of GP solutions while preserving the preferences of the decision-maker. Furthermore, this study introduces an enhancement to the mathematical framework of GP by generating a set of solutions rather than just one solution. This set reflects trade-offs between competing objectives and provides the decision-maker with a complete picture of the feasible space of solutions. In addition, the study highlights the importance of dealing with the efficiency problem of GP solutions. This is achieved by integrating the concept of the HYPssp with the proposed technique. This results in obtaining a set of Pareto optimal solutions that are more diverse and representative of the objective space. To validate the performance of the suggested technique, it has been applied to the GPFSP. This application has practical importance in the field of industrial engineering. Concerning future work, the paper recommends the following points for future research: Firstly, apply the GGP technique to different applications in other fields of study to benefit from its advantages. Secondly, conduct the proposed method using other versions of GP, such as the weighted variant. Finally, study the effect of including uncertain parameters on the mathematical structure of the GGP methodology.

Acknowledgments

The authors would like to express their deep appreciation to the anonymous reviewers for enhancing the quality of the manuscript. The technical notes from Dr. Mohamed Salman are sincerely acknowledged.

Data availability

The data that support the findings of this study are included in this published article https://doi.org/10.1016/j.ejor.2014.07.033.

Conflict of interest

The authors declare that they have no conflict of interest.

References

[1] Deb K. Multi-objective optimization using evolutionary algorithms: An introduction. In: Wang L, Ng A, Deb K. (eds.) *Multi-objective evolutionary optimisation for product design and manufacturing*. London: Wiley; 2011. Available from: https://doi.org/10.1007/978-0-85729-652-8_1.

[2] Hwang C-L, Md Masud AS. *Multiple objective decision making-methods and applications: A state-of-the-art survey*. Berlin, Heidelberg: Springer; 1979. Available from: https://doi.org/10.1007/978-3-642-45511-7.

[3] Romero C. *Handbook of critical issues in goal programming*. Elsevier; 1991. Available from: https://doi.org/10.1016/C2009-0-11180-7.

[4] Charnes A, Cooper WW, Ferguson RO. Optimal estimation of executive compensation by linear programming.

Management Science. 1955; 1(2): 138-151. Available from: https://doi.org/10.1287/mnsc.1.2.138.
[5] Tamiz M, Jones DF, El-Darzi E. A review of goal programming and its applications. *Annals of Operations Research*. 1995; 58: 39-53. Available from: https://doi.org/10.1007/BF02032309.
[6] Jones D, Tamiz M. *Practical goal programming*. New York: Springer; 2010. Available from: https://doi.org/10.1007/978-1-4419-5771-9.
[7] Min H, Storbeck J. On the origin and persistence of misconceptions in goal programming. *Journal of the Operational Research Society*. 1991; 42(4): 301-312. Available from: https://doi.org/10.1057/jors.1991.68.
[8] Larbani M, Aouni B. A new approach for generating efficient solutions within the goal programming model. *Journal of the Operational Research Society*. 2011; 62(1): 175-182. Available from: https://doi.org/10.1057/jors.2009.185.
[9] Cabllero R, Rey L, Ruiz F. Determination of satisfying and efficient solutions in convex multi-objective programming. *Optimization*. 1996; 37(2): 125-138. Available from: https://doi.org/10.1080/02331939608844204.
[10] Ignizio JP. The determination of a subset of efficient solutions via goal programming. *Computers & Operations Research*. 1981; 8(1): 9-16. Available from: https://doi.org/10.1016/0305-0548(81)90027-7.
[11] Tamiz M, Jones DF. Goal programming and Pareto efficiency. *Journal of Information and Optimization Sciences.* 1996; 17(2): 291-307. Available from: https://doi.org/10.1080/02522667.1996.10699283.
[12] Singh P, Pasha J, Moses R, Sobanjo J, Ozguven EE, Dulebenets MA. Development of exact and heuristic optimization methods for safety improvement projects at level crossings under conflicting objectives. *Reliability Engineering & System Safety*. 2022; 220: 108296. Available from: https://doi.org/10.1016/j.ress.2021.108296.
[13] Pasha J, Nwodu AL, Fathollahi-Fard AM, Tian G, Li Z, Wang H, et al. Exact and metaheuristic algorithms for the vehicle routing problem with a factory-in-a-box in multi-objective settings. *Advanced Engineering Informatics.* 2022; 52: 101623. Available from: https://doi.org/10.1016/j.aei.2022.101623.
[14] Dulebenets MA. Multi-objective collaborative agreements amongst shipping lines and marine terminal operators for sustainable and environmental-friendly ship schedule design. *Journal of Cleaner Production*. 2022; 342: 130897. Available from: https://doi.org/10.1016/j.jclepro.2022.130897.
[15] Si B, Wang J, Yao X, Shi X, Jin X, Zhou X. Multi-objective optimization design of a complex building based on an artificial neural network and performance evaluation of algorithms. *Advanced Engineering Informatics*. 2019; 40: 93-109. Available from: https://doi.org/10.1016/j.aei.2019.03.006.
[16] Dulebenets MA. A comprehensive multi-objective optimization model for the vessel scheduling problem in liner shipping. *International Journal of Production Economics*. 2018; 196: 293-318. Available from: https://doi.org/10.1016/j.ijpe.2017.10.027.
[17] Bringmann K, Friedrich T, Klitzke P. Two-dimensional subset selection for hypervolume and epsilon-indicator. In: *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation*. New York: Association for Computing Machinery; 2014. p.589-596. Available from: https://doi.org/10.1145/2576768.2598276.
[18] Bringmann K, Friedrich T, Klitzke P. Generic postprocessing via subset selection for hypervolume and epsilon-indicator. In: Bartz-Beielstein T, Branke J, Filipič B, Smith J. (eds.) *International Conference on Parallel Problem Solving from Nature*. Cham: Springer; 2014. p.518-527. Available from: https://doi.org/10.1007/978-3-319-10762-2_51.
[19] Amador F, Romero C. Redundancy in lexicographic goal programming: An empirical approach. *European Journal of Operational Research*. 1989; 41(3): 347-354. Available from: https://doi.org/10.1016/0377-2217(89)90255-5.
[20] Romero C. Carry on with redundancy in lexicographic goal programming. *European Journal of Operational Research*. 1994; 78(3): 441-442. Available from: https://doi.org/10.1016/0377-2217(94)90052-3.
[21] Zitzler E, Thiele L. Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach. *IEEE Transactions on Evolutionary Computation*. 1999; 3(4): 257-271. Available from: https://doi.org/10.1109/4235.797969.
[22] Hughes EJ. Evolutionary many-objective optimization: Many once or one many. In: *IEEE Congress on Evolutionary Computation*. IEEE; 2005. p.222-227. Available from: https://doi.org/10.1109/CEC.2005.1554688.
[23] Ruiz R, Maroto C. A comprehensive review and evaluation of permutation flowshop heuristics. *European Journal of Operational Research*. 2005; 165(2): 479-494. Available from: https://doi.org/10.1016/j.ejor.2004.04.017.
[24] Pinedo ML. *Scheduling: Theory, algorithms, and system*s. New York: Springer; 2012. Available from: https://doi.

org/10.1007/978-1-4614-2361-4.
[25] Mansouri SA, Aktas E, Besikci U. Green scheduling of a two-machine flowshop: Trade-off between makespan and energy consumption. *European Journal of Operational Research*. 2016; 248(3): 772-788. Available from: https://doi.org/10.1016/j.ejor.2015.08.064.
[26] Öztop H, Tasgetiren MF, Eliiyi DT, Pan Q-K, Kandiller L. An energy-efficient permutation flowshop scheduling problem. *Expert Systems with Applications*. 2020; 150: 113279. Available from: https://doi.org/10.1016/j.eswa.2020.113279.
[27] Saber RG, Ranjbar M. Minimizing the total tardiness and the total carbon emissions in the permutation flow shop scheduling problem. *Computers & Operations Research*. 2022; 138: 105604. Available from: https://doi.org/10.1016/j.cor.2021.105604.
[28] Stewart R, Raith A, Sinnen O. Optimising makespan and energy consumption in task scheduling for parallel systems. *Computers & Operations Research*. 2023; 154: 106212. Available from: https://doi.org/10.1016/j.cor.2023.106212.
[29] Amiri MF, Behnamian J. Multi-objective green flowshop scheduling problem under uncertainty: Estimation of distribution algorithm. *Journal of Cleaner Production*. 2020; 251: 119734. Available from: https://doi.org/10.1016/j.jclepro.2019.119734.
[30] Taillard E. Benchmarks for basic scheduling problems. *European Journal of Operational Research*. 1993; 64(2): 278-285. Available from: https://doi.org/10.1016/0377-2217(93)90182-M.
[31] Vallada E, Ruiz R, Framinan JM. New hard benchmark for flowshop scheduling problems minimizing makespan. *European Journal of Operational Research.* 2015; 240(3): 666-677. Available from: https://doi.org/10.1016/j.ejor.2014.07.033
[32] Moghaddam KS. Multi-objective preventive maintenance and replacement scheduling in a manufacturing system using goal programming. *International Journal of Production Economics.* 2013; 146(2): 704-716. Available from: https://doi.org/10.1016/j.ijpe.2013.08.027.

Appendix: Detailed computational results of the GPFSP application

Table A1. GP solutions generated from uniformly and normally distributed aspiration levels for 10 × 5-01 instance

Instance solutions (10 × 5-01)	Uniform aspiration levels (1)		Uniform GP solutions (2)		Normal aspiration levels (3)		Normal GP solutions (4)	
	C_{max}	TEC	C_{max}	TEC	C_{max}	TEC	C_{max}	TEC
1	581.782	2352.871	581.667	2337.757	580.922	2352.784	580.167	2351.27
2	584.665	2348.455	584.583	2345.964	584.949	2343.975	584.083	2342.406
3	588.5	2318.966	588.167	2316.725	588.321	2322.896	588.083	2320.267
4	590.548	2308.49	590	2293.979	591.564	2307.002	590.667	2304.126
5	594.814	2285.833	594	2283.973	596.345	2293.496	595.667	2292.925
6	598.977	2273.655	598.333	2272.936	599.582	2278.117	599.167	2263.819
7	602.264	2261.72	601.417	2257.908	602.703	2259.74	601.583	2257.968
8	606.695	2248.856	606.25	2240.398	605.487	2238.941	602.5	2233.174
9	610.035	2225.11	608	2224.807	609.848	2225.007	605.667	2191.312
10	613.926	2220.018	613.75	2214.288	613.28	2207.142	612.333	2205.168
11	618.165	2199.708	616.333	2182.056	616.319	2193.904	615	2193.473
12	619.53	2187.212	619.53	2185.706	619.864	2181.169	614.167	2163.751
13	622.871	2162.076	621.167	2158.59	625.092	2161.816	621.167	2144.588
14	628.443	2144.22	626.667	2140.028	627.251	2151.296	624.417	2149.572
15	632.105	2133.324	631.25	2130.735	631.42	2131.776	630.833	2114.663
16	635.183	2122.68	634.333	2120.552	634.607	2119.866	634.333	2119.189
17	638.464	2096.152	638.083	2082.2	638.82	2102.293	635.5	2101.674
18	643.638	2083.641	643.333	2064.347	641.619	2080.873	638.167	2080.447
19	647.551	2060.434	646.917	2059.492	646.382	2065.899	646.167	2063.802
20	649.618	2043.267	648.833	2031.986	650.035	2052.213	649.083	2052.027
21	653.889	2033.307	653.889	2012.923	653.931	2039.149	653.5	2031.022
22	657.43	2025.125	656.25	2014.264	655.417	2020.606	655.25	2002.763
23	661.99	1998.175	661.083	1996.712	661.65	2005.555	661.583	2003.967
24	662.569	1981.883	661.667	1971.992	663.752	1983.707	661.167	1969.342
25	666.074	1972.467	664.583	1969.725	667.498	1968.976	666.583	1959.988
26	671.958	1950.157	671.833	1946.888	671.416	1956.192	668.75	1949.93
27	674.495	1937.142	674.25	1936.677	676.181	1936.271	673.167	1863.321
28	679.064	1925.187	674.333	1852.836	678.06	1919.35	677.917	1915.793
29	681.149	1901.161	680.833	1877.01	682.467	1906.459	680.5	1905.658
30	684.571	1893.64	683.833	1892.356	686.517	1886.039	684.917	1884.743
31	689.613	1879.202	685.083	1859.09	690.777	1870.748	690.777	1862.624
32	692.106	1857.183	689.417	1829.119	692.932	1855.798	692.932	1851.662
33	697.139	1842.628	697.139	1841.586	696.815	1842.29	695.583	1825.692

Table A1. (cont.)

Instance solutions (10 × 5-01)	Uniform aspiration levels (1)		Uniform GP solutions (2)		Normal aspiration levels (3)		Normal GP solutions (4)	
	C_{max}	TEC	C_{max}	TEC	C_{max}	TEC	C_{max}	TEC
34	699.73	1823.042	697.417	1820.612	701.164	1827.272	701.083	1820.505
35	703.107	1801.864	702.083	1800.716	703.644	1807.311	703	1804.956
36	706.76	1800.062	705.333	1781.726	708.512	1796.795	704.75	1793.08
37	709.576	1781.543	709.576	1766.458	710.979	1777.518	708.333	1773.323
38	716.557	1753.629	715.333	1746.901	716.374	1758.509	713.667	1751.023
39	720.021	1738.561	719.583	1706.348	719.047	1741.731	714.083	1737.306
40	721.901	1728.968	713.75	1720.129	722.151	1724.301	716.167	1716.599
41	727.307	1706.468	726	1689.114	726.411	1713.787	725.75	1702.132
42	728.886	1702.436	725	1677.246	729.241	1694.849	724.75	1674.927
43	732.655	1675.102	732.333	1673.731	733.103	1680.838	731.917	1605.853
44	735.976	1670.687	734.583	1665.375	735.696	1665.551	732.5	1621.257
45	740.686	1640.042	736.083	1635.937	740.813	1650.3	720.833	1645.116
46	743.28	1630.154	742.083	1595.976	743.531	1628.247	740.5	1615.389
47	746.636	1608.82	740.167	1578.843	746.71	1612.632	743.667	1605.504
48	750.766	1591.682	750.766	1590.832	751.043	1593.784	751.043	1583.585
49	753.775	1583.4	753.667	1581.126	755.036	1581.598	747.5	1546.087
50	757.634	1568.084	757.083	1556.573	759.417	1566.303	754.917	1500.537
51	763.092	1548.43	763.083	1535.604	761.65	1551.065	761.5	1547.629
52	765.655	1534.234	755.917	1506.284	764.139	1531.458	763.333	1519.93
53	769.557	1509.853	766.833	1469.96	769.54	1512.405	769.083	1511.261
54	771.487	1496.002	763	1460.167	773.744	1502.306	770.583	1499.887
55	777.834	1478.422	770.833	1478.063	776.393	1481.643	770.5	1467.05
56	779.77	1470.942	776.75	1451.273	780.115	1469.635	775.25	1449.611
57	781.99	1446.273	781.583	1402.936	782.448	1456.237	778.083	1455.235
58	787.34	1431.715	780.25	1425.734	786.466	1434.546	786.25	1427.717
59	789.724	1416.094	789.724	1385.376	791.355	1422.603	784.833	1419.083
60	794.482	1400.869	786.333	1400.576	794.957	1402.755	788.583	1384.405
61	797.659	1389.428	796	1349.137	797.808	1388.777	797.5	1387.814
62	800.743	1367.874	794.167	1318.292	802.523	1371.344	802.523	1336.183
63	805.327	1362.385	805.083	1349.6	805.466	1356.91	795.5	1347.641
64	807.561	1346.325	803	1344.65	808.519	1337.9	806	1298.544
65	812.477	1319.023	811.25	1301.197	812.692	1327.018	812.167	1325.358
66	814.742	1300.599	809.417	1296.532	815.809	1311.081	814	1304.421
67	819.046	1291.175	819	1290.684	820.155	1284.252	819.167	1265.101
68	824.963	1274.623	816.667	1253.672	823.181	1280.95	818.667	1275.158

Table A1. (cont.)

Instance solutions (10 × 5-01)	Uniform aspiration levels (1)		Uniform GP solutions (2)		Normal aspiration levels (3)		Normal GP solutions (4)	
	C_{max}	TEC	C_{max}	TEC	C_{max}	TEC	C_{max}	TEC
69	826.621	1256.347	824.333	1252.907	826.1	1260.811	797.75	1253.203
70	830.817	1242.081	830.75	1226.81	831.206	1245.665	823.917	1228.776
71	834.83	1220.787	831	1218.352	834.453	1225.088	819.083	1218.105
72	837.631	1212.25	829.333	1183.772	838.276	1209.249	822.5	1185.952
73	841.732	1189.96	838.917	1173.131	842.312	1200.928	834.5	1192.603
74	843.719	1184.866	833.833	1183.671	845.215	1180.837	843.25	1164.97
75	847.982	1155.511	834	1145.617	848.781	1165.509	844.833	1165.358
76	853.651	1140.661	852.417	1140.058	852.888	1149.427	852	1148.865
77	854.924	1128.301	842	1113.319	855.142	1127.533	853	1124.275
78	859.736	1111.008	857.5	1110.49	860.185	1113.761	850.25	1113.47
79	862.849	1095.782	857.5	1095.048	863.443	1100.719	863	1098.116
80	865.546	1075.608	864	1082.828	866.169	1081.318	864	1082.827

Table A2. Subsets of efficient GP solutions computed from the previous table for 10 × 5-01 instance

Instance solutions (10 × 5-01)	Uniform GP solutions (1)		Uniform GP solutions (2)	
	C_{max}	TEC	C_{max}	TEC
1	864	1082.828	864	1082.827
2	857.5	1095.048	850.25	1113.47
3	842	1113.319	843.25	1164.97
4	834	1145.617	822.5	1185.952
5	829.333	1183.772	819.083	1218.105
6	816.667	1253.672	797.75	1253.203
7	809.417	1296.532	795.5	1347.641
8	794.167	1318.292	788.583	1384.405
9	789.724	1385.376	784.833	1419.083
10	781.583	1402.936	775.25	1449.611
11	780.25	1425.734	770.5	1467.05
12	763	1460.167	754.917	1500.537
13	755.917	1506.284	747.5	1546.087
14	740.167	1578.843	731.917	1605.853
15	736.083	1635.937	720.833	1645.116
16	725	1677.246	716.167	1716.599
17	719.583	1706.348	714.083	1737.306

Table A2. (cont.)

Instance solutions (10 × 5-01)	Uniform GP solutions (1)		Uniform GP solutions (2)	
	C_{max}	TEC	C_{max}	TEC
18	713.75	1720.129	708.333	1773.323
19	709.576	1766.458	704.75	1793.08
20	705.333	1781.726	703	1804.956
21	702.083	1800.716	695.583	1825.692
22	697.417	1820.612	673.167	1863.321
23	689.417	1829.119	668.75	1949.93
24	674.333	1852.836	661.167	1969.342
25	671.833	1946.888	655.25	2002.763
26	661.667	1971.992	653.5	2031.022
27	653.889	2012.923	649.083	2052.027
28	648.833	2031.986	646.167	2063.802
29	643.333	2064.347	638.167	2080.447
30	638.083	2082.2	630.833	2114.663
31	634.333	2120.552	621.167	2144.588
32	626.667	2140.028	614.167	2163.751
33	621.167	2158.59	605.667	2191.312
34	616.333	2182.056	602.5	2233.174
35	608	2224.807	599.167	2263.819
36	601.417	2257.908	595.667	2292.925
37	594	2283.973	590.667	2304.126
38	590	2293.979	588.083	2320.267
39	588.167	2316.725	584.083	2342.406
40	581.667	2337.757	580.167	2351.257