International Journal of Combinatorial Optimization Problems and Informatics, 13(2), May-Aug 2022, 1–13. ISSN: 2007-1558

www.editada.org

Analysis of Multi-objective Hyper-Heuristics Under Different Dynamic and Preferential Environments

Teodoro Macias-Escobar[1,2], Laura Cruz-Reyes[3], Bernabé Dorronsoro[2], Claudia Gómez-Santillán[3]*

[1] Department of Computing Science, Tijuana Institute of Technology, Av Castillo de Chapultepec 562, Tomas Aquino, Tijuana 22414, Mexico

[2] Departamento de Ingeniería Informática, Universidad de Cádiz, 11519 Puerto Real, Spain

[3] Graduate Program Division, Tecnológico Nacional de México, Instituto Tecnológico de Ciudad Madero, Cd. Madero 89440, Mexico

*Correspondence: teodoro_macias@hotmail.com

Abstract. The use of hyper-heuristics to solve dynamic multi-objective optimization problems (DMOPs) that incorporate decision-maker's preferences is a recently addressed research area. This paper presents the comparison and analysis of three hyper-heuristics to solve preferential DMOPs: The Dynamic Hyper-Heuristic with Plane Separation (DHH-PS) and two versions of the Dynamic Population-Evolvability based Multi-objective Hyper-Heuristic, incorporating Plane Separation in their process (DPEM-HH-PS). This work analyzes the performance of DHH-PS and both DPEM-HH-PS versions when solving DMOPs under different dynamic and preferential environments. This analysis aims to extend the study of DHH-PS and examine the capability of DPEM-HH-PS as an alternative to solve preferential DMOPs. DPEM-HH-PS exhibited suitability for type II DMOPs and randomly changing instances. DHH-PS presented a better performance for tri-objective DMOPs. The combination of genetic algorithms and differential evolution in DPEM-HH-PS proved effective for solving preferential DMOPs. DHH-PS and DPEM-HH-PS were capable of adapting to different preferential and dynamic environments.

Keywords: Hyper-heuristic, dynamic optimization, multi-objective optimization, preference incorporation.

Article Info

Received: August 27, 2021
Accepted: October 25, 2021

1 Introduction

Optimization problems can present different challenges to solve. Among these challenges, we can find problems that present multiple objectives to satisfy within an environment that changes over time. These problems are defined as multi-objective dynamic optimization problems (DMOP). Equation (1) presents their formal representation. Let $\vec{x}$ be a vector of decision variables, and *F* the set of *m* objective functions to optimize during a static time step *t*. Equality and inequality constraints are denoted by *g* and *h*, respectively.

$$\begin{aligned} \min F(\vec{x},t) &= \left\{ f_1(\vec{x},t), f_2(\vec{x},t), \ldots, f_m(\vec{x},t) \right\} \\ \text{s.t } & g(\vec{x},t) > 0,\ h(\vec{x},t) = 0\ . \end{aligned} \tag{1}$$

We can find several methods to solve DMOPs in the literature. Among those, the dynamic multi-objective evolutionary algorithms (DMOEA) are an alternative based on stochastic techniques to solve DMOPs. Various DMOEAs have proven to produce good-quality solutions for DMOPs with two or three objectives [1]. The growing interest in DMOPs and alternatives to solve them present a new area of research within optimization, focused on dynamic environments [2]. This area focuses on identifying the best alternatives to solve a set of DMOPs. Recently, researchers have detected that hyper-heuristics are potential

alternatives to solve dynamic optimization problems. A hyper-heuristic is defined as a methodology that selects or generates heuristics, named low-level heuristics (LLH), to solve problems [3],[4].

One of the main benefits of hyper-heuristics is to take benefit of the strengths of specific heuristics to cover the weaknesses of others, allowing the hyper-heuristic to obtain better results in comparison to each heuristic if used individually. A choice-function based hyper-heuristic for multi-objective optimization, defined as HH_CF, uses meta-heuristics as LLHs to solve MOPs, the heuristic selection is carried out using a choice function supported by performance metrics [5]. The work of Filho et al. [6] proposes a hyper-heuristic for solving MOPs with preferences defined by a decision maker (DM), the expert (or group of experts) in charge of selecting the better-suited alternative according to his/her/their preferences, by incorporating reference points. Hyper-heuristics have also been used for DMOPs. The study by van der Stockt and Engelbrecht [7] analyzes the use of hyper-heuristics supported by meta-heuristics as LLHs to solve DMOPs.

Hyper-heuristics based on LLH selection can be divided into two stages: *heuristic selection*, which compares the available LLHs and selects one to be applied for a number of iterations, and *move acceptance*, in which the hyper-heuristic evaluates the solution obtained by the LLH and determines whether it should replace the current solution [8]. The literature provides several methods to carry out the *heuristic selection*. Among these methods, random selection, greedy techniques, or permutations are popular options, among others [9]. In some cases, performance metrics or fitness landscape analysis (FLA) methods gather enough information to make an appropriate selection.

Another challenge that arises when solving optimization problems is the presence of a DM. When a DM inserts its preferences into a problem, the search should focus on the space that satisfies those preferences, called a region of interest (ROI). Therefore, it is logical to think that incorporating preferences in a DMOP can increase its complexity. There exist multiple approaches to incorporate preferences into a multi-objective problem using different approaches [10],[11],[12]. However, most have been tested only in static environments, and an analysis of their effectiveness is required when facing an environment that faces changes [2].

This paper proposes to use Plane Separation (PS) [13], a preference-incorporation method previously tested to solve DMOPs with preferences within hyper-heuristics. These hyper-heuristics use DMOEAs as LLHs. Performance metrics and FLA methods support their *heuristic selection* method. Every proposed hyper-heuristic is tested under multiple dynamic and preferential environments, and the results are compared and analyzed. This work aims to explore the capability of the Dynamic Population-Evolvability based Multi-objective Hyper-Heuristic (DPEM-HH) to solve preferential DMOPs. This paper also seeks to extend the study of the Dynamic Hyper-Heuristic with Plane Separation (DHH-PS) under untested instances and conditions. Both hyper-heuristics are further described in Section 2.

The main contributions of this work are as follows: i) The use of population evolvability, an FLA method, to solve preferential DMOPs, ii) the incorporation of PS in several hyper-heuristics, iii) the use of DMOEAs within hyper-heuristics in which they have not been tested and iv) the analysis of the effects of PS, performance metrics and population evolvability within a hyper-heuristic under different dynamic and preferential environments.

The remaining of this document is structured as follows. Section 2 presents previous works related to this research and the definitions necessary to understand it. Then, Section 3 contains the design of the proposed work. Next, Section 4 presents the experimentation and analysis of the obtained results. Finally, Section 5 presents the conclusions and possible future work.

2 Background

2.1 Dynamic multi-objective evolutionary algorithms with preference incorporation

The incorporation of preferences in a DMOP is a situation that may seem common in real life. However, the complexity of both representing and solving DMOPs is a cause of the limited research in this field. To our knowledge, a dynamic version of the Non-dominated Sorting Genetic Algorithm (DNSGA-II) [1] is the first DMOEA proposed to solve preferential DMOPs. Using the structure of NSGA-II [14] along with change detection and adaptation mechanisms and the use of a preferential weights vector defined *a priori* by a DM. DNSGA-II takes from the set of feasible solutions the one that comes closest to the established preferences. However, the preferences of the DM do not affect the solution process, but only the final decision, so all solutions within the feasible search space are equally evaluated.

Multiple works consider this situation. Roy and Mehnen present a version of DNSGA-II that replaces the objective functions with desirability functions [15]. These functions are specifically adapted for a given DMOP to incorporate the preferences defined

by the DM. In a more recent case, reference points are used to focus the DNSGA-II search towards a generated ROI based on the Pareto front obtained and the set of preferences given by the DM, which determines preferential ranges for each objective [13].

Reference points are among the most commonly used preference incorporation methods to solve preferential DMOPs. Some examples within the literature include the Sphere-dominance Preference Immune-inspired Algorithm (SPIA), a DMOEA that generates a hypersphere within the search space of the objective functions [16]. This hypersphere centers around a reference point defined by the DM. SPIA inserts a new layer of dominance since the solutions that are within the hypersphere dominate the solutions that are outside, having a higher priority than Pareto dominance. The Interactive Dynamic Multi-Objective Decision Making (InDM2) [17] is another reference-point based proposal, which allows the user to incorporate preferences interactively. During the solution process, the DM can modify or incorporate reference points, seeking to focus the search towards a specific area of the feasible solution space. InDM2 has been incorporated and tested in two different DMOEAs: the reference-point based NSGA-II (R-NSGA-II) [18] and the Weighting Achievement Scalarizing Function Genetic Algorithm (WASF-GA) [19].

The Plane Separation preference incorporation method (PS) [13], is a recent proposal also based on reference points. PS uses a set of upper and lower limits for each objective defined by the DM, these limits represent the maximum and minimum preference setup for that objective. PS generates a set of reference points using these values and a population generated by a DMOEA. These points allow PS to generate planes within the search space. The planes represent the ROI and areas with different degrees of separation from the ROI. Then, PS sets each solution to a plane based on its location. The population replacement technique of the DMOEA in use chooses the best individuals in each plane, based on a contribution percentage, to generate the population for the next generation. DNSGA-II and a dynamic version of the Generalized Differential Evolutionary algorithm (DGDE3) [20] have incorporated and tested PS with satisfying results, outperforming a reference-point based preference incorporation method [13].

2.2 Plane Separation method

Algorithm 1 shows the process performed by PS. First, the minimum and maximum values per objective i (min_i and max_i) obtained from the joint population Q are calculated, Q represents the parent and offspring merged population obtained by the heuristic H.

Algorithm 1. Plane Separation

```
Input: R, W, C, Q, H
Output: P
 1: {max, min} ← CalculateMaxMin(Q)
 2: i ← 1
 3: while i ≤ m do
 4:    d ← Fmax_i − Fmin_i
 5:    L_{1,min_i} ← Fmin_i + (W_{i,min} * d)
 6:    L_{1,max_i} ← Fmin_i + (W_{i,max} * d)
 7:    i ← i+1
 8: end while
 9: j ← 2
10: while j < |S| do
11:    i ← 1
12:    while i ≤ m do
13:       L_{j,min_i} ← L_{1,min_i} - (L_{1,min_i} * R_{j,i})
14:       L_{j,max_i} ← L_{1,max_i} + (L_{1,max_i} * R_{j,i})
15:       i ← i+1
16:    end while
17:    j ← j+1
18: end while
19: S ← InsertInPlanes(L, Q)
20: P ← GetNewPopulation(S, C, H)
```

The vector L contains the minimum and maximum values per objective for each plane. L_1,min_i and L_1,max_i denote the minimum and maximum values for the i-th objective of the first plane, which represents the ROI. These values are calculated considering the distance between $Fmin_i$ and $Fmax_i$, denoted as d, and a preferential weight vector W set by the DM. After obtaining the ROI,

the extreme values per objective of each next plane are calculated and assigned to *L*. These values are obtained considering the extreme values of the ROI and an expansion percentage $R_{j,i}$ corresponding to the *i-th* objective in the *j-th* plane.

After generating the planes, *Q* splits into a set of subpopulations *S*, considering the position of each solution within the search space. Each subpopulation in *S* contains the solutions in a plane. Finally, according to the DMOEA in use, the best solutions of each plane are inserted into the new population *P*. A contribution percentage vector *C* defines the number of solutions contributed by each plane towards the new population.

2.3 Dynamic hyper-heuristics

One of the main intentions of using hyper-heuristics to solve problems is to combine the strengths of different strategies to cover their respective weaknesses and offer higher quality results than those the strategies could deliver individually.

The application of hyper-heuristics to solve dynamic optimization problems (DOPs) was firstly proposed, to our knowledge, by Ozcan et al. [21]. Their work applies a greedy-based *heuristic selection* method to six different heuristics based on bit-flip mutation, and the Davis bit-escalation method [22] and five different mutation percentages are used as the LLH set. Their proposal uses All Moves [9] as the *move acceptance* criterion, which accepts all new solutions presented by the selected LLHs.

The complexity of LLHs is not limited. In some cases, meta-heuristics have been used as LLHs. The Adaptive Hill-Climbing method (AHC) [23] is a local search strategy that selects between two different hill-climbing methods. The *heuristic selection* is roulette-based, setting percentages according to the improvement or deterioration produced by each method.

Another hyper-heuristic, named HH [24], uses three different population-based meta-heuristics as LLHs to solve dynamic multi-dimensional knapsack problems. HH uses a roulette-based heuristic selection method in which each LLH is assigned a percentage based on its performance when applied to the current problem. HH adopts AHC within its change adaptation method. Both HH and AHC use a change adaptation method based on the insertion of random immigrants seeking to maintain population diversity.

The Heterogeneous Meta-Hyper-Heuristic (HMHH) [7] is a proposal in which an LLH is assigned to each solution of a population, applying it for a set of test iterations. Then, an improvement-based selection operator reassigns LLHs to each solution. It uses five meta-heuristics as LLHs, including genetic, differential evolution, and particle swarm optimization algorithms. Experiments were performed using ten different *heuristic selection* methods based on permutations, randomness, roulette, and ant-colony algorithms. Each variation produced satisfactory results under different dynamic environments.

To our knowledge, most of the proposed hyper-heuristics to solve DOPs within the literature focus solely on single-objective problems. Although some hyper-heuristics focus on DMOPs, most of those proposals focus on solving scheduling problems [25]. The Dynamic Population-Evolvability based Multi-objective Hyper-Heuristic (DPEM-HH) [26] is a hyper-heuristic that tackles DMOPs outside that environment. This hyper-heuristic incorporates an FLA method, population evolvability, to obtain information regarding both the DMOP to solve and utilized LLHs, seeking to select the most appropriate heuristic. DPEM-HH used three different variations of DNSGA-II as LLHs and was tested under two different *heuristic selection* methods (greedy and choice function) and an All Moves *acceptance criterion* [6].

Recently, a proposal presents the use of hyper-heuristics as alternatives to solve DMOPs with defined preferences. Dynamic Hyper-Heuristic with Plane Separation (DHH-PS) [13] is a hyper-heuristic that incorporates the PS method into its process. The authors carried out a set of experiments in multiple DMOPs under two different preferential environments in their work. DHH-PS uses two versions of DNSGA-II and DGDE3 as LLHs. The obtained results show that the application of hyper-heuristics to solve preferential DMOPs has the potential to produce high-quality results if they use the appropriate LLHs.

Figure 1 shows the DHH-PS flowchart. The first step consists of generating a random initial population, after which the LLH selection process is performed. In this step, each LLH is given a copy of the initial population and is run for a number of test generations. After these generations, the resulting populations of each heuristic are evaluated and the LLH with the best result is chosen based on the defined LLH selection method.

The selected LLH is executed until one of two conditions is reached. The first condition considers whether the environment has undergone a change. If true, then the last population obtained by the LLH is compared against the current population based on the determined acceptance criterion. The new population replaces the current population if the criterion accepts the change. The second condition is determined by the DHH-PS stopping criterion, which is defined by a given number of elapsed generations.

In this case, as in the first stopping condition, the last population obtained by the LLH in use is compared with the current population using an acceptance criterion to determine whether the new population is accepted and replaces the current population.

DHH-PS uses the Plane Separation method within two processes. The first stage is during the LLH selection, in which PS is incorporated within each heuristic. The second point at which PS is used in DHH-PS is when an LLH has been selected. The selected heuristic has PS incorporated within its process when executed.

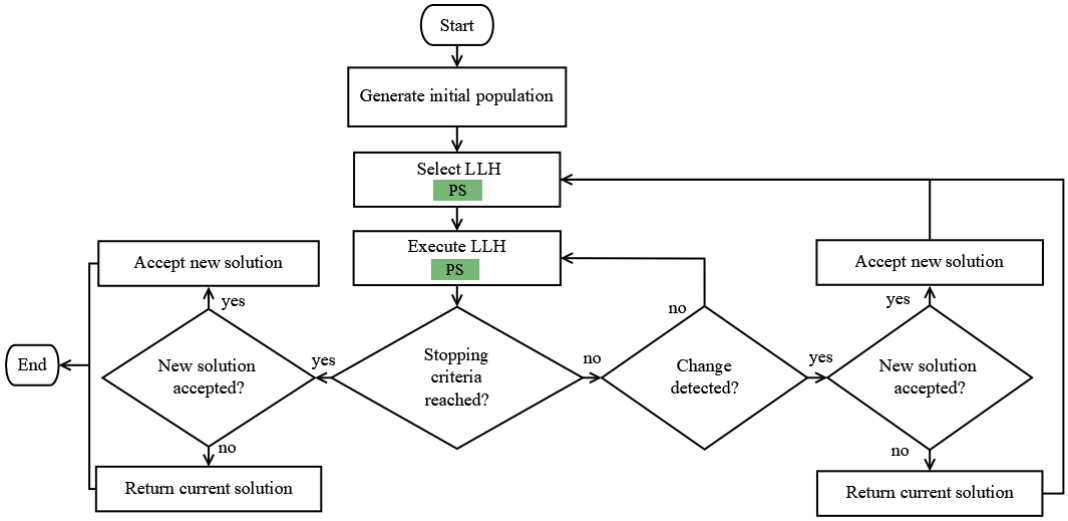

Fig. 1. DHH-PS

2.4 Population evolvability

Usually, FLA methods focus on the properties of the problem. However, many of these methods ignore the properties of the utilized algorithm. Both factors should be considered because certain algorithms could easily solve a complex problem, and an easy problem could be complicated for some algorithms [27].

Evolvability allows evaluating both criteria. It can be defined as "*the ability of a population to produce variants fitter than any yet existing*" [28]. This concept involves both, the problem and the algorithm used. Population evolvability (*evp*) extends this notion by measuring the ability of a population to evolve positively considering two factors: i) the probability of the population to evolve positively and ii) the evolution improvement degree. Equation (2) shows the evaluation of this FLA method.

$$evp(P_i) = \begin{cases} \dfrac{\sum_{P_{ij} \in N^+(P_i)} \dfrac{\left|f^b(P_i) - f^b(P_{ij})\right| / NP}{\sigma(f(P_i))}}{|N(P_i)|} & \text{if } |N^+(P_i)| > 0 \\ 0 & \text{if } |N^+(P_i)| = 0 \end{cases} \tag{2}$$

Let P_i be the current population, $f^b(P)$ represents the best fitness value obtained in a population P, $\sigma(f(P_i))$ is the standard deviation of the fitness values of the population P_i, NP is the size of P_i, $N(P_i)$ is a set of one-step offspring populations of P_i defining its neighborhood, $N^+(P_i)$ is a subset of $N(P_i)$ containing only neighbors with better fitness than P_i. For minimization problems this is denoted as $N^+(P_i) = \{P_{ij} \mid P_{ij} \in N(P_i), f^b(P_{ij}) < f^b(P_i)\}$ in which $j = 1,\ldots,| N^+(P_i)|$. The range of *evp* is $[0,+\infty)$. Higher values mean that the algorithm has better capability to produce high-quality solutions as it evolves the population.

3 Proposed work

As can be deduced from Section 2, research on the resolution of DMOPs with preference incorporation is an area under recent consideration, and there still is an extensive research field to explore. This paper proposes the use of two recently proposed hyper-

heuristics, DHH-PS and DPEM-HH. This research seeks to find new conclusions regarding the performance of DHH-PS by testing the same configuration proposed in [13] for tri-objective problems and different dynamic environments.

The *heuristic selection* method in DHH-PS follows the choice function (*CF*) presented in Maashi et al.'s work [5]. This function, shown in Equation (3), obtains the *CF* value corresponding to each heuristic *r* based on two criteria and a control variable α. c_1 matches a two-stage ranking in which the results of each LLH are compared based on a set of performance metrics. c_2 corresponds to the number of generations elapsed since the previous selection of *r*.

$$CF(r) = \alpha c_1(r) + c_2(r) \tag{3}$$

Equation (4) shows how to obtain c_1. Let *N* be the number of LLHs; $Freq_{rank}(r)$ is the position of *r* in a ranking based on the number of times *r* obtained the best value for each metric, and $\Theta_{rank}(r)$ is the position of *r* for the metric Θ. DHH-PS uses variable space generational distance [29] as Θ. In the case of DPEM-HH, this work seeks to analyze the effects of adding a preference incorporation method in its process to solve DMOPs, specifically PS.

$$c_1 = 2*(N+1) - Freq_{rank}(r) + \theta_{rank}(r) \tag{4}$$

3.1 Preference incorporation in DPEM-HH

Algorithm 2 presents DPEM-HH-PS, a proposed version of DPEM-HH that uses PS to allow the hyper-heuristic to incorporate DM's preferences. DPEM-HH-PS follows the same process defined in its original work. However, the hyper-heuristic adds conditionals within and outside the *heuristic selection* process to incorporate PS (lines 8 and 22). This proposal incorporates PS within the hyper-heuristic following the structure presented in Figure 1.

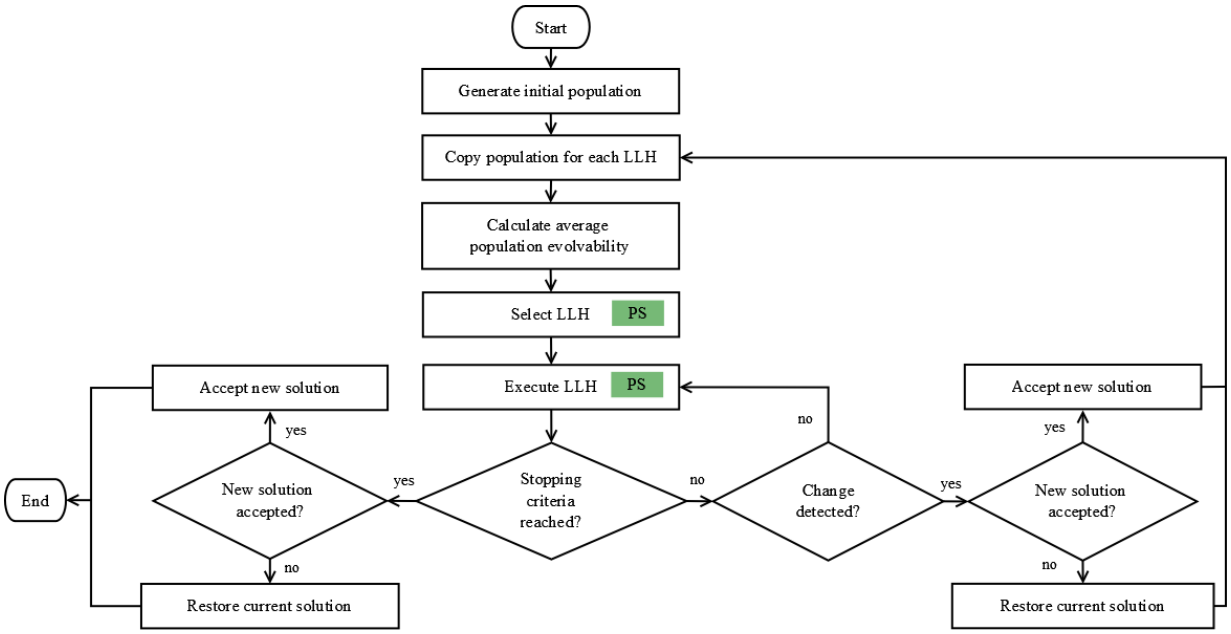

Fig. 2. DPEM-HH-PS

Figure 2 presents the DPEM-HH-PS flowchart. As can be seen, it follows a process that has similarities with DHH-PS but includes two additional steps. First, each LLH copies the current population before the heuristic selection process. Then, each LLH calculates its average population evolvability after executing for a set of sample generations. This value, supported with PS, carries out the LLH selection.

Let $P_{evp,i}$ be the population generated by the *i*-th LLH, evp_i represents the population evolvability obtained for each LLH after τ_{test} test generations have elapsed and *MET* the set of values corresponding to the performance metrics used for the *heuristic selection*.

DPEM-HH-PS performs the process defined in the original work. However, each LLH executes PS within its process, following the design of DHH-PS. At the end of the selection process, the last population of the chosen LLH is set as the current population. The selected LLH is used (utilizing PS or its replacement method based on PS-control variable τ_{ps}) until the next period.

Algorithm 2. DPEM-HH-PS
Input: τ_{ps}, τ_{test}, *LLH*, *R*, *W*, *L*, *C*
Output: *P*

```
P ← CreateInitialPopulation()
while termination criteria have not been reached
 if change is detected or first generation then
   for i = 1 to |LLH|
     P_evp,i ← CopyPopulation(P)
     for j = 1 to τ_test
       Q ← CreateJointPopulation(LLH_i, P_evp,i)
       if j = 1 or j = τ_ps then
         P_evp,i ← PS(R, W, C, Q, LLH_i)
       else
         P_evp,i ← Replacement(Q, LLH_i)
       NP ← CreateNeighborhood(LLH_i, P_evp,i)
       evp_i ← evp_i + CalculateEvolvability(P_evp,i, NP)
       end if
     end for
     evp_i ← evp_i / τ_test
   end for
   MET ← MetricEvaluation(P_evp)
   LLH_chosen ← SelectLLH(LLH, evp, MET)
   P ← SetPopulation(LLH_chosen, P_evp,chosen)
 else
   if i mod τ_ps = 0 then
     Q ← CreateJointPopulation(P, LLH_chosen)
     P ← PS(R, W, C, Q, LLH_chosen)
   else
     P ← ExecuteLLH(P, LLH_chosen)
   end if
 end if
end while
```

DPEM-HH-PS uses *CF* as a *heuristic selection* method, with *evp* replacing Θ. A single value represents the value of *evp*. Thus, DPEM-HH-PS uses a weighted sum method to add the values of the objective functions, following the original design.

3.2 Versions of DPEM-HH-PS

This paper proposes two different versions of DPEM-HH-PS based on the set of DMOEAs used as LLHs. The first one is based on the original DPEM-HH configuration, which uses three different versions of DNSGA-II (DNSGA-II-A, DNSGA-II-B, and DNSGA-II-AB). Each version of DNSGA-II has a different change adaptation method. DNSGA-II-A replaces a subset of the population for new randomly-generated solutions. DNSGA-II-B replaces mutates the solutions from said subset. Lastly, DNSGA-II-AB combines both methods. The second version of DPEM-HH-PS utilizes the set of LLHs used by DHH-PS (DNSGA-II-A, DNSGA-II-AB, and DGDE3). In this case, we would like to explore the effects of using a differential evolution algorithm as one of the LLHs.

4 Experimentation and discussion

4.1 Performance metrics

As mentioned in Section 3, the choice-function heuristic selection method requires a set of metrics that evaluate the performance of each algorithm to define the value of c_1. This paper uses the following performance metrics to calculate c_1, following Equation

(4). These metrics only consider solutions within the ROI. Since the optimal Pareto front (*POF**) of the DMOPs used in this work can be mathematically obtained, we use only the subset of solutions corresponding to the ROI for the *POF** when evaluating. Then, the subset of the Pareto front obtained by an algorithm (*POF*) within the ROI is compared against the *POF**.

Ratio of non-dominated solutions in ROI (RNI_{ROI}). Based on the original RNI equation [30]. This metric obtains a ratio regarding the non-dominated solutions within the ROI with the size of the population *P* at a time step *t*. The range of RNI is [0,1], where a higher value, closer to 1, is better.

$$RNI_{ROI,t} = \frac{|POF_{ROI,t}|}{|P_t|} \tag{5}$$

Hyper-volume ratio (*HVR*) [31]. It measures the ratio between the *POF** and the *POF* by comparing the hyper-volume obtained by both using an equal reference point. Equally than RNI, the range of HVR is [0,1], and a higher value equals better performance.

$$HVR_t = \frac{HV(POF_t)}{HV(POF_t^*)} \tag{6}$$

Variable space generational distance (*VD*) [29]. Measures the minimum distance between each solution from the *POF* with respect to *POF**. VD uses Euclidean distance $d(v, POF^*_t)$ to evaluate the closeness of a solution from the *POF* (v) to the closest solution from *POF**. The range of VD is $[0,\infty)$, where a lower value is better, as it means that the distance between the optimal and the obtained Pareto fronts is smaller.

$$VD_t = \frac{\sqrt{|POF_t| \sum_{v \in POF} d(v, POF_t^*)^2}}{|POF_t|} \tag{7}$$

Inverted generational distance (*IGD*) [32]. A variation of generational distance that reverses the roles of *POF* and *POF**. IGD measures the minimum Euclidean distance between each solution from the *POF** with respect to POF. IGD has a $[0,\infty)$ range. A lower IGD value means less distance from *POF** to *POF*. Therefore, a lower IGD represents better performance.

$$IGD_t = \frac{\sum_{v \in POF^*} d(v, POF)^2}{|POF_t^*|} \tag{8}$$

DHH-PS uses these metrics within its choice function, using *VD* as Θ. Meanwhile, DPEM-HH-PS combines these metrics with *evp*, using said FLA method as Θ. We use and present the *VD* obtained by each hyper-heuristic for performance analysis and comparison between the tested hyper-heuristics. This metric has been previously used in works related to dynamic optimization with preference incorporation [13,14]. The reason for using *VD* is that this metric can provide information regarding closeness and spread with respect to a specific subset of the *POF**, which represents the DM's preferences.

4.2 Experimental design

We used six preferential test instances to evaluate and compare the performance of the proposed hyper-heuristics. FDA1, FDA4, and FDA5 from the FDA set [33] and the dMOP1, dMOP2, and dMOP3 problems from the dMOP set [29]. This set of instances includes bi- and tri-objectives dynamic optimization problems that offer different challenges regarding how the optimal solution set and Pareto front change with time. Equation (9) shows the handling of the time variable *t* for DMOPs. Let τ be the current generation counter, τ_τ the change frequency (number of generations before a change) and n_τ the change severity (number of distinct time step that the DMOP will present).

$$t = \frac{1}{n_\tau}\left\lfloor \frac{\tau}{\tau_\tau} \right\rfloor \tag{9}$$

The stopping condition for each DMOP was set to $\tau_\tau * n_\tau$ generations, setting $n_\tau = 10$ and $\tau_\tau = \{10,25,50\}$. The τ_τ vector indicates that three different values for τ_τ are tested for every DMOP. These tests are done to analyze the behavior of each proposed hyper-heuristic under different dynamic and preferential environments. The maximum population size for each hyper-heuristic (including each LLH) is set to 100.

The values set on the preferential weight vector W are defined for each instance to have the optimal solutions inside the ROI within a feasible area of the solution search space. This work follows the preferential environments set in [13], setting the ROI towards either low or high values for the first objective.

The first ROI sets a normalized value in W as $W_{f1} = [0.1,0.4]$; $W_{f2} = [0.3,0.7]$ for FDA1 and dMOP3; $W_{f2} = [0.65,1.0]$ for dMOP1 and dMOP2 and $W_{f2} = [0.0,1.0]$, $W_{f3} = [0.0,1.0]$ for FDA4 and FDA5. This preferential configuration focuses its search on the minimum values of f_1, the values for the other objectives consider their feasible areas with respect to the values defined on W_{f1}.

The second ROI sets $W_{f1} = [0.6,0.9]$; $W_{f2} = [0.05,0.3]$ for FDA1 and dMOP3; $W_{f2} = [0.1,0.65]$ for dMOP1 and dMOP2, and lastly $W_{f2} = [0.0,0.8]$, $W_{f3} = [0.0,0.8]$ for FDA4 and FDA5. This preferential configuration focuses on the maximum values of f_1, the other objectives are defined in consideration of their feasible areas with respect to the values defined on W_{f1}.

The design of PS follows the structure defined in its original work [13]. PS generates three planes, dividing the search space into four different areas, where the last area consists of all the feasible space beyond the last plane, each one with different closeness towards the ROI. The expansion matrix is set as $R = \{0\%, 15\%, 30\%\}$ for each dimension, and the contribution percentage by area is set as $C = \{80\%, 15\%, 3\%, 2\%\}$ (from closest to farthest area regarding the defined ROI).

DHH-PS, and the first variant of DPEM-HH-PS, called DPEM-HH-PS-A, use as their LLH set two versions of DNSGA-II (A and AB) and DGDE3. The second variant, DPEM-HH-PS-B, uses three DNSGA-II (A, B, and AB) versions, following the LLH structure proposed in [26]. The properties of the operators of each LLH are as follows.

The three versions of DNSGA-II use binary tournament selection, simulated binary crossover, and polynomial distribution as selection, crossover, and mutation operators, respectively [34]. Crossover and mutation probabilities are set to 0.9 and 1/n, let n be the decision variable vector size. Distribution indexes for crossover and mutation are set to 10 and 20. DGDE3 follows a DE/1/rand/bin structure. Both crossover and scaling factor are set to 0.5. The parameter values were set after exhaustive preliminary experimentations.

When a change is detected, 20% of the population is replaced based on the change adaptation method of the LLH currently selected. DNSGA-II-A and DGDE3 generate new randomly generated solutions. DNSGA-II-B mutates a subset of solutions to obtain new solutions, and DNSGA-II-AB combines the process of DNSGA-II-A and DNSGA-II-B, replacing half of the new subset by random solution and the other by mutated solutions.

The tested hyper-heuristics use choice function as their *heuristic selection* method using the metrics mentioned at the end of Section 4.1. All new solutions are accepted, following the *All Moves* move acceptance criterion [6]. The number of sample generations τ_{test} is set as 20% of τ_τ. Finally, the control variable for the choice function method is set to $\alpha = \tau_\tau$.

The tested algorithms were implemented using the jMetal 5.2 framework and executed 30 times for each DMOP on a server equipped with an AMD Ryzen 5 2.0 GHz processor and 12GB RAM.

4.3 DPEM-HH-PS versus DHH-PS

Table 1 presents offline median and standard deviation regarding *VD* obtained by each hyper-heuristic for each DMOP under all three dynamic environments while having a preferential setup of $W_{f1} = [0.1,0.4]$ for all instances. Table 2 shows the results obtained for the preferential configuration $W_{f1} = [0.6,0.9]$. Friedman aligned ranks test with a 0.05 significance level [35], and Holm post-hoc method [36] are used to determine if the best solution obtained by a hyper-heuristic is significantly superior to others. The best value is highlighted in bold and compared to the values obtained by the other hyper-heuristics. If the difference between both values is statistically significant, * is added after the value of the respective outperformed hyper-heuristic.

As the tables show, the three hyper-heuristics distribute among themselves the best results for the DMOPs tested. However, it is possible to analyze the results and identify the dynamic and preferential scenarios better suited for each hyper-heuristic to solve DMOPs. Each alternative presents a better performance in a specific subset of instances.

Table 1. Offline *VD* median and standard deviation with W_{fl} = [0.1,0.4]

τ_τ	DMOP	DHH-PS	DPEM-HH-PS-A	DPEM-HH-PS-B
10	FDA1	1.460e+00(2.282e-01)	1.467e+00(2.561e-01)	1.973e+00(3.045e-01)*
	FDA4	4.273e-01(8.853e-02)*	3.122e-01(6.394e-02)	1.536e+00(1.520e-01)*
	FDA5	3.853e-01(8.892e-02)*	2.966e-01(6.473e-02)	5.206e-01(9.022e-02)*
	dMOP1	2.319e+00(8.881e-01)	2.658e+00(1.151e+00)*	3.906e+00(1.483e+00)*
	dMOP2	4.358e-01(1.142e-01)	4.655e-01(1.350e-01)	6.790e-01(1.290e-01)*
	dMOP3	2.301e-01(4.211e-02)	2.238e-01(6.034e-02)	3.569e-01(5.845e-02)*
25	FDA1	1.532e-01(2.734e-02)	1.680e-01(4.569e-02)*	1.574e-01(2.048e-02)
	FDA4	1.072e-01(1.682e-02)	1.164e-01(2.394e-02)	1.417e-01(2.405e-02)*
	FDA5	1.158e-01(1.238e-02)	1.114e-01(1.090e-02)	1.371e-01(1.300e-02)*
	dMOP1	2.747e-02(1.244e-02)	3.077e-02(1.121e-01)*	1.347e-01(1.219e-01)*
	dMOP2	3.661e-02(2.907e-03)*	3.890e-02(3.224e-03)*	3.426e-02(3.889e-03)
	dMOP3	2.347e-02(1.820e-03)	2.305e-02(3.141e-03)	7.766e-02(3.296e-02)*
50	FDA1	3.101e-02(6.285e-03)*	3.274e-02(1.008e-02)*	2.787e-02(1.811e-03)
	FDA4	8.911e-02(2.880e-03)	9.040e-02(2.534e-03)	9.062e-02(3.481e-03)*
	FDA5	8.752e-02(2.661e-03)	8.926e-02(2.826e-03)	8.873e-02(3.588e-03)
	dMOP1	5.143e-03(1.056e-03)	5.390e-03(2.579e-03)	8.742e-03(4.928e-03)*
	dMOP2	9.783e-03(1.196e-03)*	9.722e-03(1.162e-03)*	8.972e-03(5.889e-04)
	dMOP3	6.996e-03(2.973e-04)	6.829e-03(2.574e-04)	7.038e-03(5.461e-03)*

Note: The best value is highlighted in bold, * indicates that the best hyper-heuristic is significantly better than that specific hyper-heuristic

Table 2. Offline *VD* median and standard deviation with W_{fl} = [0.6,0.9]

τ_τ	DMOP	DHH-PS	DPEM-HH-PS-A	DPEM-HH-PS-B
10	FDA1	2.185e+00(2.889e-01)	1.994e+00(3.201e-01)	2.987e+00(4.744e-01)*
	FDA4	9.045e-02(2.264e-02)	9.416e-02(3.216e-02)	2.506e-01(4.097e-02)*
	FDA5	8.410e-01(2.283e-01)	8.639e-01(3.170e-01)	1.435e+00(2.333e-01)*
	dMOP1	1.050e+00(2.548e-01)	1.130e+00(3.089e-01)	1.279e+00(4.795e-01)*
	dMOP2	2.446e-01(3.911e-02)	2.420e-01(3.844e-02)	2.867e-01(5.926e-02)*
	dMOP3	2.578e-01(4.865e-02)	2.747e-01(4.590e-02)	4.595e-01(1.038e-01)*
25	FDA1	2.305e-01(4.860e-02)	2.351e-01(4.276e-02)	2.249e-01(4.215e-02)
	FDA4	1.402e-01(2.503e-02)	1.734e-01(7.943e-02)*	3.768e-01(8.484e-02)*
	FDA5	1.441e-01(1.020e-01)	1.642e-01(1.279e-01)	2.781e-01(5.972e-02)*
	dMOP1	1.058e-01(1.449e-01)*	6.005e-02(6.643e-02)	1.360e-01(8.296e-02)*
	dMOP2	2.831e-02(4.908e-03)*	2.699e-02(4.304e-03)	2.798e-02(3.704e-03)
	dMOP3	3.066e-02(4.480e-03)	3.100e-02(6.230e-03)	1.053e-01(5.268e-02)*
50	FDA1	3.989e-02(6.420e-03)	4.145e-02(9.943e-03)*	3.639e-02(1.886e-02)
	FDA4	9.346e-02(4.421e-03)	9.682e-02(9.856e-03)	1.087e-01(1.127e-02)*
	FDA5	9.267e-02(4.951e-03)	9.221e-02(6.135e-03)	1.043e-01(1.167e-02)*
	dMOP1	4.954e-03(2.742e-02)	5.007e-03(3.605e-02)	5.580e-03(4.324e-02)
	dMOP2	7.503e-03(1.139e-03)*	7.024e-03(3.349e-04)	7.383e-03(2.366e-04)*
	dMOP3	8.496e-03(5.550e-04)	8.083e-03(5.456e-04)	7.968e-03(6.799e-03)

Note: The best value is highlighted in bold, * indicates that the best hyper-heuristic is significantly better than that specific hyper-heuristic

Initially, DHH-PS offers a better performance in environments with a low τ_τ value (i.e., 10). However, there are cases where a τ_τ increase presents more benefits in favor of DPEM-HH-PS. A particular case can be seen in the results presented in FDA1, a type I DMOP. For this instance, DPEM-HH-PS-B outperforms the other two hyper-heuristics in τ_τ = 50 in both dynamic environments, as shown in Tables 1 and 2. The improvement in the quality of the solution of DPEM-HH-PS could be because a greater number

of generations between each change means a greater number of sample generations in each heuristic selection process. With more sample generations, DPEM-HH-PS can explore the neighborhood of the current population in greater depth and adequately analyze its population evolvability. Having more information while performing a heuristic selection, DPEM-HH-PS can most likely select the LLH to generate the best results.

The effectiveness of DPEM-HH-PS in dMOP2, a type II instance, may suggest that population evolvability is effective when dealing with this problem type. In a type III problem, both Pareto front and optimal solutions change with time. Based on the results obtained in dMOP1, a type III problem, DE algorithms as LLHs are effective since their properties can obtain better results compared to DPEM-HH-PS-B, which only use GA. DHH-PS is particularly effective, thus focusing on performance metrics only and not on population evolvability can perhaps be a better alternative in type III DMOPs.

In dMOP3, the variable that represents f_1 is chosen randomly after each time step. This DMOP presents a new challenge since an algorithm must adapt to the environmental changes and the abrupt alteration of an objective function. DPEM-HH-PS is superior to DHH-PS in most of the environments presented for dMOP3. The use of population evolvability allows LLHs to explore the new environment and analyze the quality of neighboring solutions. This method provides the algorithm with a better adaptation to the changes presented by dMOP3, as it can identify the likeliness of each LLH to obtain better solutions immediately considering the current variable assignment in f_1. The only exceptions to this condition occur when $\tau_\tau = \{10,25\}$ under $W_{f1} = [0.6,0.9]$, confirming the analysis of the previous paragraph.

In previous works, all hyper-heuristics analyzed in this field only considered bi-objective problems. FDA4 and FDA5 are tri-objective DMOPs. DHH-PS can adapt better to this new condition, mainly for FDA4, a type I DMOP. Both DHH-PS and DPEM-HH-PS-A have a similar behavior when solving FDA5, a type II DMOP. The analysis made when reviewing the results on dMOP2 supports the quality of DPEM-HH-PS, reaffirming its ability to solve DMOPs of this type.

The aggregation method used in DPEM-HH-PS is a critical factor when solving FDA4 and FDA5. Although the weighted sum method presents accurate results for bi-objective problems, it finds difficulties when dealing with DMOPs with more objectives.

In conclusion, while DHH-PS has better behavior in most instances, there are multiple cases where DPEM-HH-PS is highly effective, generating better results than DHH-PS. Specifically, DPEM-HH-PS shows effectiveness for bi-objective type I (FDA1 and dMOP3) and type II (dMOP2) problems. Therefore, population evolvability is a potential adequate alternative to solve these preferential optimization problem types, and its use is likely to produce good-quality solutions.

4.4 Analysis of both versions of DPEM-HH-PS

The second part of this analysis focuses on the results obtained by both versions of DPEM-HH-PS. DPEM-HH-PS-A uses both GA and DE algorithms, while DPEM-HH-PS-B uses three versions of DNSGA-II as LLHs. A review of the results presented in Tables 1 and 2 allow to observe that the performance of DPEM-HH-PS-A is better than DPEM-HH-PS-B for FDA1, a type I problem, when $\tau_\tau = 10$. However, as τ_τ increases, the quality of the results of DPEM-HH-PS-B improves, to the point of obtaining statistically significant better results than DPEM-HH-PS-A in FDA1 for $\tau_\tau = 50$ in both dynamic environments.

This same situation is observed for dMOP2, a type II problem, when $W_{f1} = [0.1,0.4]$. It is possible to imply that using a set of genetic and differential evolution algorithms as LLHs is effective when handling quickly-changing environments for this type of instance. On the other hand, as the time between each change increases, the functionality of a set of LLHs formed solely by GAs seems to be a more appropriate option.

The results obtained for dMOP3 under both preferential environments show that the use of DE within the set of LLHs provides DPEM-HH-PS-A with higher adaptability to the randomness of this instance. This outcome suggests that population evolvability provides the hyper-heuristic a better adaptation ability for that situation, as mentioned in Section 4.3. Also, the use of DE algorithms can explore deeper the neighborhood of the current population.

DPEM-HH-PS-A finds difficulties in solving tri-objective DMOPs (FDA4 and FDA5). This issue denotes the problems encountered by its LLHs to solve problems with more than two objectives. DPEM-HH-PS-B emphasizes one of the main properties of hyper-heuristics. In this case, the strengths of DGDE3 are used to cover the weaknesses presented by other LLHs under this environment.

5 Conclusions and future work

This work proposed the use of PS, a preference incorporation method, in various previously proposed hyper-heuristics, DHH-PS, and two versions of DPEM-HH (one using GAs as LLHs second combines GA and DE). The capability of each hyper-heuristic to solve dynamic problems was tested under various bi- and tri-objective DMOPs. Their preferential and dynamic adaptation capabilities were also tested under two different preferential environments and three dynamic environments.

The obtained results allow to suppose that the use of population evolvability in hyper-heuristics can be useful when solving type II DMOPs with preferences, where both sets of optimal solutions and their Pareto front (and therefore the ROI) change over time. This may be due to the ability of this FLA method to explore new areas within the solution search space while exploiting the current area where the population is located.

In the literature, the use of PS in DMOEAs has been shown to be effective for solving DMOPs with preferences. The experiments performed in this paper have shown that PS, as well as the performance metrics set used in this work, applied within hyper-heuristics are also feasible and effective for solving dynamic optimization problems with preferences. Likewise, the effects of using PS in DMOEAs in which this FLA method had not been tested were mostly positive and suggest that further study may provide even better results.

An analysis of the experimental results favors the use of hyper-heuristics that combine GA and DE within their set of LLHs. However, there were conditions in which using hyper-heuristics with a GA-only LLH set may be more effective, especially when handling low dynamic DMOPs (high τ_t). These results lead us to consider that although hyper-heuristics can cover a broader spectrum of problems with different characteristics than other heuristics, it is unlikely that a single hyper-heuristic can effectively solve in all existing problems.

As future work, we recommend developing variants of DHH-PS, using a different set of DMOEAs as LLHs or heuristic selection methods, such as greedy, random, or even other heuristics such as simulated annealing. For DPEM-HH-PS, we recommend using different aggregation methods, such as the Tchebycheff method or another strategy to evaluate population evolvability. Also, likewise DHH-PS, future researchers are suggested to try different heuristics as LLH. Another potential aspect to consider is the use of more strict acceptance criteria.

References

1. K. Deb, U. B. Rao N., and S. Karthik, "Dynamic multi-objective optimization and decision-making using modified NSGA-II: A case study on hydro-thermal power scheduling", *Lecture Notes in Computer Science*, pp. 803–817, 2007. doi.org/10.1007/978-3-540-70928-2_60
2. M. Helbig and A. P. Engelbrecht, "Challenges of dynamic multi-objective optimisation", *2013 BRICS Congress on Computational Intelligence and 11th Brazilian Congress on Computational Intelligence*, pp. 254–261, 2013. doi.org/10.1109/brics-cci-cbic.2013.49
3. E. K. Burke, M. Hyde, G. Kendall, G. Ochoa, E. Özcan, and J. R. Woodward, "A classification of hyper-heuristic approaches", *Handbook of Metaheuristics*, pp. 449–468, 2010. doi.org/10.1007/978-1-4419-1665-5_15
4. G. Rivera, C. Gómez, L. Cruz, R. García, F. Balderas, E. Fernández, & F. López, "Solution to the social portfolio problem by evolutionary algorithms", *International Journal of Combinatorial Optimization Problems and Informatics*, 2012, vol. 3, no. 2, pp. 21-30.
5. M. Maashi, E. Özcan, and G. Kendall, "A multi-objective hyper-heuristic based on choice function", *Expert Systems with Applications*, vol. 41, no. 9, pp. 4475–4493, 2014. doi.org/10.1016/j.eswa.2013.12.050
6. H. Luiz Jakubovski Filho, T. Nascimento Ferreira, and S. Regina Vergilio, "Incorporating user preferences in a software product line testing hyper-heuristic approach", *2018 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1–8, 2018. doi.org/10.1109/cec.2018.8477803
7. S. A. G. van der Stockt and A. P. Engelbrecht, "Analysis of selection hyper-heuristics for population-based meta-heuristics in real-valued dynamic optimization", *Swarm and Evolutionary Computation*, vol. 43, pp. 127–146, 2018. doi.org/10.1016/j.swevo.2018.03.012
8. B. Bilgin, E. Özcan, and E. E. Korkmaz, "An experimental study on hyper-heuristics and exam timetabling", *Practice and Theory of Automated Timetabling VI*, pp. 394–412, 2006. doi.org/10.1007/978-3-540-77345-0_25
9. P. Cowling, G. Kendall, and E. Soubeiga, "A hyperheuristic approach to scheduling a sales summit", *Lecture Notes in Computer Science*, pp. 176–190, 2001. doi.org/10.1007/3-540-44629-x_11
10. S. Bechikh, M. Kessentini, L. B. Said, and K. Ghédira, "Preference incorporation in evolutionary multiobjective optimization", *Advances in Computers*, pp. 141–207, 2015. doi.org/10.1016/bs.adcom.2015.03.001
11. C. Gomez-Santillán, L. Cruz-Reyes, G. Rivera, N. Rangel-Valdez, M. L. Morales-Rodríguez, & M. Pérez-Villafuerte, "Interdependent Projects selection with preference incorporation", *New Perspectives on Applied Industrial Tools and Techniques*. Springer, Cham, 2018. p. 253-271. doi.org/10.1007/978-3-319-56871-3_13

12. G. Rivera, R. Porras, J. P. Sanchez-Solis, R. Florencia, V. García, "Outranking-based multi-objective PSO for scheduling unrelated parallel machines with a freight industry-oriented application", *Engineering Applications of Artificial Intelligence*, vol. 108, 104556, 2022, doi.org/10.1016/j.engappai.2021.104556
13. T. Macias-Escobar, L. Cruz-Reyes, H. Fraire, and B. Dorronsoro, "Plane separation: A method to solve dynamic multi-objective optimization problems with incorporated preferences", *Future Generation Computer Systems*, vol. 110, pp. 864–875, 2020. doi.org/10.1016/j.future.2019.10.039
14. K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II", *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002. doi.org/10.1109/4235.996017
15. R. Roy and J. Mehnen, "Dynamic multi-objective optimisation for machining Gradient Materials", *CIRP Annals*, vol. 57, no. 1, pp. 429–432, 2008. doi.org/10.1016/j.cirp.2008.03.020
16. R. Liu, W. Zhang, L. Jiao, F. Liu, and J. Ma, "A sphere-dominance based preference immune-inspired algorithm for dynamic multi-objective optimization", *Proceedings of the 12th annual conference on Genetic and evolutionary computation - GECCO '10*, pp. 423–430, 2010. doi.org/10.1145/1830483.1830565
17. A. J. Nebro, A. B. Ruiz, C. Barba-González, J. García-Nieto, M. Luque, and J. F. Aldana-Montes, "Indm2: Interactive dynamic multi-objective decision making using evolutionary algorithms", *Swarm and Evolutionary Computation*, vol. 40, pp. 184–195, 2018. doi.org/10.1016/j.swevo.2018.02.004
18. K. Deb and J. Sundar, "Reference point based multi-objective optimization using evolutionary algorithms", *Proceedings of the 8th annual conference on Genetic and evolutionary computation - GECCO '06*, pp. 635–642, 2006. doi.org/10.1145/1143997.1144112
19. A. B. Ruiz, R. Saborido, and M. Luque, "A preference-based evolutionary algorithm for multiobjective optimization: The weighting achievement scalarizing function genetic algorithm", *Journal of Global Optimization*, vol. 62, no. 1, pp. 101–129, 2014. doi.org/10.1007/s10898-014-0214-y
20. S. Kukkonen and J. Lampinen, "GDE3: The third evolution step of Generalized Differential Evolution", *2005 IEEE Congress on Evolutionary Computation*, pp. 443–450, 2005. doi.org/10.1109/cec.2005.1554717
21. E. Ozcan, S. E. Uyar, and E. Burke, "A greedy hyper-heuristic in Dynamic Environments", *Proceedings of the 11th annual conference companion on Genetic and evolutionary computation conference - GECCO '09*, pp. 2201–2204, 2009. doi.org/10.1145/1570256.1570302
22. L. Davis, "Bit-climbing, representational bias, and test suit design", *Proceedings of the Fourth International Conference on Genetic Algorithms*, pp. 18–23, 1991.
23. H. Wang, D. Wang, and S. Yang, "A memetic algorithm with adaptive hill climbing strategy for dynamic optimization problems", *Soft Computing*, vol. 13, no. 8-9, pp. 763–780, 2008. doi.org/10.1007/s00500-008-0347-3
24. A. Baykasoğlu and F. B. Ozsoydan, "Evolutionary and population-based methods versus constructive search strategies in dynamic combinatorial optimization", *Information Sciences*, vol. 420, pp. 159–183, 2017. doi.org/10.1016/j.ins.2017.08.058
25. 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, 2017. doi.org/10.1007/s40747-017-0036-x
26. T. Macias-Escobar, L. Cruz-Reyes, B. Dorronsoro, H. Fraire-Huacuja, N. Rangel-Valdez, and C. Gómez-Santillán, "Application of population evolvability in a hyper-heuristic for dynamic multi-objective optimization", *Technological and Economic Development of Economy*, vol. 25, no. 5, pp. 951–978, 2019. doi.org/10.3846/tede.2019.10291
27. M. Wang, B. Li, G. Zhang, and X. Yao, "Population evolvability: Dynamic Fitness Landscape Analysis for population-based metaheuristic algorithms", *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 4, pp. 550–563, 2018. doi.org/10.1109/tevc.2017.2744324
28. L. Altenberg, "The evolution of evolvability in Genetic Programming", *Advances in Genetic Programming*, pp. 47–74, 2003. doi.org/10.7551/mitpress/1108.003.0008
29. C.-K. Goh and K. C. Tan, "A competitive-cooperative coevolutionary paradigm for dynamic multiobjective optimization", *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 1, pp. 103–127, 2009. doi.org/10.1109/tevc.2008.920671
30. K. C. Tan, T. H. Lee, and E. F. Khor, "Evolutionary algorithms for multi-objective optimization: Performance assessments and comparisons", *Proceedings of the 2001 Congress on Evolutionary Computation*, vol. 17, no. 4, pp. 251–290, 2002. doi.org/10.1109/cec.2001.934296
31. D. A. Van Veldhuizen, "Multiobjective evolutionary algorithms: Classifications, analyses, and New Innovations", dissertation, Wright Patterson AFB, Greene, Ohio, 1999.
32. M. R. Sierra and C. A. Coello Coello, "Improving PSO-based multi-objective optimization using crowding, mutation and ∈-dominance", *Lecture Notes in Computer Science*, pp. 505–519, 2005. doi.org/10.1007/978-3-540-31880-4_35
33. M. Farina, K. Deb, and P. Amato, "Dynamic multiobjective optimization problems: Test cases, approximations, and applications", *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 5, pp. 425–442, 2004. doi.org/10.1109/tevc.2004.831456
34. K. Deb and R. B. Agrawal, "Simulated binary crossover for continuous search space", *Complex systems*, vol. 9, no. 2, pp. 115–148, 1995.
35. J. L. Hodges and E. L. Lehmann, "Rank methods for combination of independent experiments in analysis of variance", *Selected Works of E. L. Lehmann*, pp. 403–418, 2011. doi.org/10.1007/978-1-4614-1412-4_35
36. S. Holm, "A simple sequentially rejective multiple test procedure", *Scandinavian Journal of Statistics*, vol. 6, no. 2, pp. 65–70, 1979.