Contents lists available at ScienceDirect

Swarm and Evolutionary Computation

journal homepage: www.elsevier.com/locate/swevo

A perturbation adaptive pursuit strategy based hyper-heuristic for multi-objective optimization problems

Shuyan Zhang[a], Zhilei Ren[b,*], Cuixia Li[a], Jifeng Xuan[c]

[a] School of Software, Zhengzhou University, Zhengzhou, 450002, China
[b] School of Software, Dalian University of Technology, Dalian, 116024, China
[c] School of Computer Science, Wuhan University, Wuhan, 430072, China

A R T I C L E I N F O

Keywords:
Multi-objective optimization
Selection hyper-heuristic
Learning mechanism
Perturbation adaptive pursuit strategy

A B S T R A C T

For multi-objective optimization problems, obtaining a uniformly distributed approximation set is among the most important issues. During the past decades, various diversity mechanisms have been proposed to address this challenge. However, the existing diversity mechanisms tend to be problem-specific, and may not generalize well over different problem domains. Inspired by the idea of utilizing multiple low-level heuristics to achieve better diversity performance in multi-discipline problem solving, we focus on efficient algorithm design based on the methodology of selection hyper-heuristics. This study proposes a novel selection hyper-heuristic operating over multiple diversity mechanisms. The unique feature of the proposed approach lies in its ability to intelligently learn, select, and combine different diversity mechanisms with the purpose of taking advantages of them to obtain well-distributed approximation sets. Moreover, this work develops a new learning mechanism, the perturbation adaptive pursuit strategy, which is incorporated into the proposed hyper-heuristic to improve the decision-making process of selecting suitable diversity mechanisms for the problem at hand. The performance of the proposed hyper-heuristic is tested on 2-objective ZDT, 3-objective DTLZ, and 5-objective WFG test suites. Additionally, experiments are also conducted to investigate the ability of the novel hyper-heuristic to integrate existing multi-objective meta-heuristics on MaOP test suite from 3- to 10-objectives. Experimental results demonstrate the effectiveness of the proposed selection hyper-heuristic for cross-domain capacity, particularly in producing well distributed approximation set with respect to Spacing metric and Hypervolume metric.

1. Introduction

Without loss of generality, a minimization multi-objective optimization problem (MOP) can be mathematically formulated as

$$\min \; F(x) = (f_1(x), f_2(x), \ldots, f_m(x))^T \tag{1}$$

s.t. $x = (x_1, x_2 \ldots, x_n) \in \Omega$.

Where $x \in \Omega$ is a n-dimensional decision vector and Ω is the n-dimensional decision space. $f_1(x)$, $f_2(x)$, …, $f_m(x)$ are m conflicted objective functions, and $\Omega \to R^m$ is the objective space. For x^1, $x^2 \in \Omega$, x^1 dominates x^2, iff $f_i(x^1) \le f_i(x^2)$ for every $i \in \{1,2,\ldots m\}$ and $f_j(x^1) < f_j(x^2)$ for at least one dimension $j \in \{1,2,\ldots m\}$. $x^* \in \Omega$ is a Pareto-optimal solution to (1), iff there is no $x \in \Omega$ such that x dominates x^*. The set of all the Pareto-optimal solutions is called the Pareto-optimal set while the image of the Pareto-optimal set in the objective space is called the Pareto-optimal front.

Over the past twenty years, a number of meta-heuristic algorithms have been proposed for MOPs, such as multi-objective evolutionary algorithms (MOEAs) [1] and multi-objective particle swarm optimizers (MOPSOs) [2]. MOEAs and MOPSOs are popular because of their population-based nature. Thus, they can get the Pareto-optimal set in a single run by evolving a set of solutions [1–3]. More than 4900 relative papers have been published by 2017 according to EMOO web site (http://delta.cs.cinvestav.mx/~ccoello/EMOO/EMOOjournals.html) maintained by Professor Coello.

The quality of an algorithm for MOPs is usually considered from two points of view: (i) the distribution and (ii) the convergence [4,5]. An ideal algorithm should get an approximation set well converged and uniformly distributed along the Pareto-optimal front. In real-life applications, obtaining the complete Pareto-optimal set is very time-consuming and unnecessary for decision makers. So, how to find a

* Corresponding author. No.321 Tuqiang Street, Economy & Technology Development Zone, Dalian, 116620, China.
E-mail addresses: syzhang@zzu.edu.cn (S. Zhang), zren@dlut.edu.cn (Z. Ren), lcxxcl@zzu.edu.cn (C. Li), jxuan@whu.edu.cn (J. Xuan).

https://doi.org/10.1016/j.swevo.2020.100647
Received 14 December 2018; Received in revised form 6 November 2019; Accepted 22 January 2020
Available online 29 January 2020
2210-6502/© 2020 Elsevier B.V. All rights reserved.

manageable number of Pareto-optimal solutions which are uniformly distributed in the objective space is one of the major issues in current multi-objective optimization research. Toward this issue, several current researches concentrate on the distribution aspect and try to find reasonable compromise in the convergence aspect [6,7]. It has been found that a well-distributed solution set could reduce the probability of getting trapped in local optima in the decision space of MOPs during the search process, and help to prevent the premature convergence in some extent [8,9]. Besides, in many-objective optimization (the number of objectives is more than 2 or 3), the distribution criterion is assumed to be more important than the convergence criterion [10]. A key issue is the fact that the number of non-dominated solutions becomes very large as the number of objectives increases, and the convergence pressure and selection approaches may be insufficient to make progress for algorithms using dominance to drive the search [7,11].

To address the distribution problem, several diversity mechanisms have been proposed. The most classical ones include the hyper-grid strategy from PESA-II [12], the density estimation strategy from SPEA2 [13], the crowded-comparison strategy from NSGA-II [14], the weight vectors used in MOEA/D [15] and quality indicator used in IBEA [16]. Despite the success of existing methods, there are still some drawbacks. In the literature, it has already been observed that diversity performance of different meta-heuristics is varied on different MOPs [17–19]. The current methods tend to be problem-specific, and may not generalize well to new problem domains, or even new instances of the same problem. That is, single method cannot guarantee the effectiveness on all the problem instances. As a consequence, when faced with new problem domains, specific algorithms have to be designed, which may lead to an expensiveness to develop, tune, and maintain new methods [20].

Based on the above observations, it would be an advantage to combine different diversity mechanisms to derive strengths of them and avoid their weakness for MOPs. In the meantime, there is a growing number of studies on the hyper-heuristics [20,21]. The main motivation of hyper-heuristic is to combine the merits of different heuristics into a single heuristic, and develop a more generally applicable heuristic to solve a large range of search problems [20]. In particular, in this study, we pay attention to the selection hyper-heuristics, due to its simplicity and effectiveness. Unlike the traditional problem-solving approaches that try to solve the problem directly, the selection hyper-heuristics attempt to manage and select from a set of heuristics or heuristic components (called low-level heuristics) to solve the problem at hand [21]. By the careful selection of suitable low-level heuristics at different search processes, the cooperation of low-level heuristics via selection hyper-heuristics could obtain promising results compared to each low-level heuristic being used alone [22,23]. However, there are only limited studies on selection hyper-heuristics for MOPs using learning strategies to improve the adaptive selection ability for choosing suitable low-level heuristics [20, 24–26]. Additionally, none of these studies concern about the diversity aspect of the approximation set obtained by a multi-objective algorithm.

In this study, our contributions can be summarized as follows. (i) We propose a novel multi-objective selection hyper-heuristic framework with learning, namely the perturbation adaptive pursuit strategy based hyper-heuristic (PAPHH). The proposed hyper-heuristic focuses on improving the diversity aspect of MOPs. (ii) We propose a new effective learning strategy, namely the perturbation adaptive pursuit strategy (PAP), which is incorporated into PAPHH as a learning strategy. The PAP strategy contains two learning components, the adaptive pursuit strategy and the perturbation strategy. The central idea of the PAP strategy is to select suitable low-level heuristics and try to avoid trapping into local optima in the low-level heuristic search space. (iii) PAPHH with several diversity mechanisms as low-level heuristics is tested on 2-objective ZDT, 3-objective DTLZ, and 5-objective WFG test suites against a variety of meta-heuristics and hyper-heuristics, including a variant of PAPHH. (iv) Moreover, to test the integration ability, the performance of PAPHH incorporating several multi-objective meta-heuristics (NSGA-III, IBEA, MOEA/D and SMPSO) as low-level heuristics is also experimentally analyzed on a novel test suite MaOP from 3- to 10-objectives in detail. Empirical results demonstrate the adaptive selection capability of the PAP strategy as well as the effectiveness and generality of PAPHH.

The remainder of this paper is structured as follows. Section 2 introduces the background of meta-heuristics for MOPs, learning based multi-objective selection hyper-heuristics and the adaptive pursuit strategy. Section 3 describes the details of the proposed PAPHH using diversity mechanisms as low-level heuristics. Experimental results of PAPHH comparing with several low-level meta-heuristics and hyper-heuristics on ZDT, DTLZ, and WFG test suites are conducted in Section 4. Comparison results with existing MOEAs and MOPSOs are presented in Section 5. Section 6 modifies PAPHH by incorporating multiple MOEAs and MOPSOs as low-level heuristics. The performance of PAPHH is tested on MaOP test suite with different objectives. Finally, conclusions are drawn in Section 7.

2. Backgrounds

2.1. Meta-heuristics for MOPs

This subsection describes the well-known meta-heuristics, including MOEAs and MOPSOs, for MOPs and their performance comparisons.

MOEAs can be roughly classified into three categories generally [2, 27,28]. (i) Pareto-based MOEAs: In this framework, all the objectives are optimized simultaneously. Two main strategies are adopted here. The first one is the fitness assignment strategy based on the Pareto dominance mechanism to guide the convergence of solution sets. The second one is the diversity mechanism used to maintain the solution set diversity. Representative MOEAs of this type include PESA-II, SPEA2, and NSGA-II. Recently, some researches propose more effective Pareto-based MOEAs, such as NSGA-III [29] for many-objective optimization problems. (ii) Decomposition-based MOEAs: In this framework, MOPs are converted into a set of single-objective optimization sub-problems by decomposition approaches. Then the decomposition-based MOEAs are actually solving the single-objective sub-problems. In this way, solution set diversity is maintained by the weight vectors in decomposition approaches. Representative decomposition-based MOEAs include MOEA/D [15] and MOGLS [30]. Recently, some other MOEAs of this type focus on many-objective optimization, such as MOEA/DD [31]. (iii) Indicator-based MOEAs: In this framework, the selection criteria in the environmental selection are measured by a single indicator for the solution qualities, such as hypervolume [32] and R2 indicator [33]. These indicators integrate with both the convergence and diversity performance measurement. Some of the remarkable indicator-based MOEAs are IBEA [16], MaOEA/IGD [34], and AR-MOEA [27].

MOPSOs are also population-based meta-heuristics inspired by social behavior, such as bird flocking. In the past years, a lot of studies have been focusing on MOPSOs, trying to address two main issues. The first one is how to deal with personal and global optimal. The second one is how to balance convergence and diversity of swarms [35]. These MOPSOs can be roughly classified into two categories according to the approaches to identify swarm leaders, Pareto-based MOPSOs and Decomposition-based MOPSOs [35]. Representative MOPSOs includes NMPSO [2], SMPSO [36], WOF-SMPSO [37], and CMOPSO [38].

Despite their success and promising results, the meta-heuristics above are problem-specific, i.e., their performance may vary on different problems. In the literature, some studies have compared the performance of meta-heuristics for MOPs against each other. The diversity performance of NSGA-II and PESA-II is difficult to be effective in cases with more than 2 and 3 objectives, respectively. However, SPEA2 is able to perform well for 3 or more objectives [4]. NSGA-II outperforms SPEA2 on 2-objective WFG problems with respect to Epsilon metric and Hypervolume metric, while SPEA2 performs better in 3 objectives [18]. SPEA2 is superior to NSGA-II and PESA-II in terms of Spacing metric for most of ZDT problems and DTLZ problems, while NSGA-II is better than PESA-II in most cases [19]. SPEA2 and NSGA-II performed similarly on diversity maintenance for DTLZ problems according to two different diversity metrics [17]. After comparing NSGA-III with MOEA/D in Ref. [29], the two algorithms do not dominate each other. However, NSGA-III can work better on MOPs with nonuniformly distributed Pareto-optimal front than MOEA/D [29].

Moreover, the performance of most meta-heuristics may strongly depend on the shape of Pareto-optimal front of the problem at hand [27]. For example, pareto-based algorithms tend to have inferior performance for problems with difficult Pareto-optimal set and high-dimensional objectives [29]. For decomposition-based algorithms, good performance can be achieved only if the weight vectors coordinate with the shape of Pareto-optimal front [39]. For example, MOEA/D is not able to get a well distributed solution set on problems with nonuniformly distributed Pareto-optimal front (such as DTLZ4) [29]. It is also observed in Ref. [15] that the performance of MOEA/D with different weight vectors are varied which are also verified in Ref. [29] on 3- to 15-objective MOPs. Some indicator-based algorithms have difficulties in dealing with problems with different types of Pareto-optimal front [27]. The indicator-based algorithms with the hypervolume indicator prefer the border front and some Pareto-optimal solutions may be eliminated, and thus getting a non-well distributed solution set [40].

2.2. *Related work on learning-based multi-objective selection hyper-heuristics*

To alleviate the problem-specific challenges of meta-heuristics, hyper-heuristic is a possible way to mix the advantages of existing meta-heuristics and avoid their drawbacks. The hyper-heuristic is described as "a search method or learning mechanism for selecting or generating heuristics to solve computational search problems" [20,21]. Generally, there are two main hyper-heuristic categories: (i) selection hyper-heuristic and (ii) generation hyper-heuristic [20]. The former one is heuristics to choose or select existing heuristics, while the latter one is heuristics to generate new heuristics with existing components. The existing heuristics or components are called low-level heuristics. In this paper we focus on the selection hyper-heuristic to solve MOPs, especially for the diversity aspect, due to its simplicity meanwhile promising generality.

A general selection hyper-heuristic works as follows. After generating an initial solution/solution set, a series of iterations begin to improve the solution/solution set by applying selected low-level heuristics. During each iteration, the selection hyper-heuristic decides which low-level heuristic will be selected and applied to generate new solution/solution set and whether the new generated solution/solution set is accepted into the next iteration or not.

One of the primary challenges for the selection hyper-heuristics is how to improve the adaptive selection capability and to decide the most suitable low-level heuristics at a certain solving process. Learning is a key solution [24]. Online-learning represents a subset that selection decisions are taken based on the feedback from previous search processes. Some recent studies of the selection hyper-heuristics incorporate online-learning strategies, relying on the learning strategies' predictive power to improve their overall selection effectiveness. Examples of the online-learning strategies in selection hyper-heuristic include machine learning [41] and reinforcement learning [42].

Although a majority of online-learning based selection hyper-heuristics have been limited to single-objective optimization problems, still, there are some studies for MOPs. Burke et al. proposed a selection hyper-heuristic using reinforcement learning with a fixed size tabu list as the learning strategy [43]. Each low-level heuristic with respect to each objective is maintained with scores which are got from the learning to decide which low-level heuristic will be chosen and applied next in the following stages. Vrugt and Robinson presented an adaptive multi-objective algorithm (AMALGAM) to manage a set of population-based multi-objective algorithms. This method uses an adaptive strategy to learn the performance of each algorithm and then favors the highest reproductive ones [44]. McClymont and Keedwell described another selection hyper-heuristic approach (MCHH) using the Markov chain and reinforcement learning scheme to adaptively learn and select four low-level heuristics (three perturbative heuristics and one ineffective heuristic) [45]. The Markov chain provides the probability of transferring between low-level heuristics, while the reinforcement learning is to adapt the selection process by updating the transition weight matrix according to the performance of each low-level heuristic. Afterwards, Kumari, and Srinivas incorporated the reinforcement learning with adaptive weights into a multi-objective selection hyper-heuristic evolutionary algorithm (MHypEA), which learns and manages twelve low-level heuristics (including different EA, selection, crossover, and mutation operations) for scheduling and inspection planning problems in software development projects [46].

More recently, Maashi et al. and Li et al. proposed new learning-based selection hyper-heuristics which are among the best performing hyper-heuristics for MOPs [24–26]. In Ref. [25], Maashi et al. described a multi-objective selection hyper-heuristic (HH-CF) with the choice function as the learning mechanism to learn and manage three low-level heuristics (NSGA-II, SPEA2, and MOGA) adaptively. Four performance evaluation metrics are incorporated into the choice function to help rank the performance of each low-level heuristic. The low-level heuristic with the top ranking is selected and applied in the next stage. Experiments show that HH-CF performs better than three low-level heuristics (NSGA-II, SPEA2, and MOGA) and two hyper-heuristics (a random hyper-heuristic and AMALGAM) on WFG test suite and multi-objective vehicle crashworthiness design problems. Followed [25], Maashi et al. extended HH-CF in Ref. [26] to invest the effectiveness in selection hyper-heuristics with different combinations of learning strategies and acceptance strategies. Later, Li et al. proposed two selection hyper-heuristics, namely HH-LA and HH-RILA, based on the learning automata [24]. The two new hyper-heuristics contain four key components: initialization, low-level heuristic selection, reinforcement learning, and switching. HH-LA and HH-RILA are exactly the same except the initialization process. The proposed hyper-heuristics maintain a transition matrix during the search process, describing the transforming probability between low-level heuristics. The selected low-level heuristic is applied until the condition of switching is satisfied. Then, the transition matrix is updated by the reinforcement learning mechanism with the changes in hypervolume values as the feedback in searching processes. The results in Ref. [24] show that HH-RILA outperforms the three low-level heuristics (NSGA-II, SPEA2, IBEA) and three hyper-heuristics (a random hyper-heuristic, HH-CF, and HH-LA) on WFG and DTLZ test suites, as well as vehicle crashworthiness design problems in the overall.

Besides, a different learning mechanism, the adaptive operator selection, has been proposed to be incorporated into selection hyper-heuristics. The adaptive operator selection, within the evolutionary computation community, can be considered closely relevant to the learning selection approach of hyper-heuristics [47,48]. Several studies have shown the effectiveness by incorporating hyper-heuristics with the adaptive operator selection [44,49]. The adaptive operator selection (such as the adaptive pursuit [50] and multi-arm bandits [51]) has the ability to deal with the exploration versus exploitation dilemma, preferring to select outstanding low-level heuristics while giving chances to explore poor ones. Whereas, there are only a few studies on selection hyper-heuristics based on the adaptive operator selection mechanism for MOPs. For example, Guizzo et al. introduced a new selection hyper-heuristic (HITO) with the choice function and the multi-armed bandit function as learning strategies. HITO uses the dominance concepts and the elapse number of low-level heuristics as the quality measurement to adaptively select low-level heuristics for multi-objective Integration and Test Order Problem [52]. Additionally, Hitomi and Selva presented a selection hyper-heuristic based on adaptive pursuit (HH-AP) to solve a multi-objective design problem for an earth observation satellite system [53]. HH-AP is based on the framework of a steady-state multi-objective evolutionary algorithm ε-MOEA. The major difference from ε-MOEA is the offspring generation process. At each iteration, HH-AP adaptively learns and selects one of the multiple domain-specific heuristics, which are analogous to mutation operators, to create new offspring.

In this study, we present a novel learning-based selection hyper-heuristic for MOPs. The design of the proposed hyper-heuristic PAPHH is motivated in three ways. Firstly, most of the above multi-objective selection hyper-heuristics pay attention to the improvement of EAs by incorporating multiple mating operators in the evolutionary process [45,46,52,53]. However, the diversity aspect, as one of the most important issues of MOPs, has not been thoroughly investigated in any of the above hyper-heuristics by making use of components specifically designed for getting well distributed approximation sets. Secondly, in the multi-objective optimization field, it is generally recognized that no single heuristic/heuristic component excels in all performance measures [54]. Previous studies show that diversity performance of multi-objective meta-heuristics differs from each other. Hence, the combination of different diversity mechanisms or meta-heuristics via a hyper-heuristic is a feasible way to benefit from each of them, meanwhile avoid their weakness, and finally possess an algorithm with better generality and effectiveness. Furthermore, learning to explore and exploit in the low-level heuristic search space is important for a selection hyper-heuristic. Then, designing a new learning strategy is reasonable. A major difference of PAPHH and the above multi-objective selection hyper-heuristics is that PAPHH tries to improve the generality and effectiveness across multiple MOPs especially on the diversity aspect. Additionally, a new effective learning strategy, the PAP strategy, is proposed to be embedded into PAPHH as the learning mechanism during the search.

2.3. The adaptive pursuit strategy

The most relevant learning mechanism of the PAP strategy is the adaptive pursuit strategy [50], which is one of the adaptive operator selection mechanisms. The adaptive pursuit strategy is straightforward and has shown promising results on several test problems based on its exploration and exploitation ability [53]. The adaptive pursuit strategy has two main tasks: the credit assignment and the operator selection. The former one defines how to credit or reward an operator based on its historical performance in search processes, while the latter one is an operator selection mechanism that can automatically select and apply an operator in the next step based on current reward values of operators [50].

Without loss of generality, given a set of l operators, denoted as $H = \{h_1, \ldots, h_l\}$. Let the selection probability vector be $W_t = (w_{1,t}, \ldots, w_{l,t})$ (i.e. $0 \le w_{i,t} \le 1$ for all i=1, …, l) , and the quality vector be $Q_t = (q_{1,t}, \ldots, q_{l,t})$ at decision point t. The selection probability value and the quality value of operator h_i are $w_{i,t}$ and $q_{i,t}$, respectively, at decision point t. Then the adaptive pursuit strategy:

1) selects an operator h_a ($a \in \{1, \ldots, l\}$) according to the selection probability vector W_t and then estimates the reward value $r_{a,t}$ after applying h_a.
2) updates the quality value of h_a by a relaxation scheme with formula (2) while maintaining quality values of other operators with formula (3).

$$q_{a,t+1} = (1-\alpha) \times q_{a,t} + \alpha \times r_{a,t} \quad (0 \le \alpha \le 1) \tag{2}$$

$$(\forall i \ne a): q_{i,t+1} = q_{i,t} \tag{3}$$

3) increases the selection probability value of operator h_{a^*} ($h_{a^*} = argmax\{Q_{t+1}\}$) with formula (4) and decreases the selection probability values of other operators with formula (5).

$$w_{a^*,t+1} = (1-\beta) \times w_{a^*,t} + \beta \times w_{max} \quad (0 \le \beta \le 1) \tag{4}$$

$$(\forall i \ne a^*): \; w_{i,t+1} = (1-\beta) \times w_{i,t} + \beta \times w_{min} \tag{5}$$

4) if the algorithm is not terminated, the adaptive pursuit strategy goes to step 1) to select an operator based on W_{t+1} at the next decision point.

α and β are two control parameters, s.t.$0 \le \alpha \le 1$ and $0 \le \beta \le 1$. To avoid some of operators no longer being chosen for the reason that their selection probability values converge to 0, two parameters $w_{\min}$ and $w_{\max}$ are introduced to enforce the selection probability value within the interval $[w_{\min}, \ldots, w_{\max}]$, s.t.$0 < w_{\min} < w_{\max} < 1$ and $w_{\max} = 1-(l-1) \times w_{\min}$.

3. Hyper-heuristic frameworks for multi-objective optimizationAlgorithm 1

Procedure for PAPHH.

```
 P_0: a uniformly random initial solution set
H: a set of diversity low-level heuristics {h_1, h_2, ..., h_l}
1 [W_0, Q_0] = Initialise(H);
2 While NotTerminated do
3    P_{t+1} = EvolutionPopution(P_t);
   //the diversity management procedure
4    If |P_{t+1}| > N then
5       h_a = LowLevelHeuristicSelection (W_t);
6       P_{t+1} = ApplyLowLevelHeuristic (h_a, P_{t+1});
7       r_{a,t} = GetReward(h_a);
8       [W_{t+1}, Q_{t+1}] = PerturbationAdaptivePursuitStrategy (W_t, Q_t, h_a, r_{a,t})
9    End
10 End
```

This paper proposes a new selection hyper-heuristic, PAPHH, using the PAP strategy to determine which diversity low-level heuristic to be applied to manage the diversity of the population for a given problem. The framework of PAPHH is illustrated in Algorithm 1. For a test problem *pro*, given a uniformly random initial solution set P_0 of size N, and a set of diversity low-level heuristics $H = \{h_1, h_2, \ldots, h_l\}$, the selection probability vector W_0 and the quality vector Q_0 are initialized in the beginning. The selection probability value $w_{i,0}$ of the low-level heuristic h_i is initially set to $1/l$ and the quality value $q_{i,0}$ of h_i is set to 1 at the initialization phase (Step 1). Then an iteration begins to iteratively update the solution set step by step (Steps 2–10). We describe the tth iteration of the proposed algorithm below. At first, the population P_t is evolved and a new population P_{t+1} is produced to the next iteration (Step 3). If $|P_{t+1}| > N$, then the diversity management procedure is invoked to remove extra solutions from P_{t+1} until its size to N (Steps 4–9). In the diversity management procedure of PAPHH, firstly, a low-level heuristic h_a from $H = \{h_1, \ldots, h_l\}$ is selected using a roulette wheel method according to the selection probability vector W_t (Step 5). Then, h_a is applied to delete $|P_{t+1}|$ - N extra solutions from P_{t+1} (Step 6), and the reward value $r_{a,t}$ with respect to the diversity aspect is obtained after applying h_a (Step 7). Next, according to the reward value $r_{a,t}$, the PAP strategy updates the selection probability vector W_{t+1} and the quality vector Q_{t+1} (Step 8). Afterwards, in the next iteration, the selection process is conducted with the updated selection probability vector W_{t+1} to decide which low-level heuristics should be selected. One low-level heuristic is selected at each solving iteration, but every low-level heuristic shares the same population. Steps 3–9 are repeated until the termination criterion is met. Here, we choose the max iteration number as the termination criterion.

There are three key components: evolutionary process, the PAP strategy and reward assigning. The following subsections describe the three components in detail.

3.1. Evolutionary process

In the literature, many MOEAs share more or less the same framework as NSGA-II. So, PAPHH uses the similar framework to NSGA-II in the population evolutionary process to form a new population P_{t+1} for the next iteration. Firstly, the current population P_t is evolved and an offspring population E_t of size N is produced. Then, a union population U_t of size 2 N is formed ($U_t = E_t \cup P_t$). Next, the population U_t is sorted into different non-domination levels with the domination concept. Population subset of U_t with the lowest non-domination level is firstly selected into P_{t+1}. If $|P_{t+1}| < N$, the subset with the second lowest non-domination level is selected next and so forth. Evolution operators used above are the same with that in NSGA-II. The difference from NSGA-II is that the process of forming P_{t+1} is stopped until the size of P_{t+1} is larger than or equals to N. Then, the extra $|P_{t+1}|$ - N solutions will be deleted in the diversity management procedure.

3.2. The perturbation adaptive pursuit strategy

The PAP strategy is the core in the diversity management procedure, considering two components, both the adaptive pursuit strategy and the perturbation strategy. The adaptive pursuit strategy is used to perceive the state of each low-level heuristic and help the hyper-heuristic to search in the low-level heuristic search space by utilizing a selection probability vector. The perturbation strategy is used to help the search to escape from local optima. The design of the perturbation strategy in this paper is to reset the selection probability vector with the number of improvement times of each low-level heuristic. The framework of the PAP strategy is presented in Algorithm 2. In the PAP strategy:

Algorithm 2
Procedure for the PAP strategy.

```
 W_t: the selection probability vector of the tth iteration
 Q_t: the quality vector of the tth iteration
 h_a: the selected low-level heuristic
 r_{a,t}: the reward value after applying h_a
   //update the quality vector Q_{t+1}
 1 $q_{a,t+1} = (1-\alpha) \times q_{a,t} + \alpha \times r_{a,t}$
 2 $q_{i,t+1} = q_{i,t}$ for $i \neq a$
 3 If $r_{a,t} > 0$ then
     //the number of invocations of $h_a$ that receiving positive reward values is increased by 1
 4     $g_a = g_a + 1$;
 5 End
 6 If improvement can be achieved for at least one time within $K$ iterations then
     //the selection probability vector $W_{t+1}$ is updated using the adaptive pursuit strategy
 7     $h_{a^*} = argmax\{Q_{t+1}\}$
 8     $w_{a^*,t+1} = (1-\beta) \times w_{a^*,t} + \beta \times w_{\max}$
 9     $w_{i,t+1} = (1-\beta) \times w_{i,t} + \beta \times w_{\min}$ for $i \neq a^*$
 10 Else//the selection probability vector $W_{t+1}$ is reset using the perturbation strategy
 11     $w_{i,t+1} = \frac{g_i + 1}{\sum_{j=1}^{l}(g_j + 1)}$ for $i = 1,\ldots,l$
 12 End
```

1) After applying the low-level heuristic h_a and getting the reward value $r_{a,t}$, the quality value $q_{a,t+1}$ of h_a is updated by using formula (2) (Step 1) while quality values $q_{i,t+1}$ ($i \neq a$) of other low-level heuristics are updated with formula (3) (Step 2).
2) If h_a receiving a positive reward value (means $r_{a,t} > 0$), then the value of g_a is increased by 1. Here, g_a is the number of invocations of operator h_a that receiving positive reward values accumulated from the beginning of PAPHH to the current decision point (Steps 3–5). The value of g_a is initialized to 0 in the beginning of PAPHH.
3) If improvements in diversity aspect can be achieved for at least one time within the last K consecutive iterations, the selection probability vector is updated using the adaptive pursuit strategy (Steps 6–9). That is, the selection probability value $w_{a^*,t+1}$ of operator h_{a^*} ($h_{a^*} = argmax\{Q_{t+1}\}$) is increased with formula (4) (Step 8) and the selection probability values $w_{i,t+1}$ ($i \neq a^*$) of other operators are decreased with formula (5) (Step 9).
4) Otherwise, if there is no diversity improvement during the last K consecutive iterations, the perturbation strategy is applied to generate a new selection probability vector for the next iteration (Steps 10–12). We can assume that the searching of low-level heuristics is trapped into a local optima, and the adaptation does not work significantly at this solving stage. In order to improve the ability to select suitable operators, the selection probability vector is reset in a perturbative way utilizing the times of improvement of each low-level heuristic with the purpose to get out of local optima. The selection probability value $w_{i,t+1}$ of h_i is reset with formula (6) for every $i \in \{1, \ldots, l\}$ (Step 11).

$$w_{i,t+1} = \frac{g_i + 1}{\sum_{j=1}^{l}(g_j + 1)} \tag{6}$$

Formula (6) guarantees that $w_{i,t+1} > 0$ for all i=1, …, l and $\sum_{i=1}^{l} w_{i,t+1} = 1$. The parameter K is a predefined threshold for detecting whether the search falls into local optimum or not. The improvement is defined as receiving a positive reward value.

3.3. Reward assigning

In order to provide useful information as feedback in the learning phase, the reward of a low-level heuristic requires to be assessed after applying it. The reward values are easy to be determined with fitness functions. For our investigation aims to get a uniformly distributed approximation set in the objective space, Spacing metric [55] which evaluates the distribution of a solution set (details in Section 4.1) is employed as the fitness function. A smaller value of Spacing metric indicates better performance. This metric is chosen as it has been commonly used for MOEAs to evaluate the distribution of approximation sets over the Pareto-optimal front [19,56]. We do not choose other diversity metrics for their similarity to Spacing metric or their assessed value is affected by the degree of convergence of the approximation set. In addition, we do not combine Spacing metric with other diversity metrics into a single fitness function, because these metrics provide values that are in different scalar units. Simple combination will lead the search to a wrong direction. Moreover, as discussed in Ref. [57], larger fitness improvement values are easier to be got at earlier stages than latter ones. The low-level heuristic luckily to be chosen in the beginning is apt to have large fitness improvement values even if it has trivial improvement later. Directly using raw fitness improvement values will affect the robustness of the algorithm [57]. So, we do not use the raw fitness improvement as the reward value in our work. The reward value $r_{a,t}$ after applying h_a is calculated by the fitness improvement rate function with the best fitness value as the baseline, shown as formula (7).

$$r_{a,t} = \begin{cases} 1 + \frac{f_{best} - f_{a,t}}{f_{best}}, & \text{if } f_{a,t} \le f_{best} \\ 0 \quad , & \text{otherwise} \end{cases} \tag{7}$$

where $f_{a,t}$ is the current fitness value after applying h_a at the tth iteration. f_{best} is the best fitness value found so far. In the beginning, f_{best} is initially set as the Spacing metric value of P_0.

3.4. Computational complexity of PAPHH

In this subsection, we discuss the computational complexity of PAPHH. Since the proposed algorithm works in an iterative paradigm, we focus on the complexity of each iteration of PAPHH. More specifically, the non-dominated sorting in evolving a solution set of size N with m objectives requires $O(mN^2)$ computations [14]. Other operations in evolutionary process have smaller complexities (Step 3 in Algorithm 1).

Selection of a low-level heuristic h_a from a set of l low-level heuristics requires $O(l)$ computations (Step 5). The computations of applying the selected low-level heuristic h_a in Step 6 depend on the computations of h_a, denoted as $O(X_a)$. Getting reward needs O(mN) computations when the fitness function is calculated by the Spacing metric (Step 7). The PAP strategy requires $O(l)$ computations to reset the quality values and selection probability values of l low-level heuristics (Step 8). Usually, $N > l$. Among the above computations, $O(X_a)$ is not sure. After considering all the computations above, the computational complexity of one iteration of PAPHH is max{$O(mN^2)$, $O(X_a)$}. Assuming the low-level heuristic with the lowest and the highest computational complexity are h_i and h_j, respectively, the average computational complexity of PAPHH in one iteration is between max{$O(mN^2)$, $O(X_i)$} and max{$O(mN^2)$, $O(X_j)$}.

4. Experimental results on ZDT, DTLZ and WFG test suites

4.1. Experimental settings and performance evaluation criteria

4.1.1. Test problems and comparative algorithms

Experiments are tested on three test suites, including 2-objective ZDT test suite (ZDT1-ZDT4 and ZDT6) [58], 3-objective DTLZ test suite (DTLZ1-DTLZ7) [59] and 5-objective WFG test suite (WFG1-WFG9) [60]. The three test suites are common choice for most researchers to assess new algorithms [24,25,45,61].

For the comparative algorithms in this study, we consider the following three meta-heuristics and four hyper-heuristics. The three meta-heuristics are using each individual low-level heuristic (h_1, h_2, and h_3) run on its own, named h1EA, h2EA, and h3EA, respectively, where their evolution phrases are the same as PAPHH. The motivations of choosing the three meta-heuristics for comparison are to (i) analyze the performance of each low-level heuristic when it is applied alone, (ii) analyze the effectiveness of leaning in PAPHH, and (iii) compare the behaviors of PAPHH with each meta-heuristic.

The hyper-heuristics in comparison include RANDHH, MCHH[45], HH-RILA [24], and APHH. Among these approaches, RANDHH is a simple random selection hyper-heuristic without reward calculation and any learning mechanism in the selection process. Because of its simplicity, RANDHH is commonly used as a baseline for comparison [24, 25,45]. MCHH is a hyper-heuristic which employs Markov chains and reinforcement learning to guide the selection procedure. HH-RILA uses reinforcement learning scheme and a switch scheme to learn and select low-level heuristics. The major reasons for choosing MCHH and HH-RILA for comparison are that they are based on learning strategy and are rare studies to solve continuous MOPs. To the extension of our knowledge, HH-RILA is one of the best performing learning-based selection hyper-heuristic for MOPs. APHH is a variant of PAPHH that we proposed. This variant eliminates the perturbation part of PAPHH to verify the effectiveness of the perturbation strategy in PAPHH. APHH also differs from HH-AP on that APHH utilizes the evolution phrase of NSGA-II and raises the level of diversity maintaining process, instead of raising the level of evolution process of ε-MOEA.

4.1.2. Performance evaluation criteria

Comparing qualities of approximation sets obtained by multi-objective optimization algorithms are more complex than single-objective optimization algorithms. We use four widely used performance metrics to assess the quality of an approximation set from different aspects: (i) Spacing metric [55], (ii) Convergence metric [62], (iii) Hypervolume metric [32], and (iv) Running time.

Let $P^* = \{p^1, p^2, \ldots, p^{|P^*|}\}$ be the uniformly distributed Pareto-optimal set and $P = \{x^1, x^2, \ldots, x^{|P|}\}$ be the approximation set obtained by an algorithm.

Spacing metric $S(P)$: evaluates the distribution of the approximation set P. A smaller value of Spacing metric indicates a better uniformly distributed solution set.

$$S(P) = \sqrt{\frac{1}{|P|-1}\sum_{i=1}^{|P|}\left(\overline{d} - d_i^s\right)^2} \tag{8}$$

$$\text{s.t. } d_i^s = \min_{j=1}^{|P|} \sum_{k=1}^{m} \left|f_k(x^i) - f_k(x^j)\right| \text{and } \ \overline{d} = \sum_{i=1}^{|P|} d_i^s \tag{9}$$

Convergence metric $C(P)$**:** evaluates the distance between the approximation set P to the Pareto-optimal set P^*. A smaller value of Convergence metric indicates a better approximation to the Pareto-optimal front. $f_k^{\max}$ and $f_k^{\min}$ are the maximal and minimal value for the k-th objective in P^*, respectively.

$$C(P) = \sum_{i=1}^{|P|} d_i^c / |P| \tag{10}$$

$$\text{s.t. } \ d_i^c = \min_{j=1}^{|P^*|} \sqrt{\sum_{k=1}^{m} \left(\frac{f_k(x^i) - f_k(p^j)}{f_k^{\max} - f_k^{\min}}\right)^2} \tag{11}$$

Hypervolume metric $I_H(P)$**:** evaluates the size of the objective space covered by the approximation set P. A higher value of Hypervolume indicates that the approximation set is well spread and well approximated to the Pareto-optimal front in the objective space. $r = (r_1, r_2, \ldots, r_m)$ is the reference point in the objective space that is dominated by all the solutions in P^*.

$$I_H(P) = volume\left(\bigcup_{x \in P} [f_1(x), r_1] \times \ldots .[f_m(x), r_m]\right) \tag{12}$$

Running time: evaluates the running time of an algorithm for a given problem. The running time is measured in seconds. A smaller value indicates the algorithm runs faster.

Furthermore, to have statistically sound conclusions, a Wilcoxon's rank sum test at a 0.05 significance level is adopted to compare the average performance of a pair of algorithms with respect to each metric averaged over 30 trials. Given a pair of algorithms, A and B, "+", "-", and "≈" denote the performance of A is better than, worse than, and similar to that of B.

4.1.3. Low-level heuristics

Theoretically, any existing diversity mechanism can be incorporated into PAPHH as low-level heuristics. In the experiments, we consider a set of 3 well-known diversity mechanisms as low-level heuristics (denoted $H = \{h_1, h_2, h_3\}$). h_1 is the crowded-comparison strategy from NSGA-II [14]. h_2 is the hyper-grid strategy from PESA-II [12]. h_3 is the density estimation strategy from SPEA2 [13]. Although the three low-level heuristics are no longer considered good, they are still viewed as the baseline in multi-objective research area. Each of them is to promote the spread of an approximation set along the Pareto-optimal front by biasing the search in the relative lonely region. However, the three low-level heuristics are different in the way of how to estimate the degree of the isolation of a solution and how to remove crowded solutions from solution sets.

h_1: the crowded-comparison strategy defines a crowding distance for every solution by calculating the distance of two nearest neighbors on either side of this solution along each objective. A solution with the smallest crowding distance value is most crowded by other solutions and will be removed from the solution set firstly.

h_2: the hyper-grid strategy is a region-based mechanism which divides objective space into hyper-boxes. Each solution has an attribute called squeeze factor to identify its degree of crowding with respect to the total number of solutions in the same hyper-box. The solution with the maximum squeeze factor is chosen to be removed firstly.

h_3: the density estimation strategy is an adaptation of the k-th nearest neighbor method. By calculating the distance of a solution to its k-th nearest solution, the density estimation strategy firstly removes the solution with the minimum distance. If several solutions have the same minimum distance, the 2nd minimum distance is considered and so forth.

4.1.4. Parameter settings

Following the recommendation in Refs. [14,24], all the algorithms use binary tournament selection operator, SBX crossover operator, and polynomial mutation operator in evolutionary steps. s.t. crossover probability $pc = 0.9$, distribution index for SBX is 20, mutation probability $pm = 1/n$ (n is the number of parameters), distribution index for polynomial mutation is 20. The population size $N = 100$ for 2- and 3-objective problems, while $N = 200$ for 5-objective problems. The max evaluation is set to 50 000. That is, the max iteration is 500 for 2- and 3-objective problems, while 250 for 5-objective problems.

The analyses in Ref. [50] show that high values of control parameters (α and β) in the adaptive pursuit strategy are able to produce good results. We set α and β with recommendations in their original papers, $\alpha = 0.8$ and $\beta = 0.8$. Besides, as in Ref. [50], we set $w_{\min} = 1/2l$ and $w_{\max} = 1/2 + 1/2l$. This means that, half probability is allocated to the most suitable low-level heuristic while the other half probability is equally shared by all the low-level heuristics [50]. Other parameters in comparative algorithms are the same as their original papers.

All the source code are written in Java with the help of a popular public-domain software named *Jmetal*, which can be downloaded from (http://jmetal.sourceforge.net/). For 2-objective ZDT and 3-objective DTLZ test suites, the Pareto-optimal front can be download from (http://www.cs.cinvestav.mx/~emoobook). The size of Pareto-optimal set is set 200 in Convergence metric. In Hypervolume metric, the reference point is considered. For 2-objective ZDT and 3-objective DTLZ test suites, we normalize each objective function value in [0,1], and the reference point is set as $r_i = 1$, for $i = 1,\ldots,m$. For 5-objective WFG test suite, the reference point for objective i is set as the largest possible performance, that is $r_i = 2i$, for $i = 1,\ldots,m$ [25]. Experiments are performed under Windows 10 on an AMD Athlon II 3.20 GHz with 8G memory computer.

4.2. Sensitivity analysis of K in PAPHH

PAPHH contains one essential parameter: a threshold K. To investigate the sensitivity of the performance to K, PAPHH is tested on the ZDT and DTLZ test suites under the same experimental settings described in section 4.1 except for K. The value of K ranges in [0, 500] with 5 increment. The analysis is given in terms of the average proportion value over 30 trials versus different K values, illustrated in Fig. 1. The proportion value is the number of selecting the current best low-level heuristic divided by the total number of decision points (or the max iterations) in one trial. Here, the current best low-level heuristic means the low-level heuristic which can get the best diversity performance at the current decision point (or iteration). We could see that average proportion values for ZDT1-4, ZDT6, and DTLZ5 increase dramatically as the values of K are small. Especially, the max average proportion values of the six test problems are greater than 2/3, which is the value of $w_{\max}$ we set in the adaptive pursuit strategy in this experiment. This shows that PAPHH is able to adaptively select the current best low-level heuristic, compared with APHH on ZDT1-4, ZDT6, and DTLZ5. The average proportion values for DTLZ2, DTLZ4, and DTLZ6 increase slightly when K is small, while for DTLZ1, DTLZ3, and DTLZ7, the average proportion values fluctuate around 1/3 which seems the selection procedure is basically random.

Experiments are also conducted using the measurement μ_{norm} [63] to compare the overall performance of heuristics crossing different problem domains or benchmarks. In this experiment, as we are trying to maximize the proportion values, a variant μ_{norm} is used as [24]. Assume every algorithm $a \in Alg$ obtains a set of proportion values for a test problem $p \in Pro$. Alg and Pro are the set of algorithms (PAPHH with different K values) and test problems, respectively. In the experiments, 30 independent trials with 30 proportion values are obtained by an algorithm a on a test problem p. Then, a normalized proportion values $f_{norm}(a, p)$ is computed as $f_{norm}(a, p) = (s_{\max}(p) - ave(a, p))/(s_{\max}(p) - s_{\min}(p))$, where $ave(a, p)$ is the average proportion values averaged from 30 proportion values gotten by the algorithm a on the test problem p, $s_{\max}(p)$, and $s_{\min}(p)$ are the maximal and minimal proportion values from all the algorithms in Alg on the test problem p, respectively. Then $\mu_{norm}(a) = ave_{\forall p \in Pro}\, (f_{norm}(a, p))$, meaning the average performance of algorithm a across all the test problems in Pro. A lower μ_{norm} indicates better performance.

Let $\mu_{norm}(k)$ be the average performance of PAPHH over 30 independent trials across ZDT and DTLZ test suites with $K = k$, $k \in [0, 500]$ with 5 increment. The setting of K in PAPHH that can achieve the lowest μ_{norm} value could be the best parameter across all the ZDT and DTLZ test problems. From Fig. 2, it is clear that the best configuration is $K = 15$. Hence, the setting of $K = 15$ is used in this paper for the rest of the experiments.

The horizontal line stands for the mean μ_{norm} across $k = 0\ldots500$. Remarkably, when K is smaller than 5 and larger than 100, the μ_{norm} values are almost higher than mean μ_{norm}. The fact that PAPHH with smaller K (smaller than 5) works poorly could be explained by that

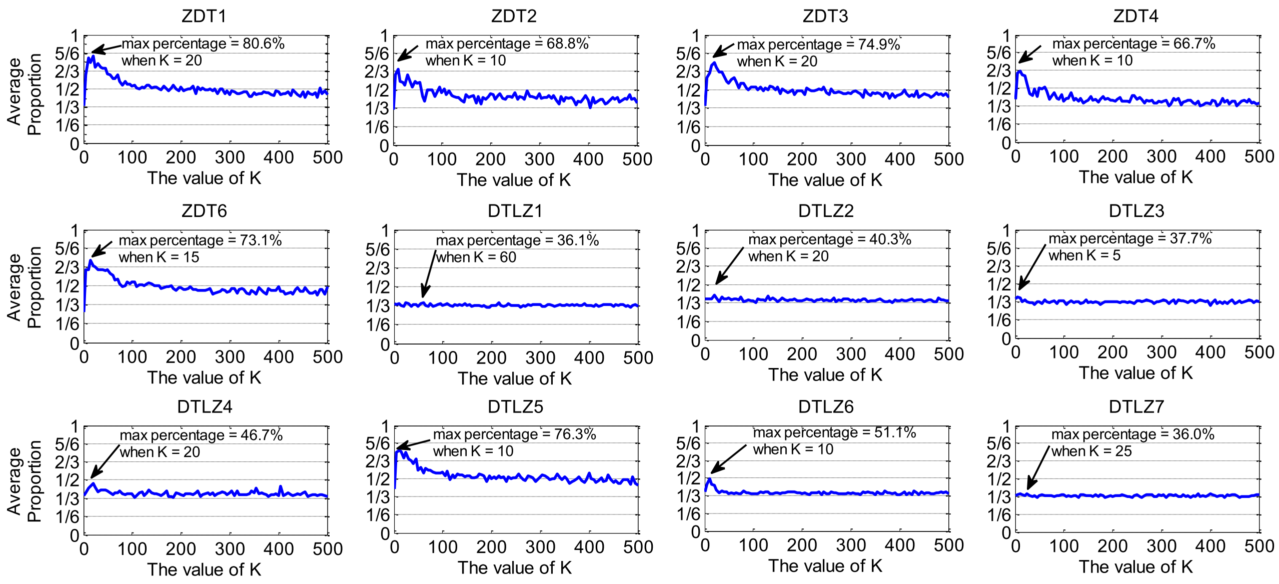

Fig. 1. The average proportions of selecting the best low-level heuristic of each iteration across 30 trials.

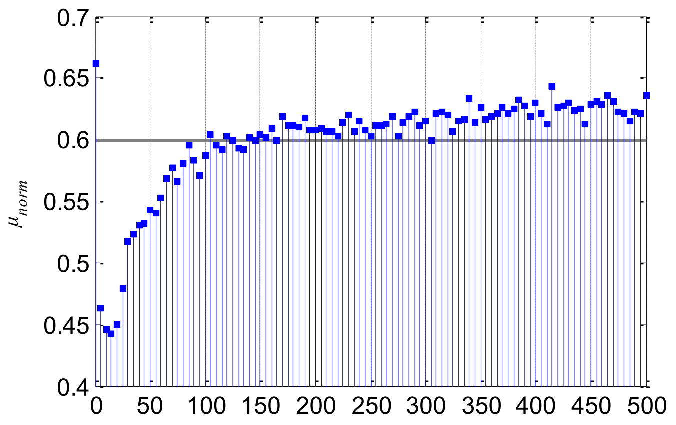

Fig. 2. The μ_{norm} values of PAPHH with different K values over 30 independent trials on ZDT and DTLZ test suites. A lower μ_{norm} value indicates better performance.

PAPHH may wrongly label the algorithm trapped into local optima. On the contrary, PAPHH with larger K (larger than 100) may delay to detect the local optima and thus waste a series of solving iterations. Both cases could weak the effectiveness of the algorithm. Furthermore, one interesting value for K is 500. For the total iteration is set 500 for ZDT and DTLZ test suites, there is no perturbation in PAPHH when $K = 500$. At this point ($K = 500$), PAPHH is APHH essentially. Because $\mu_{norm}(15)$ is much smaller than $\mu_{norm}(500)$, we can conclude again that PAPHH can adaptively select the current best low-level heuristic, compared with APHH.

4.3. Behavior of meta-heuristics and PAPHH

In the following, we compare the performance of meta-heuristics (h1EA, h2EA, and h3EA) with PAPHH over 30 independence runs. The detailed comparison results with respect to Spacing metric, Convergence metric, Hypervolume metric and Running time are illustrated in Table 1. Each meta-heuristic is compared with PAPHH, respectively, by Wilcoxon's rank sum test at a 0.05 significance level to test the significant performance difference. The average utilization rates of each low-level heuristic are presented in Fig. 3 to indicate the frequency of selecting each low-level heuristic across all iterations in PAPHH.

4.3.1. Experimental results of each meta-heuristic and PAPHH

From the Spacing metric values of Table 1, we can make two observations. First, the analysis of h1EA, h2EA, and h3EA shows that three meta-heuristics are non-dominated in terms of diversity. That is, no low-level heuristic performs better than the other two when used in isolation on all the test problems with respect to the Spacing metric. Comparing h1EA, h2EA, and h3EA, h1EA is the best algorithm for three test problems (DTLZ3, DTLZ6, and DTLZ7). h2EA performs the best on DTLZ1. h3EA is overall outstanding than h1EA and h2EA for the rest eight test problems. Our results above are consistent with those in Ref. [19]. Second, the final results obtained by PAPHH are the best for all the test problems except ZDT2, ZDT4, ZDT6, and DTLZ3 compared with h1EA, h2EA, and h3EA. More specifically, For ZDT4 and ZDT6, h3EA is significantly better than PAPHH, and gets the best Spacing metric value. For DTLZ3, h1EA is the best algorithm, while performing similarly to PAPHH. Overall, in the final rank with respect to the Spacing metric across ZDT and DTLZ test suites, PAPHH ranks first. Meanwhile, h1EA, h2EA, and h3EA rank third, fourth, and second, respectively.

From the Convergence metric values of Table 1, we can observe that, comparing the performance among h1EA, h2EA and h3EA, h1EA is the best on seven test problems (ZDT3-6, DTLZ1, DTLZ3-4, and DTLZ6), h3EA is the best for the rest five test problems (ZDT1-2, DTLZ2, DTLZ5, and DTLZ7), while h2EA is the worst meta-heuristic with respect to Convergence metric. These results indicate that using different diversity mechanisms to delete extra solutions in the diversity maintaining step could affect the effectiveness of algorithm's convergence. After comparing PAPHH with h1EA, h2EA, and h3EA, PAPHH works the best on ZDT1. PAPHH is the second best on five test problems (ZDT2-3, DTLZ3, DTLZ5, and DTLZ7), while being the third best on the rest six test problems. The statistics by the Wilcoxon's rank sum tests in Table 1 indicate that PAPHH ranks third across ZDT and DTLZ test suites for Convergence metric among the four algorithms. PAPHH does not rank first for the reason that PAPHH does not learn the convergence performance of each low-level heuristic. However, PAPHH is not the worst algorithm on any test problem. This shows that simple combination (relatives to the Convergence metric) of low-level heuristics could yield acceptable results, at least better than the worst one.

As for the performance comparison on the Hypervolume metric shown in Table 1, among the three meta-heuristics, h3EA outperforms h1EA and h2EA on all the test problems except DTLZ3-4 and DTLZ6, h1EA performs the best on DTLZ3-4, while h2EA is the best on DTLZ6. When comparing PAPHH with the three meta-heuristics, PAPHH can get the best Hypervolume values on 4 out of 12 test problems, including ZDT1-2, DTLZ3, and DTLZ6. For the rest 8 test problems, PAPHH performs the second best on 7 test problems, including ZDT3, ZDT6, DTLZ1-2, DTLZ4-5, and DTLZ7, while performing the third best on ZDT4. The approximation set that is well distributed and approximated to the Pareto-optimal front in the objective space can get a higher value of Hypervolume metric (indicates better performance). As concluded before, the approximation set achieved by PAPHH is always among the top performance for diversity aspect. Although the performance of PAPHH for Hypervolume metric is affected by their not top convergence performance (or may losing some part of Pareto-optimal front), PAPHH is still the overall best algorithm with respect to Hypervolume metric across ZDT and DTLZ test suites, as shown by the Wilcoxon's rank sum tests in Table 1. Although h3EA also ranks first as PAPHH, h3EA works the worst on DTLZ4 and DTLZ6. On the contrary, PAPHH is not the worst algorithm on any ZDT or DTLZ test problem.

According to the Running time values, h1EA, h2EA, and h3EA have different effectiveness in terms of running time. h1EA is the best one, and h3EA is the worst one. This is consistent with that described in Ref. [17]. PAPHH invokes three low-level heuristics (h_1, h_2, and h_3) during the search iterations, that is why the running time values of PAPHH are greater than the smallest running time value and smaller than the largest running time value of three meta-heuristics.

4.3.2. Utilization rates of low-level heuristics in PAPHH

Fig. 3 shows the average utilization rates of h_1, h_2, and h_3 over 30 independent trials in PAPHH. The utilization rate is the number of invocations of a low-level heuristic divided by the number of total iterations during a whole search process of PAPHH. Fig. 3 illustrates which low-level heuristic is selected more frequency. The results show that for nine problems (ZDT1-6, DTLZ2, and DTLZ4-6), the utilization rate of h_3 is significantly larger than that of h_1 and h_2. Especially, the utilization rates of h_3 for ZDT1, ZDT3, and DTLZ5 reach a high value above 66.7% (the value of $w_{\max}$). The reason why the utilization rates of h_3 are with large values for the nine test problems (except for DTLZ6) can be explained by that h_3 is more suitable to deal with these problems than h_2 and h_1 with respect to the Spacing metric according to Table 1. This means PAPHH has the ability to select the overall best low-level heuristics. For DTLZ6, as described in Fig. 3 and Table 1, h_1 is the best low-level heuristic, and h_3 is the worst low-level heuristic when uses in isolation. However, h_3 has an average utilization rate over 50% for DTLZ6. This is due to the different performance of low-level heuristics at different solving stages. The combination of low-level heuristics can explore 'poor' operator to produce good results, and increase its chance to be selected in the future search. Moreover, for the rest three test problems (DTLZ1, DTLZ3, and DTLZ7), the average utilization rates of three low-level heuristics are similar, around 33.3% which is interestingly consistent with Fig. 1.

Table 1
Test results of h1EA, h2EA, h3EA, and PAPHH on 2-objective ZDT and 3-objective DTLZ test suites with respect to Spacing metric, Convergence metric, Hypervolume metric and Running time from 30 trials. The best average value on each test problem is highlight in bold.

Pro.	**Spacing metric value**											
	h1EA			h2EA			h3EA			PAPHH		
	AVG	STD	Rank	AVG	STD	Rank	AVG	STD	Rank	AVG	STD	Rank
ZDT1	0.00773	0.00069	3−	0.00798	0.00060	4−	0.00376	0.00029	2−	**0.00361**	0.00053	1
ZDT2	0.00813	0.00076	3−	0.00840	0.00078	4−	**0.00322**	0.00039	1≈	**0.00322**	0.00051	1
ZDT3	0.00842	0.00079	3−	0.00915	0.00087	4−	0.00426	0.00035	2≈	**0.00415**	0.00068	1
ZDT4	0.00872	0.00060	3−	0.08764	0.20622	4−	**0.00348**	0.00023	1+	0.00785	0.00680	2
ZDT6	0.00877	0.00058	3−	0.01027	0.00674	4−	**0.00240**	0.00024	1+	0.00424	0.00190	2
DTLZ1	0.16963	0.55704	3≈	0.05688	0.05825	2−	0.57059	1.15300	4≈	**0.05662**	0.01944	1
DTLZ2	0.11969	0.02154	3≈	0.11969	0.02349	3≈	0.11403	0.01700	2≈	**0.11208**	0.02043	1
DTLZ3	**0.20161**	0.06138	1≈	1.32196	3.92591	3≈	1.98916	5.33277	4≈	0.44591	0.58607	2
DTLZ4	0.12438	0.03260	3−	0.13867	0.03407	4−	0.08764	0.04519	2≈	**0.08659**	0.03074	1
DTLZ5	0.01054	0.00096	3−	0.01215	0.00171	4−	0.00487	0.00046	2−	**0.00484**	0.00143	1
DTLZ6	0.01919	0.00659	2≈	0.02998	0.02782	3≈	0.03411	0.02570	4−	**0.01864**	0.01066	1
DTLZ7	0.16645	0.04009	2≈	0.18690	0.03211	4−	0.16926	0.02824	3≈	**0.16337**	0.02139	1
Total			32			43			28			15
Final Rank			3			4			2			1

Pro.	**Convergence metric value**											
	h1EA			h2EA			h3EA			PAPHH		
	AVG	STD	Rank	AVG	STD	Rank	AVG	STD	Rank	AVG	STD	Rank
ZDT1	0.00236	0.00015	3−	0.00274	0.00064	4−	0.00230	0.00028	2≈	**0.00223**	0.00019	1
ZDT2	0.00206	0.00018	3≈	0.00225	0.00029	4−	**0.00192**	0.00028	1+	0.00200	0.00019	2
ZDT3	**0.00192**	0.00014	1≈	0.00202	0.00064	4≈	0.00193	0.00028	2≈	0.00193	0.00013	2
ZDT4	**0.00300**	0.00039	1+	0.13409	0.21206	4−	0.00334	0.00061	2≈	0.01363	0.02205	3
ZDT6	**0.00200**	0.00013	1+	0.01480	0.02069	4≈	0.00222	0.00037	2≈	0.00568	0.00666	3
DTLZ1	**0.04870**	0.06286	1+	0.05359	0.12264	2+	0.11927	0.14847	4≈	0.05611	0.05873	3
DTLZ2	0.03822	0.00186	2+	0.04223	0.00169	4−	**0.03780**	0.00168	1+	0.04060	0.00166	3
DTLZ3	**0.33957**	0.47035	1≈	1.08638	0.92277	4≈	0.93650	1.11485	3≈	0.90593	1.39779	2
DTLZ4	**0.04189**	0.00167	1+	0.04702	0.00247	4≈	0.04614	0.00494	2+	0.04688	0.00332	3
DTLZ5	0.00538	0.00047	3≈	0.00677	0.00083	4−	**0.00514**	0.00041	1≈	0.00535	0.00037	2
DTLZ6	**0.03660**	0.03150	1≈	0.04489	0.03913	2≈	0.06918	0.05954	4≈	0.04721	0.03922	3
DTLZ7	0.03408	0.00272	3≈	0.03559	0.00237	4−	**0.03016**	0.00429	1+	0.03291	0.00337	2
Total			21			44			25			29
Final Rank			1			4			2			3

Pro.	**Hypervolume metric value**											
	h1EA			h2EA			h3EA			PAPHH		
	AVG	STD	Rank	AVG	STD	Rank	AVG	STD	Rank	AVG	STD	Rank
ZDT1	0.66005	0.00026	3−	0.65806	0.00045	4−	0.66163	0.00013	2≈	**0.66164**	0.00013	1
ZDT2	0.32694	0.00023	3−	0.32517	0.00050	4−	**0.32844**	0.00008	1≈	**0.32844**	0.00023	1
ZDT3	0.51522	0.00009	3−	0.51306	0.00099	4−	**0.51552**	0.00005	1+	0.51542	0.00016	2
ZDT4	0.65929	0.00068	2≈	0.65649	0.00126	4−	**0.66057**	0.00097	1+	0.65915	0.00141	3
ZDT6	0.39806	0.00034	3−	0.39671	0.00076	4−	**0.40055**	0.00015	1+	0.40026	0.00063	2
DTLZ1	0.75433	0.00811	3−	0.73778	0.01710	4−	**0.78081**	0.00300	1+	0.76337	0.00593	2
DTLZ2	0.37444	0.00501	3≈	0.35121	0.00996	4−	**0.40507**	0.00221	1+	0.37881	0.01380	2
DTLZ3	0.24165	0.11500	2≈	0.13167	0.15000	4−	0.19201	0.15300	3≈	**0.27435**	0.08440	1
DTLZ4	**0.37539**	0.00466	1≈	0.36418	0.03030	3−	0.32233	0.13200	4−	0.36631	0.04160	2
DTLZ5	0.09217	0.00017	3−	0.09020	0.00047	4−	**0.09273**	0.00014	1≈	0.09259	0.00037	2
DTLZ6	0.07690	0.01330	3−	0.08607	0.00881	2−	0.07205	0.01740	4−	**0.08684**	0.00943	1
DTLZ7	0.28352	0.00518	3≈	0.25229	0.01210	4−	**0.29739**	0.00210	1+	0.28386	0.01030	2
Total			32			45			21			21
Final Rank			3			4			1			1

Pro.	**Running time value**											
	h1EA			h2EA			h3EA			PAPHH		
	AVG	STD	Rank	AVG	STD	Rank	AVG	STD	Rank	AVG	STD	Rank
ZDT1	**554.7**	69.02	1+	981.73	19.35	2+	2027.23	111.74	4−	1776.63	175.55	3
ZDT2	**547.43**	54.87	1+	935.47	13.78	2+	1855.3	71.5	4−	1270.53	466.61	3
ZDT3	**548.47**	69.05	1+	904.07	22.17	2+	1922.57	56.03	4−	1571.63	333.07	3
ZDT4	**490**	35.48	1+	712.07	34.48	2+	1324.97	41.13	4−	1006.1	229.17	3
ZDT6	**447.7**	34.17	1+	813.8	17.16	2+	1503.47	37.93	4−	1388.17	134.64	3
DTLZ1	**670.53**	61.33	1+	1593.67	155.54	3−	1985.83	142.18	4−	1414.53	307.6	2
DTLZ2	**561.5**	30.64	1+	2909.5	71.06	3−	3503.37	27.12	4−	2412.2	765.87	2
DTLZ3	**616.2**	23.78	1+	782.6	107.41	2+	1061.23	115.49	4−	923.17	165.7	3
DTLZ4	**570.97**	6.48	1+	2557.9	471.01	3−	2927.93	888.06	4−	2284.97	712.11	2
DTLZ5	**432.1**	10.44	1+	1828.6	24.42	2+	2344.63	27.23	4−	2007.5	215.5	3
DTLZ6	**606.8**	12.14	1+	1723.33	55.76	2+	2113.87	36.56	4−	1770.43	299.27	3
DTLZ7	**606.2**	20.86	1+	2504.57	45	3−	2945.97	61.29	4−	2072.3	538.4	2
Total			12			28			48			32
Final Rank			1			2			4			3

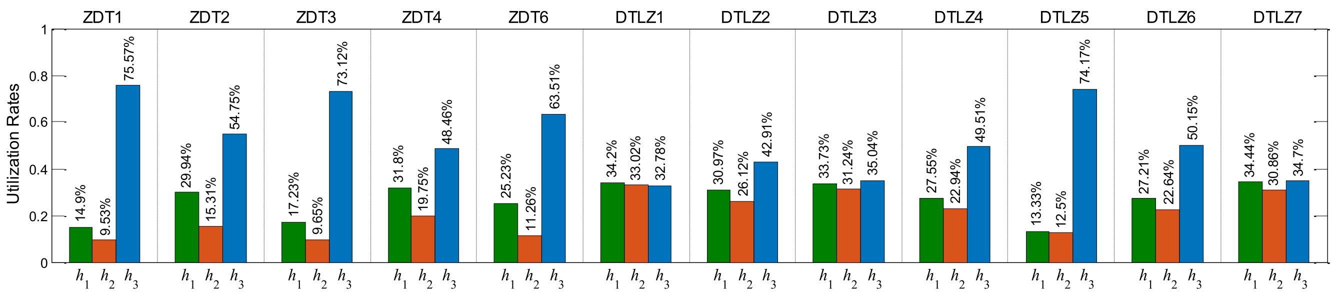

Fig. 3. The average utilization rates of h_1, h_2, and h_3 in PAPHH.

4.3.3. *Effectiveness of combining meta-heuristic with other low-level heuristic*

In this experiment, we intend to verify the effectiveness of combining different algorithms, by conducting single-point replacement tests. In the tests, only single decision point of each meta-heuristic (h1EA, h2EA, and h3EA) is replaced by other low-level heuristics, to examine. For example, h_1 in h1EA is replaced by h_2 (denoted as 'h1EA→h_2') in the ith decision point, i∈{1,2,...,500}, to examine if combining h_2 and h_1 could achieve better solutions. Table 2 gives statistics about the average percentage of improvement times after replacing the ith decision point of one meta-heuristic with other low-level heuristics by comparing with the meta-heuristic (averaged from $i = 1$ to 500, each replacement is run for 30 trials). As observed from Table 2, most of the replacements can get improvements with a probability over 50% for the Spacing, Convergence, and Hypervolume metrics. This observation confirms the assumption that combining multiple low-level heuristics is able to achieve better performance. However, we should note that we do not know in advance which decision point or low-level heuristic can achieve an improvement. We also do not know which replacement can bias to a certain metric, e.g. Spacing metric or Convergence metric. This is also the reason why hyper-heuristic is necessary to adaptively select the most suitable low-level heuristics.

Table 2 also provides the average number of deleted solutions averaged from the ith single-point replacement, i∈{1,2,...,500}. As described in previous subsection, only if $|P_{t+1}| > N$, the extra $|P_{t+1}|$-N solutions are deleted, where $|P_{t+1}|$-N ranges in [0, 100]. We can see that on DTLZ2, more than 50 solutions are deleted from h1EA, h2EA, and h3EA on average. DTLZ3 has the fewest deleted solutions, which is also greater than 15. Fig. 4 shows that the number of deleted solutions of h1EA on ZDT1, and h3EA on DTLZ7 is increasing generally as the process evolving. This tendency can also be found in h1EA, h2EA, and h3EA on other ZDT and DTLZ test problems. An explanation can be that the number of non-dominated solutions increases as evolution goes on. More solutions need to be deleted and the effect of diversity control on solution sets is increasing too.

4.4. *Behavior of other multi-objective hyper-heuristics and PAPHH*

The experiments are conducted to examine the performance of PAPHH compared with four hyper-heuristics: RANDHH, MCHH[45], HH-RILA [24], and APHH. The four comparative hyper-heuristics use the same low-level heuristics with PAPHH. In this section, Spacing metric and Hypervolume metric are used as the performance indicators. We choose the two metrics for two reasons. First, Spacing metric is used to verify the effectiveness of hyper-heuristics in diversity aspect. Second, maximizing the Hypervolume metric is equivalent to optimize the overall aspect (the diversity and the convergence) of approximation sets [24].

4.4.1. *Experimental results of each hyper-heuristic*

The performance results on ZDT and DTLZ test suites with respect to Spacing metric and Hypervolume metric over 30 independence runs are displayed in Table 3. The Wilcoxon's rank sum test at a 0.05 significance level is given to test if there is a statistically significant performance

Table 2
The average percentage of improvement times and the average number of deleted solutions after replacing single decision point in each meta-heuristic with other low-level heuristics. The improvement value that is greater than 50% is highlight in bold.

Metrics		ZDT1	ZDT2	ZDT3	ZDT4	ZDT6	DTLZ1	DTLZ2	DTLZ3	DTLZ4	DTLZ5	DTLZ6	DTLZ7
h1EA→h_2	Spacing	**93.6%**	**99.0%**	**90.8%**	**89.4%**	**71.6%**	48.0%	20.0%	32.4%	18.6%	**89.6%**	**91.6%**	26.2%
	Convergence	48.8%	**76.2%**	**65.8%**	26.8%	**61.4%**	24.6%	**74.2%**	**78.4%**	43.4%	14.6%	**97.6%**	**85.4%**
	Hypervolume	**95.8%**	**99.0%**	**67.6%**	**57.0%**	**80.8%**	45.8%	36.2%	**88.4%**	31.6%	**97.6%**	**96.2%**	**62.0%**
h1EA→h_3	Spacing	**91.2%**	**98.8%**	**90.0%**	**99.6%**	**69.8%**	**72.0%**	22.2%	33.6%	21.0%	**88.2%**	**97.0%**	23.4%
	Convergence	**52.6%**	**80.4%**	**70.2%**	48.0%	**81.2%**	**51.0%**	**83.2%**	**80.4%**	**52.0%**	17.4%	**99.4%**	**86.4%**
	Hypervolume	**94.6%**	**98.6%**	**72.4%**	**62.4%**	**78.8%**	**65.4%**	35.4%	**88.0%**	43.0%	**98.8%**	**95.4%**	**66.4%**
Average number of deleted solutions in h1EA		30.78	28.67	26.62	19.71	31.08	28.58	55.40	17.17	52.18	36.36	22.41	45.27
h2EA→h_1	Spacing	**66.4%**	**74.4%**	**57.4%**	**66.4%**	48.8%	**63.4%**	38.4%	**78.0%**	**56.2%**	**63.4%**	27.2%	**69.4%**
	Convergence	**92.2%**	**73.6%**	**79.2%**	**78.6%**	45.2%	**69.8%**	**87.8%**	**99.6%**	38.8%	**90.2%**	**61.6%**	**76.6%**
	Hypervolume	**57.4%**	44.4%	41.0%	**57.4%**	29.6%	**51.2%**	**60.4%**	**99.4%**	**63.6%**	**74.8%**	43.4%	35.8%
h2EA→h_3	Spacing	**58.6%**	**74.0%**	**58.6%**	**72.2%**	**50.8%**	**88.8%**	38.8%	**78.8%**	**51.0%**	**65.6%**	32.4%	**68.8%**
	Convergence	**79.6%**	**70.0%**	**74.8%**	**82.8%**	48.0%	**74.4%**	**88.4%**	**99.4%**	41.2%	**74.8%**	**63.2%**	**79.4%**
	Hypervolume	**53.0%**	44.6%	39.4%	**62.8%**	34.6%	48.6%	**61.6%**	**99.8%**	**60.4%**	**78.0%**	47.2%	37.4%
Average number of deleted solutions in h2EA		28.96	27.88	25.63	18.75	29.63	30.04	54.10	16.80	49.45	36.98	29.31	47.87
h3EA→h_1	Spacing	41.6%	**64.8%**	44.2%	32.8%	34.4%	**73.8%**	40.0%	**71.8%**	**54.6%**	**71.4%**	**85.4%**	**50.2%**
	Convergence	**88.6%**	**86.6%**	**53.2%**	**70.0%**	**86.8%**	**74.8%**	**67.2%**	**99.4%**	**75.2%**	23.6%	**99.8%**	**89.0%**
	Hypervolume	**76.4%**	**71.0%**	36.4%	**52.0%**	**99.8%**	**97.2%**	**63.0%**	**99.6%**	**75.4%**	**77.8%**	**99.8%**	**57.8%**
h3EA→h_2	Spacing	36.6%	**53.4%**	42.4%	23.6%	40.6%	**72.6%**	36.8%	**68.0%**	**52.2%**	**61.6%**	**86.8%**	**53.4%**
	Convergence	**86.8%**	**74.2%**	**61.2%**	47.2%	**76.4%**	**74.2%**	**74.2%**	**99.6%**	**73.0%**	16.6%	**99.8%**	**97.2%**
	Hypervolume	**70.6%**	**58.4%**	34.0%	45.6%	**99.4%**	**99.6%**	**63.6%**	**98.8%**	**77.6%**	**65.4%**	**99.8%**	**59.2%**
Average number of deleted solutions in h3EA		32.16	29.99	28.72	20.14	28.43	28.55	58.55	15.63	50.04	39.98	24.88	48.30

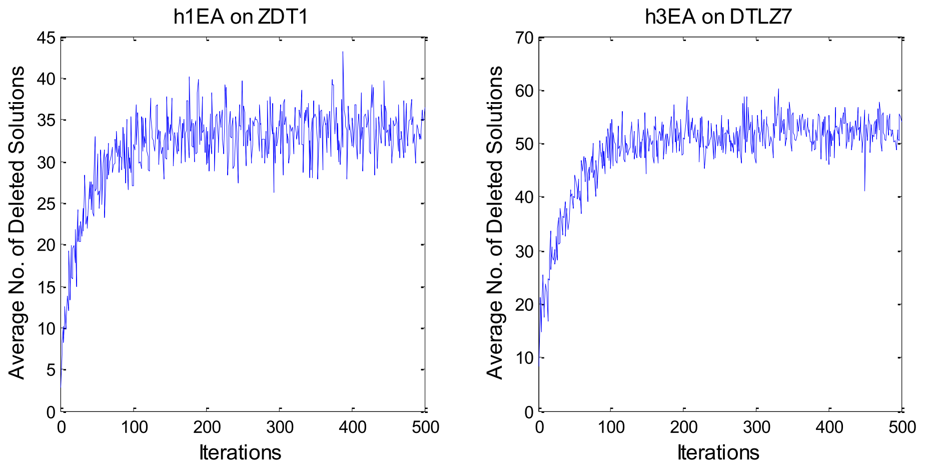

Fig. 4. The number of deleted solutions in h1EA on ZDT1 and h3EA on DTLZ7 during the evolutionary process averaged from 30 trials.

difference between each comparative hyper-heuristic and PAPHH. The statistical results summarized in Table 3 can be concluded as follows.

With respect to the Spacing metric, PAPHH is always among the top performing hyper-heuristic on the ZDT and DTLZ test suites. PAPHH performs significantly better than (or similar to) RANDHH, MCHH, and APHH on all test problems, while PAPHH is significantly better than (or similar to) HH-RILA except ZDT6. These indicate that the PAP strategy in PAPHH has superior ability than the learning strategies (or no learning) in comparative hyper-heuristics to select suitable low-level heuristics across ZDT and DTLZ test suites.

According to the metric of Hypervolume, PAPHH obviously performs the best on ZDT and DTLZ test suites in the overall. Specifically, PAPHH works significantly better than (or similar to) RANDHH and APHH on all the test problems, while significantly outperforming (or similar to) MCHH on all the test problems except for DTLZ3. Besides, PAPHH is significantly better than (or similar to) HH-RILA on all the test problems except for ZDT4 and ZDT6.

The Final rank in Table 3 assesses the generality level of a hyper-heuristic across different test problems. The empirical ranking results show PAPHH delivers the best performance on both Spacing and Hypervolume metric, while HH-RILA is the second-best performing algorithm. MCHH and APHH follow HH-RILA. RANDHH is the worst

Table 3
Test results of comparing hyper-heuristics and PAPHH on 2-objective ZDT and 3-objective DTLZ test suites with respect to Spacing metric and Hypervolume metric from 30 trials. The best average value on each test problem is highlight in bold.

Pro.	**Spacing metric value**														
	RANDHH			MCHH[45]			HH-RILA[24]			APHH			PAPHH		
	AVG	STD	Rank	AVG	STD	Rank	AVG	STD	Rank	AVG	STD	Rank	AVG	STD	Rank
ZDT1	0.00719	0.00096	5−	0.00571	0.00156	3−	0.00386	0.00036	2−	0.00660	0.00122	4−	**0.00361**	0.00053	1
ZDT2	0.00701	0.00108	5−	0.00622	0.00183	3−	0.00331	0.00039	2≈	0.00691	0.00105	4−	**0.00322**	0.00051	1
ZDT3	0.00798	0.00148	5−	0.00701	0.00205	3−	0.00437	0.00074	2≈	0.00745	0.00103	4−	**0.00415**	0.00068	1
ZDT4	0.02188	0.03670	2−	0.04395	0.12197	3≈	0.06769	0.25257	4−	0.07832	0.17565	5−	**0.00785**	0.00680	1
ZDT6	0.01980	0.03307	5−	0.00854	0.00864	4−	**0.00329**	0.00307	1+	0.00649	0.00139	3−	0.00424	0.00190	2
DTLZ1	0.15272	0.39599	5≈	0.09307	0.16947	2≈	0.13643	0.29451	4≈	0.12656	0.21799	3≈	**0.05662**	0.01944	1
DTLZ2	0.12686	0.01922	5−	0.12235	0.02372	3≈	0.12558	0.02560	4−	**0.11127**	0.01923	1≈	0.11208	0.02043	2
DTLZ3	1.81845	2.69686	5≈	0.49690	1.44440	2≈	1.68274	2.54584	4≈	0.57116	0.60236	3≈	**0.44591**	0.58607	1
DTLZ4	0.13288	0.02971	5−	0.11424	0.04085	3−	0.09791	0.05204	2−	0.12321	0.03200	4−	**0.08659**	0.03074	1
DTLZ5	0.00958	0.00121	5−	0.00919	0.00268	4−	0.00530	0.00100	2−	0.00882	0.00228	3−	**0.00484**	0.00143	1
DTLZ6	0.04915	0.04334	5−	0.02674	0.02740	3≈	0.03691	0.03722	4−	0.02415	0.01075	2−	**0.01864**	0.01066	1
DTLZ7	0.17578	0.03955	3≈	0.18021	0.05669	5≈	0.17821	0.04081	4≈	0.16352	0.04285	2≈	**0.16337**	0.02139	1
Total			55			38			35			38			14
Final Rank			5			3			2			3			1
Pro.	**Hypervolume metric value**														
	RANDHH			MCHH[45]			HH-RILA[24]			APHH			PAPHH		
	AVG	STD	Rank	AVG	STD	Rank	AVG	STD	Rank	AVG	STD	Rank	AVG	STD	Rank
ZDT1	0.65983	0.00064	5−	0.66052	0.00107	3−	0.66160	0.00017	2≈	0.66014	0.00060	4−	**0.66164**	0.00013	1
ZDT2	0.32678	0.00070	5−	0.32705	0.00104	3−	0.32841	0.00012	2≈	0.32688	0.00042	4−	**0.32844**	0.00023	1
ZDT3	0.51472	0.00053	3−	0.51471	0.00075	4−	0.51531	0.00013	2−	0.51433	0.00098	5−	**0.51542**	0.00016	1
ZDT4	0.65890	0.00151	5≈	0.65957	0.00123	2≈	**0.66037**	0.00084	1+	0.65900	0.00121	4≈	0.65915	0.00141	3
ZDT6	0.39842	0.00101	5−	0.39871	0.00148	4−	**0.40079**	0.00028	1+	0.39883	0.00093	3−	0.40026	0.00063	2
DTLZ1	0.76106	0.00802	5≈	0.76284	0.01200	3≈	**0.76601**	0.00981	1≈	0.76270	0.00664	4≈	0.76337	0.00593	2
DTLZ2	0.37128	0.00808	5−	0.37756	0.01150	3≈	**0.38060**	0.01263	1≈	0.37682	0.00911	4≈	0.37881	0.01380	2
DTLZ3	0.19997	0.15200	3≈	**0.30492**	0.12100	1+	0.19066	0.15483	4≈	0.18219	0.16000	5≈	0.27435	0.08440	2
DTLZ4	0.35674	0.05690	4−	0.35964	0.05180	3≈	0.34565	0.09448	5≈	**0.37096**	0.03200	1≈	0.36631	0.04160	2
DTLZ5	0.09169	0.00052	5−	0.09203	0.00065	3−	**0.09273**	0.00016	1≈	0.09180	0.00046	4−	0.09259	0.00037	2
DTLZ6	0.07569	0.01430	5−	0.08213	0.00829	2≈	0.07726	0.01935	3−	0.07646	0.01920	4−	**0.08684**	0.00943	1
DTLZ7	0.27540	0.00930	4−	0.27346	0.00909	5−	0.27770	0.01644	2≈	0.27686	0.00768	3−	**0.28386**	0.01030	1
Total			54			36			25			45			20
Final Rank			5			3			2			4			1

algorithm, ranking 5.

4.4.2. Effectiveness of the PAP strategy in PAPHH

Fig. 5 presents the selection probabilities of h_1, h_2, and h_3 averaged from 30 independent trials versus solving iterations on DTLZ2 and DTLZ6 obtained by APHH and PAPHH separately. Both APHH and PAPHH give h_3 higher selection probabilities than h_1 and h_2. This preference is more obvious in PAPHH. The selection probability of both APHH and PAPHH on DTLZ2 and DTLZ6 are basically random in the beginning. The difference of the two hyper-heuristics comes up in later stages. It is worth noting that, for APHH, there is no obvious difference in selection probabilities between the two low-probability low-level heuristics (we mean h_1 and h_2 here for the two test problems). But this difference is very clear in PAPHH. Such effect may be due to the selection probability reset mechanism in the PAP strategy of PAPHH.

As described in Section 4.2, the max average proportion values of PAPHH on ZDT1-4, ZDT6, and DTLZ5 are larger than $w_{\max}$ in the adaptive pursuit strategy ($w_{\max} = 2/3$ in APHH in Section 4.2). This shows a superiority of PAPHH in selecting the current best low-level heuristics over APHH. In the following, an investigation of the effectiveness of APHH and PAPHH with different $w_{\min}$ and $w_{\max}$ values are conducted. Five values are considered for $w_{\min} \in \{0.1, 0.15, 1/6, 0.2, 0.25\}$, and the corresponding $w_{\max}$ values are set as $\{0.8, 0.7, 2/3, 0.6, 0.5\}$. The measurement used in this section is μ_{norm} from three aspects: proportions of selecting the current best low-level heuristics, Spacing metric values and Hypervolume metric values. As proportion values and Hypervolume values are needed to be maximized, we use the same μ_{norm} described in Section 4.2. Spacing values should be minimized, so we modify $f_{norm}(a, p)$ in μ_{norm} as $f_{norm}(a, p) = (ave(a, p) - s_{\min}(p))/(s_{\max}(p) - s_{\min}(p))$. After modification, lower μ_{norm} values indicate better performance.

The μ_{norm} values of APHH and PAPHH with different $w_{\min}$ and $w_{\max}$ values over 30 independent trials across ZDT and DTLZ test suites are plot in Fig. 6. The horizontal axis indicates the $w_{\min}$ values with their corresponding $w_{\max}$ values in the bracket bellow. The vertical axis shows the μ_{norm} values. Subgraph (a), (b), and (c) are for proportion, Spacing and Hypervolume, respectively. From subgraph (a), both APHH and PAPHH with lower $w_{\min}$ (higher $w_{\max}$) values have more chance to select the current best low-level heuristics except $w_{\min} = 0.15$ ($w_{\max} = 0.7$) for PAPHH. Besides, from subgraph (b) and (c), the performance of PAPHH is superior to APHH when they are set the same $w_{\min}$ and $w_{\max}$ values on both Spacing and Hypervolume metrics, except $w_{\min} = 0.15$ ($w_{\max} = 0.7$) for Spacing and $w_{\min} = 0.25$ ($w_{\max} = 0.5$) for Hypervolume.

4.4.3. Comparison of each hyper-heuristic on the utilization rates of low-level heuristics

Fig. 7 shows the average utilization rates of h_1, h_2, and h_3 averaged over 30 trials on ZDT and DTLZ test suites produced by MCHH, HH-RILA, APHH, and PAPHH. Test results of RANDHH are not visualized for its utilization rates of h_1, h_2, and h_3 are similarly around 33.3%. From Fig. 7, the following observations are summarized. PAPHH and HH-RILA provide an overwhelming bias towards using the best performing low-level heuristic with the Spacing metric as the guidance in the learning mechanism. Specifically, PAPHH selects h_3 more frequently on all the test problems except DTLZ1, DTLZ3, and DTLZ7, while HH-RILA selects h_3 more frequently except DTLZ3. Whereas, the bias in MCHH and APHH is not that significant. However, it can still figure out that MCHH and APHH give more chances to h_3 than h_1 and h_2 for almost all the test problems.

4.4.4. An analysis of search process

In this section, we compare PAPHH with all the comparative algorithms on DTLZ2 and DTLZ6. Fig. 8 shows the average Spacing and Hypervolume values obtained by the eight algorithms versus the number of iterations over 30 trials on DTLZ2 and DTLZ6.

For the Spacing metric, eight algorithms perform somewhat similar in the beginning and remain stable after about 50 iterations on DTLZ2. At later stages, h3EA is superior to h1EA and h2EA, while PAPHH and

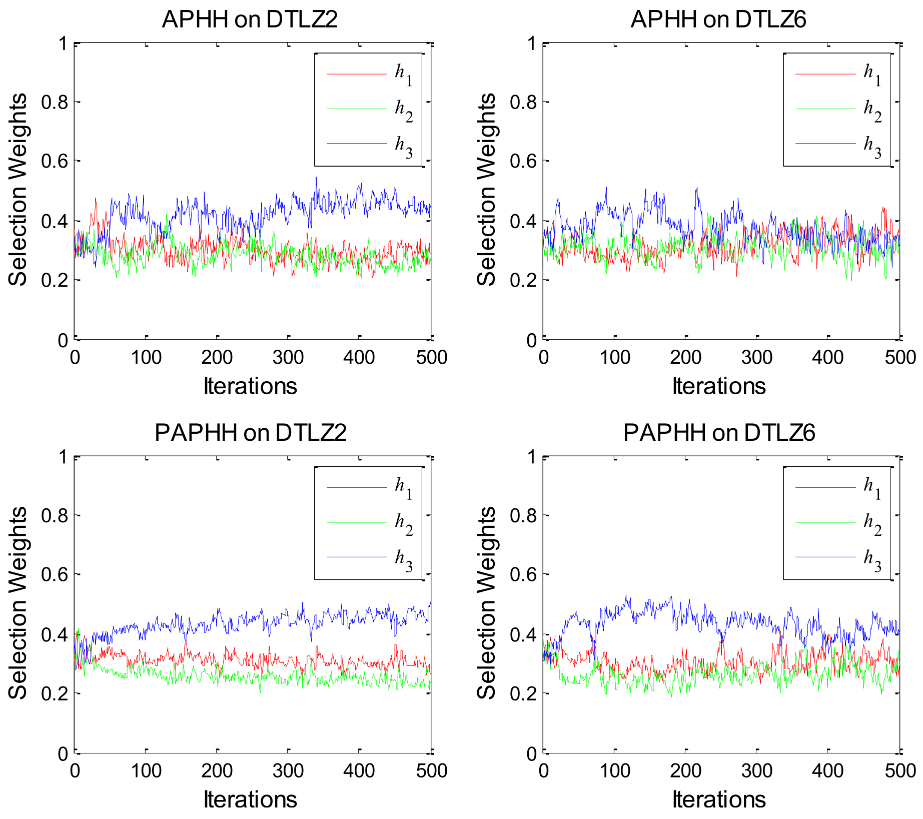

Fig. 5. The average selection probabilities of each low-level heuristic in APHH and PAPHH during the evolutionary process averaged from 30 trials on DTLZ2 and DTLZ6.

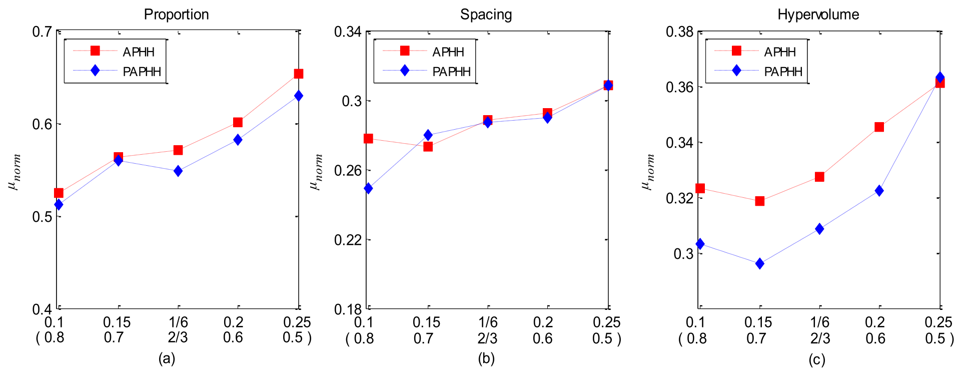

Fig. 6. The μ_{norm} values of APHH and PAPHH with different $w_{\min}$ and $w_{\max}$ values over 30 independent trials on ZDT and DTLZ test suites with respect to three criteria. A lower μ_{norm} value indicates better performance.

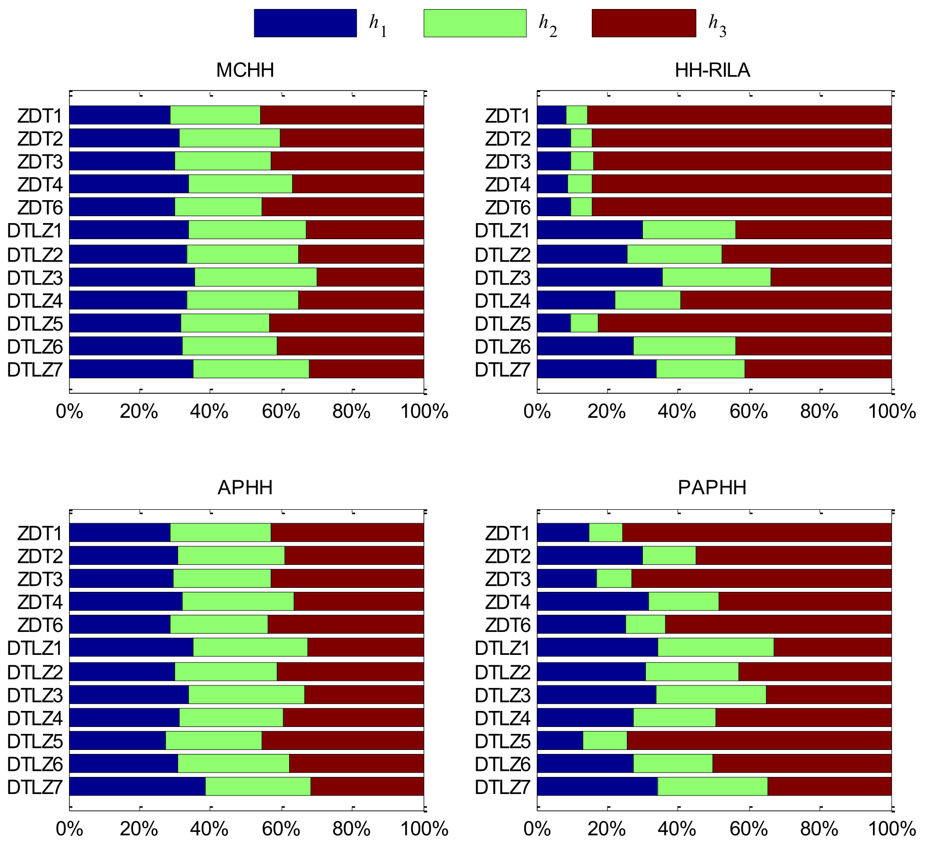

Fig. 7. The average utilization rates of h_1, h_2, and h_3 in MCHH[45], HH-RILA [24], APHH, and PAPHH.

APHH have similar diversity performance on DTLZ2. For DTLZ6, MCHH, and h2EA decrease rapidly with respect to the Spacing metric, especially after around 350 iterations. Later, PAPHH and HH-RILA begin with a fast decrease in Spacing near the end of iterations.

The Hypervolume values are zero at early stages for all the eight algorithms on DTLZ2 and DTLZ6. Especially for DTLZ6, the hypervolume values do not increase until running 300 iterations. This is because the solution sets obtained by eight algorithms for DTLZ6 are still far away from the Pareto-optimal front. h2EA and MCHH increase their Hypervolume values firstly since their solution sets converge faster than that of other algorithms on DTLZ6. However, PAPHH catches up MCHH and h2EA later with respect to the Hypervolume metric on DTLZ6.

Fig. 9 plots the final solution sets with the lowest Spacing metric value from 30 runs obtained by different algorithms on DTLZ2. It can be seen from these figures that h3EA, PAPHH, HH-RILA, APHH, and MCHH get a well distributed solution set. Meanwhile, h1EA, h2EA, and RANDHH perform worse in the diversity aspect. On DTLZ6, as shown in Fig. 10, h2EA, h3EA, RANDHH could yield some solutions that are not close to the Pareto-optimal front. In the contrary, the rest five algorithms can approximate the Pareto-optimal front very well and their solution sets can be well distributed in the objective space.

4.5. Performance comparison with other multi-objective hyper-heuristics on many-objective WFG test suite

Many-objective problems have attracted growing attention in real-world applications. In this section, we will investigate the effectiveness of PAPHH for dealing with many-objective problems. Experiments are conducted on nine 5-objective test problems from WFG test suite comparing RANDHH, MCHH, HH-RILA, APHH, and PAPHH after

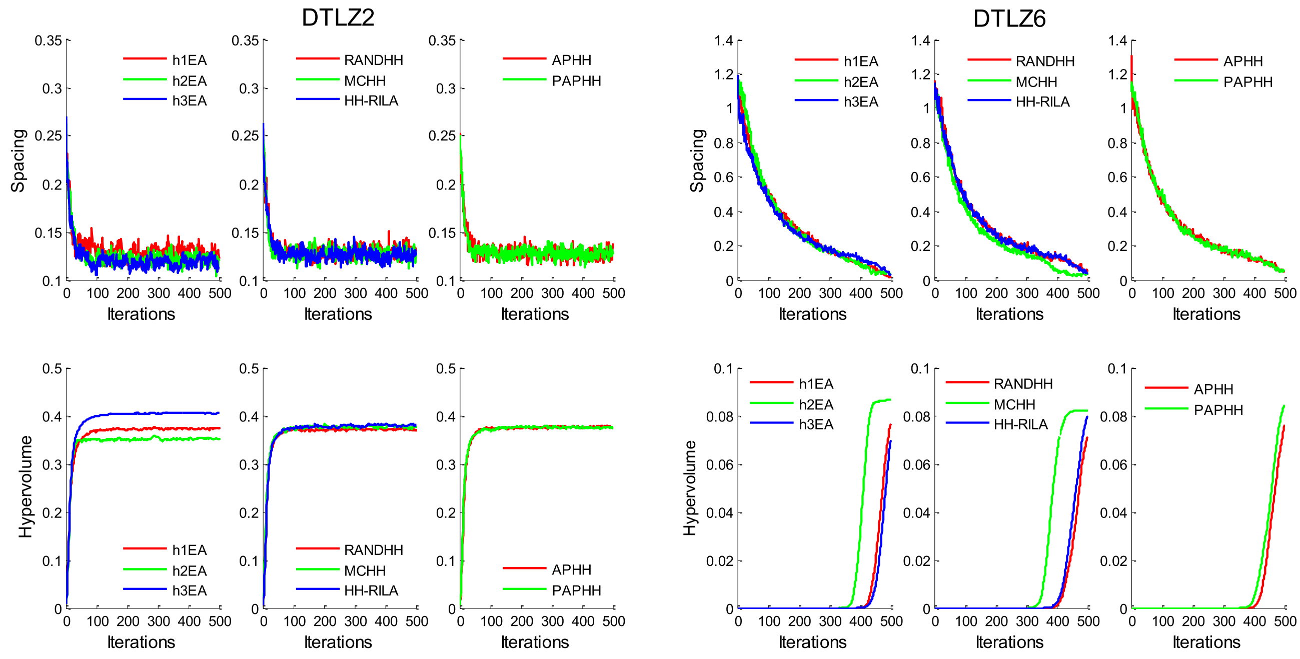

Fig. 8. The average Spacing and Hypervolume values found by comparing algorithms and PAPHH during the evolutionary process from 30 trials on DTLZ2 and DTLZ6.

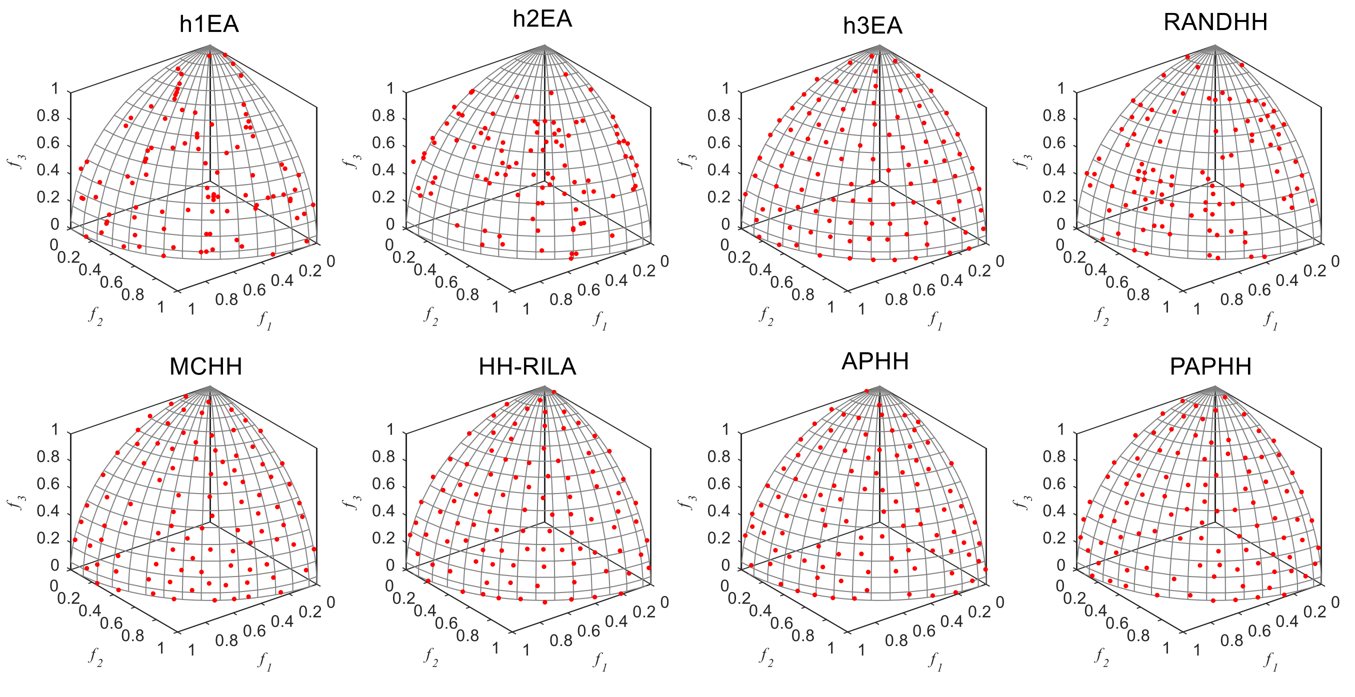

Fig. 9. Plots of the final approximation sets with the lowest Spacing metric values found by comparing algorithms and PAPHH from 30 trials in the objective space on DTLZ2.

running 30 independent trials. Both Spacing metric and Hypervolume metric are employed as the performance metrics here.

For detailed analysis, the statistic results and Wilcoxon's rank sum test values at a 0.05 significance level are summarized in Table 4. Firstly, according to the average Spacing metric values, as expected, PAPHH is always among the top performing algorithms on all the test problems except for WFG4 and WFG8-9. To be specific, the performance of PAPHH on WFG4 and WFG8-9 with respect to Spacing metric is similar to the best corresponding algorithm HH-RILA. Secondly, comparing the average performance of a pair of hyper-heuristics with respect to the Hypervolume metric value, it is good to report that PAPHH performs significantly better than RANDHH, MCHH, and APHH on 7 out of 9 WFG test problems (WFG1 and WFG4-9 for RANDHH and APHH, while WFG2-6 and WFG8-9 for MCHH). Meanwhile, PAPHH is significantly better than (or similar to) HH-RILA on 4 out of 9 problems for Hypervolume metric, including WFG1-2 and WFG5-6. Interestingly, RANDHH is the

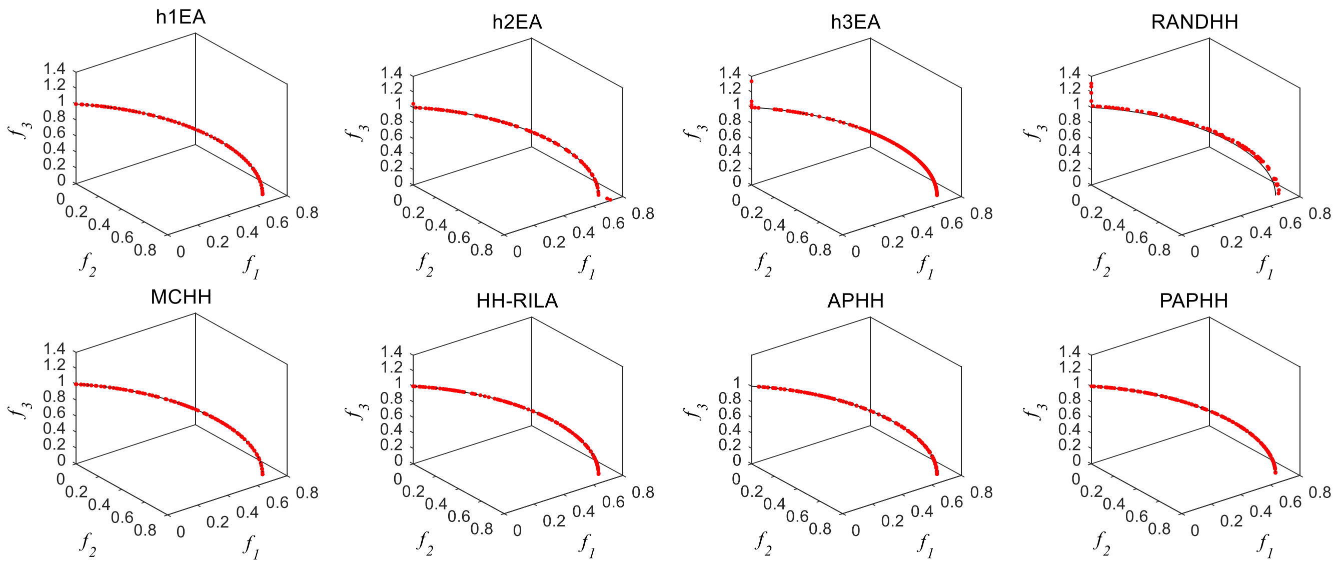

Fig. 10. Plots of the final approximation sets with the lowest Spacing metric values found by comparing algorithms and PAPHH from 30 trials in the objective space on DTLZ6.

Table 4
Test results of comparing hyper-heuristics and PAPHH on 5-objective WFG test problems respect to Spacing metric and Hypervolume metric from 30 trials. The best average value on each test problem is highlight in bold.

Pro.	**Spacing metric value**														
	RANDHH			MCHH[45]			HH-RILA[24]			APHH			PAPHH		
	AVG	STD	Rank	AVG	STD	Rank	AVG	STD	Rank	AVG	STD	Rank	AVG	STD	Rank
WFG1	0.06351	0.02440	3≈	0.06172	0.01228	2≈	0.07495	0.01318	4−	0.07749	0.00939	5−	**0.05200**	0.01961	1
WFG2	0.59993	0.09167	5−	0.53157	0.12808	3≈	0.52307	0.07111	2−	0.57525	0.08253	4−	**0.43427**	0.08696	1
WFG3	0.50986	0.10278	4≈	0.50738	0.10073	3≈	0.45806	0.05957	2≈	0.54543	0.10794	5−	**0.44765**	0.06972	1
WFG4	0.88922	0.12615	5≈	0.80796	0.10334	2≈	**0.79654**	0.09090	1≈	0.86965	0.10621	4≈	0.81510	0.17113	3
WFG5	1.02013	0.13962	4−	0.98761	0.15399	5−	0.91114	0.10854	2≈	0.93533	0.14903	3≈	**0.86603**	0.09623	1
WFG6	1.12142	0.10135	4≈	1.07251	0.12361	3≈	1.00581	0.10756	2≈	1.15879	0.15662	5−	**0.97824**	0.21796	1
WFG7	1.07217	0.12628	5−	1.04737	0.08509	4−	0.92952	0.09054	2≈	0.94498	0.08628	3≈	**0.90018**	0.12916	1
WFG8	1.04606	0.12648	5≈	1.02015	0.21285	4≈	**0.91524**	0.03919	1≈	0.96140	0.14724	3≈	0.95400	0.08875	2
WFG9	0.99960	0.13117	3≈	1.00036	0.09941	4≈	**0.91941**	0.09661	1≈	1.06310	0.11547	5−	0.96754	0.11735	2
Total			38			30			17			37			13
Final Rank			5			3			2			4			1
Pro.	**Hypervolume metric value**														
	RANDHH			MCHH[45]			HH-RILA[24]			APHH			PAPHH		
	AVG	STD	Rank	AVG	STD	Rank	AVG	STD	Rank	AVG	STD	Rank	AVG	STD	Rank
WFG1	0.14949	0.03189	4−	**0.25754**	0.05504	1+	0.13940	0.02318	5−	0.16174	0.03864	3−	0.21000	0.08026	2
WFG2	**0.79340**	0.05908	1+	0.75846	0.00986	5−	0.75878	0.00917	4−	0.77855	0.04823	2+	0.76662	0.01151	3
WFG3	**0.39394**	0.04468	1+	0.37030	0.04256	5−	0.38309	0.05146	3+	0.38496	0.02754	2+	0.38077	0.05472	4
WFG4	0.29298	0.02100	3−	0.27418	0.02339	5−	**0.36748**	0.04551	1+	0.27829	0.02766	4−	0.31734	0.07077	2
WFG5	0.32465	0.02378	3−	0.30986	0.06719	4−	0.32721	0.02647	2≈	0.30286	0.02039	5−	**0.32727**	0.06612	1
WFG6	0.31747	0.01209	5−	0.34410	0.02468	3−	0.35642	0.03506	2−	0.32929	0.01603	4−	**0.36745**	0.03758	1
WFG7	0.26325	0.01806	4−	**0.30518**	0.02056	1+	0.28915	0.02694	2+	0.24952	0.01855	5−	0.28482	0.04123	3
WFG8	0.25073	0.02049	4−	0.22111	0.04584	5−	**0.27017**	0.02228	1+	0.26014	0.01814	3−	0.26741	0.06471	2
WFG9	0.37751	0.01287	3−	0.37361	0.01843	4−	**0.41779**	0.00835	1+	0.36398	0.01464	5−	0.38895	0.01453	2
Total			28			33			21			33			20
Final Rank			3			4			2			4			1

best performer according to the Hypervolume metric on 2 test problems (WFG2-3). According to the final rank, PAPHH ranks first in both Spacing metric and Hypervolume metric on 5-objective WFG test problems.

5. Performance comparison with different types of meta-heuristics

In this section, we compare PAPHH with NSGA-III [29], IBEA [16], MOEA/D [15], and SMPSO [36] across 2-objective ZDT, 3-objective DTLZ, and 5-objective WFG test suites by assessing Spacing metric values and Hypervolume metric values obtained from 30 independent trials. The four comparative algorithms are selected because they are from different types of meta-heuristics for MOPs. NSGA-III, IBEA, MOEA/D, and SMPSO are Pareto-based MOEA, Indicator-based MOEA, Decomposition-based MOEA and MOPSO, respectively.

All the parameters in PAPHH and four comparative algorithms are set the same as in Section 4.1. Besides, the number of divisions in NSGA-III are set to 96, 12, and 6 for 2-, 3-, and 5-objective MOPs, respectively. The

Table 5
Test results of low-level heuristics and PAPHH on 2-objective ZDT, 3-objective DTLZ, and 5-objective WFG test problems respect to Spacing metric and Hypervolume metric from 30 trials. The best average value on each test problem is highlight in bold.

Pro.	Obj.	**Spacing metric value**														
		NSGA-III[29]			IBEA[16]			MOEA/D[15]			SMPSO[36]			PAPHH		
		AVG	STD	Rank	AVG	STD	Rank	AVG	STD	Rank	AVG	STD	Rank	AVG	STD	Rank
ZDT1	2	0.01077	6.49E−04	5−	0.00633	1.29E−03	4−	0.00495	4.13E−04	3−	**0.00111**	1.88E−04	1+	0.00361	5.40E−04	2
ZDT2	2	0.00559	3.44E−04	4−	0.00983	2.48E−03	5−	0.00442	2.06E−04	3−	**0.00114**	3.21E−04	1+	0.00320	4.55E−04	2
ZDT3	2	0.01304	8.48E−04	4−	0.01039	8.83E−04	3−	0.01713	6.65E−04	5−	**0.00288**	2.89E−04	1+	0.00415	6.95E−04	2
ZDT4	2	0.01067	6.32E−04	4−	0.01449	9.16E−03	5−	0.00515	6.17E−04	2+	**0.00144**	2.62E−04	1+	0.00785	6.92E−03	3
ZDT6	2	0.00479	3.42E−04	3−	0.00747	8.69E−04	4−	**0.00282**	1.82E−05	1+	0.04682	1.04E−01	5−	0.00424	1.93E−03	2
DTLZ1	3	0.05202	2.62E−02	3+	**0.04299**	5.07E−02	1+	0.04884	5.61E−03	2≈	0.06368	2.15E−02	5≈	0.05662	1.98E−02	4
DTLZ2	3	0.13175	3.08E−02	2−	0.18411	1.50E−02	5−	0.13685	3.22E−02	3−	0.14245	3.30E−02	4−	**0.11208**	2.08E−02	1
DTLZ3	3	0.33745	3.83E−01	3≈	**0.02095**	7.85E−02	1+	1.45598	2.45E+00	5≈	0.16571	8.83E−02	2≈	0.44591	5.96E−01	4
DTLZ4	3	**0.08033**	5.85E−02	1≈	0.10796	9.23E−02	3≈	0.11117	3.23E−02	4−	0.14333	3.42E−02	5−	0.08659	3.13E−02	2
DTLZ5	3	0.01638	3.27E−03	4−	0.01284	2.66E−03	3−	0.29320	2.77E−02	5−	**0.00357**	1.33E−03	1+	0.00484	1.46E−03	2
DTLZ6	3	0.06098	5.34E−02	4−	0.03052	2.12E−02	3−	0.28443	3.54E−02	5−	**0.00361**	7.90E−04	1+	0.01864	1.08E−02	2
DTLZ7	3	0.18967	4.69E−02	5−	**0.12091**	9.87E−02	1+	0.17046	4.46E−02	3−	0.18124	3.17E−02	4≈	0.16337	2.18E−02	2
WFG1	5	0.09133	2.95E−02	3−	**0.03252**	3.63E−03	1+	0.40125	2.52E−02	4−	0.92587	1.50E−01	5−	0.05200	2.07E−02	2
WFG2	5	0.33328	1.21E−01	2+	**0.21706**	4.55E−02	1+	0.38897	5.68E−02	3≈	0.66822	6.52E−02	5−	0.43427	9.17E−02	4
WFG3	5	0.51837	9.61E−02	3−	**0.00789**	2.74E−03	1+	0.71722	6.67E−02	5−	0.69755	3.91E−02	4−	0.44765	7.35E−02	2
WFG4	5	1.13700	7.70E−02	3−	**0.17884**	3.03E−01	1+	1.34868	9.37E−02	5−	1.25050	8.68E−02	4−	0.81510	1.80E−01	2
WFG5	5	1.20424	9.02E−02	3−	1.42709	6.75E−02	5−	1.22564	7.28E−02	4−	1.05801	8.73E−02	2−	**0.86603**	1.01E−01	1
WFG6	5	1.25480	7.67E−02	3−	1.45581	1.09E−01	4−	1.05183	9.09E−02	2≈	1.45713	9.10E−02	5−	**0.97824**	2.30E−01	1
WFG7	5	1.31105	9.00E−02	4−	1.23669	1.45E−01	2−	1.51465	1.03E−01	5−	1.26974	8.33E−02	3−	**0.90018**	1.36E−01	1
WFG8	5	1.30978	1.59E−01	5−	1.02908	2.28E−01	2≈	1.25266	1.16E−01	3−	1.26144	9.04E−02	4−	**0.95400**	9.36E−02	1
WFG9	5	1.14721	7.79E−02	3−	1.35428	9.56E−02	5−	1.27909	8.62E−02	4−	1.14089	7.14E−02	2−	**0.96754**	1.24E−01	1
Total				71			60			76			65			43
Final Rank				4			2			5			3			1

Pro.	Obj.	**Hypervolume metric value**														
		NSGA-III[29]			IBEA[16]			MOEA/D[15]			SMPSO[36]			PAPHH		
		AVG	STD	Rank	AVG	STD	Rank	AVG	STD	Rank	AVG	STD	Rank	AVG	STD	Rank
ZDT1	2	0.66142	9.32E−05	3−	0.66128	3.01E−04	4−	0.64810	2.70E−03	5−	**0.66198**	3.84E−05	1+	0.66164	1.31E−04	2
ZDT2	2	0.32833	3.03E−05	3−	0.32804	7.40E−05	4−	0.31289	4.31E−03	5−	**0.32870**	3.88E−05	1+	0.32844	2.30E−04	2
ZDT3	2	0.51473	1.55E−04	3−	0.51261	8.42E−04	4−	0.50112	4.06E−03	5−	**0.51570**	5.59E−05	1+	0.51542	1.66E−04	2
ZDT4	2	0.66021	8.72E−04	2−	0.24424	8.39E−02	5−	0.65525	3.83E−03	4−	**0.66177**	6.80E−05	1+	0.65915	1.44E−03	3
ZDT6	2	0.40024	2.51E−04	4≈	0.39747	3.56E−04	5−	0.40140	1.54E−05	2+	**0.40140**	4.83E−05	1+	0.40026	6.44E−04	3
DTLZ1	3	**0.77940**	3.79E−03	1+	0.16984	4.83E−02	5−	0.75086	1.78E−02	3−	0.73655	5.96E−03	4−	0.76337	6.03E−03	2
DTLZ2	3	0.41307	4.29E−04	2+	**0.42147**	3.72E−04	1+	0.35412	6.04E−03	5−	0.35605	5.41E−03	4−	0.37881	1.40E−02	3
DTLZ3	3	0.30477	1.05E−01	2+	0.00000	0.00E+00	5−	0.12974	1.35E−01	4−	**0.31383**	8.76E−02	1+	0.27435	8.58E−02	3
DTLZ4	3	0.35426	1.07E−01	3−	0.32220	1.08E−01	5−	0.33847	1.49E−02	4−	0.36392	5.75E−03	2−	**0.36631**	4.23E−02	1
DTLZ5	3	0.08694	1.12E−03	4−	0.09244	1.01E−04	3−	0.08581	2.23E−04	5−	**0.09332**	5.06E−05	1+	0.09259	3.74E−04	2
DTLZ6	3	0.06852	1.26E−02	5−	0.07280	1.01E−02	4−	0.08633	1.52E−04	3−	**0.09342**	2.01E−05	1+	0.08684	9.59E−03	2
DTLZ7	3	**0.29365**	8.13E−03	1+	0.23427	5.77E−02	4≈	0.15127	3.00E−02	5−	0.27629	4.46E−03	3−	0.28386	1.05E−02	2
WFG1	5	0.30453	4.25E−02	2+	**0.48589**	3.78E−02	1+	0.23108	4.20E−03	4−	0.25356	9.61E−04	3≈	0.21000	8.46E−02	5
WFG2	5	0.75867	1.42E−02	5≈	0.76301	7.98E−03	4≈	0.76485	3.90E−02	3≈	**0.82539**	1.53E−02	1+	0.76662	1.21E−02	2
WFG3	5	**0.55831**	1.15E−02	1+	0.10891	1.06E−02	5−	0.40806	1.18E−02	3+	0.50589	5.60E−03	2+	0.38077	5.77E−02	4
WFG4	5	**0.45381**	4.85E−02	1+	0.22081	4.65E−02	5−	0.23114	1.53E−02	4−	0.26631	1.67E−02	3≈	0.31734	7.46E−02	2
WFG5	5	0.51154	7.17E−03	2+	**0.61408**	2.11E−02	1+	0.33195	1.69E−02	3−	0.23346	7.91E−03	5−	0.32727	6.97E−02	4
WFG6	5	0.51530	1.27E−02	2+	**0.62732**	2.65E−02	1+	0.20836	2.52E−02	5−	0.35026	1.50E−02	4≈	0.36745	3.96E−02	3
WFG7	5	0.48447	1.31E−02	2+	**0.53796**	5.31E−02	1+	0.22491	1.13E−02	5−	0.24304	1.10E−02	4−	0.28482	4.35E−02	3
WFG8	5	0.38723	1.06E−02	2+	**0.40309**	5.09E−02	1+	0.12890	1.45E−02	5−	0.18846	1.22E−02	4−	0.26741	6.82E−02	3
WFG9	5	0.45048	3.20E−02	2+	**0.54298**	3.82E−02	1+	0.29602	2.07E−02	4−	0.25788	1.99E−02	5−	0.38895	1.53E−02	3
Total				52			69			86			52			56
Final Rank				1			4			5			1			3

archive size in IBEA is set to the population size N. In MOEA/D, the neighborhood selection probability is set to 0.9, and the neighbor size is set to 20. The penalty-based boundary intersection method is used in MOEA/D as in the literature [15]. In SMPSO, the neighborhood selection probability is set to 0.9, and the archive size and swarm population size are set to N.

The detailed results and Wilcoxon's rank sum test values at a 0.05 significance level are shown in Table 5. According to the Spacing metric values, PAPHH performs the best on 6 out of 21 test problems (DTLZ2 and WFG5-9), and the second best on 11 test problems (ZDT1-3, ZDT6, DTLZ4-7, WFG1, and WFG3-4). For a pair comparison, PAPHH is better than NSGA-III on 17 test problems except for DTLZ1, DTLZ3-4, and WFG2, while performing better than MOEA/D on 17 test problems except for ZDT4, ZDT6, DTLZ1, and WFG2. PAPHH performs better than IBEA and SMPSO on 14 out of 21 test problems (ZDT1-6, DTLZ2, DTLZ4-6, and WFG5-9 for IBEA, while ZDT6, DTLZ1-2, DTLZ4, DTLZ7, and WFG1-9 for SMPSO). Although PAPHH is not always among the top performing algorithms for all the test problems by comparing the four comparative meta-heuristics, its overall Spacing performance is still the best based on the final rank.

On the Hypervolume metric, PAPHH outperforms the other comparative meta-heuristics on one out of 21 test problems (DTLZ4), and performs the second best on 9 test problems. The overall performance of PAPHH is not outstanding, ranking third according to the final rank. In the worst case, PAPHH is even the worst algorithm on WFG1.

The performance of PAPHH on Hypervolume may be explained as follows. The evolutionary process in PAPHH is based on the Pareto-dominance mechanism, which is nearly the same as that used in NSGA-II in the experiments. Pareto-based algorithms sometimes have difficulties in offering selection pressures to push the solution set towards the Pareto-optimal front [2]. Thus, inferior converged performance will result in low hypervolume values, and even affect the diversity performance of algorithms.

6. *PAPHH with meta-heuristics as low-level heuristics*

In this section, we are interested in the following research question, i.e., could we combine multiple mechanisms (not only Pareto-dominance mechanism) into PAPHH to improve its ability to approach the Pareto-optimal front, meanwhile helping to get a well distributed solution set for multi- or many-objective optimization problems? One solution is to incorporate different types of meta-heuristics, such as Pareto-based MOEAs, Indicator-based MOEAs, Decomposition-based MOEAs and MOPSOs, into PAPHH as low-level heuristics.

In this section, we choose four meta-heuristics (denoted as $H = \{h_1, h_2, h_3, h_4\}$) from different categories, where h_1: NSGA-III, h_2: IBEA, h_3: MOEA/D, and h_4: SMPSO.

h_1: NSGA-III is a new version of NSGA-II. NSGA-III uses non-dominated sorting to rank solution set in the evolving process and adopts reference points for diversity control instead of crowding distance in NSGA-II.

h_2: IBEA is an indicator-based MOEA which defines an arbitrary indicator and directly uses this indicator in the selection process of the evolving stages. IBEA does not need any extra diversity preservation techniques.

h_3: MOEA/D is a decomposition-based MOEA. MOEA/D converts MOPs into a set of single-objective sub-problems. The neighborhood relationships among sub-problems are used to evolve solution set, and a number of given weight vectors are applied to control the diversity.

h_4: SMPSO is a particle swarm optimization algorithm utilizing Pareto-dominance concept to evolve. An external archive is used and the crowding distance from NSGA-II is adopted to control diversity.

In order to incorporate NSGA-III, IBEA, MOEA/D, and SMPSO into PAPHH, we slightly modify the framework of PAPHH as follows.

Table 6
Comparison of test results of low-level heuristics and PAPHH on MaOP1-MaOP6 with 3, 5 and 10 objectives using Hypervolume from 30 trials. The best average value on each test problem is highlight in bold.

Pro.	Obj.	**Hypervolume metric value**														
		NSGA-III[29]			IBEA[16]			MOEA/D[15]			SMPSO[36]			PAPHH		
		AVG	STD	Rank	AVG	STD	Rank	AVG	STD	Rank	AVG	STD	Rank	AVG	STD	Rank
MaOP1	3	0.21373	1.66E−03	3−	**0.22924**	4.69E−04	1≈	0.05805	3.82E−06	5−	0.21206	2.93E−03	4−	0.22923	4.94E−04	2
	5	0.01101	3.31E−04	3−	0.01517	1.17E−04	2−	0.00976	4.19E−04	5−	0.01044	2.82E−04	4−	**0.19064**	3.92E−02	1
	10	0.00000	2.86E−08	3−	0.00000	4.39E−08	2−	0.00000	5.59E−10	5−	0.00000	4.22E−07	4−	**0.03109**	2.83E−03	1
MaOP2	3	0.88355	1.31E−04	2+	0.86213	8.97E−03	4≈	**0.88597**	1.25E−04	1+	0.85979	6.44E−03	5≈	0.86329	9.11E−03	3
	5	0.99277	1.28E−04	2+	0.94596	8.74E−03	5≈	**0.99315**	2.32E−04	1+	0.97718	1.87E−03	4≈	0.97967	2.48E−02	3
	10	0.99993	1.53E−04	2+	0.99504	1.26E−03	4−	0.97151	1.84E−03	5−	**0.99993**	2.33E−05	1+	0.99943	2.54E−03	3
MaOP3	3	0.41385	4.94E−06	2−	0.40997	2.22E−03	4−	0.41385	5.14E−07	3−	0.37669	8.41E−03	5−	**0.41951**	2.89E−04	1
	5	0.69804	3.80E−04	3−	0.69504	1.34E−03	4−	0.69827	4.03E−04	2−	0.56710	5.03E−03	5−	**1.00000**	5.01e−21	1
	10	0.85210	6.62E−02	4−	0.93242	8.77E−04	2−	0.92202	3.00E−04	3−	0.72467	1.07E−02	5−	**1.00000**	9.83E−14	1
MaOP4	3	0.41385	4.26E−06	2−	0.40800	1.01E−03	4−	0.41385	1.17E−06	3−	0.37974	4.61E−03	5−	**0.41948**	4.48E−04	1
	5	0.69824	4.13E−04	3−	0.69508	2.21E−03	4−	0.69870	5.60E−04	2−	0.57393	7.83E−03	5−	**1.00000**	1.25e−10	1
	10	0.90182	4.51E−02	4−	0.93202	7.85E−04	2−	0.92184	1.72E−04	3−	0.72540	1.45E−02	5−	**1.00000**	4.74E−13	1
MaOP5	3	0.30731	7.48E−04	3−	**0.32109**	1.57E−03	1+	0.28513	5.11E−05	5−	0.29530	5.55E−03	4−	0.31920	4.83E−04	2
	5	0.00162	9.04E−05	3−	0.00116	7.37E−05	4−	0.00005	1.16E−06	5−	0.00167	6.05E−05	2−	**0.30480**	1.46E−02	1
	10	0.00000	6.58E−10	3−	0.00000	2.12E−10	5−	0.00000	1.19E−09	4−	0.00000	7.36E−10	2−	**0.04749**	1.03E−02	1
MaOP6	3	0.77150	4.85E−04	3−	0.77438	4.26E−04	2≈	0.73520	2.28E−03	5−	0.75142	3.51E−03	4−	**0.77446**	3.59E−04	1
	5	0.05871	5.85E−04	3−	0.06146	2.01E−05	2−	0.05669	1.21E−04	4−	0.03900	5.66E−03	5−	**0.49729**	1.39E−01	1
	10	0.00132	4.66E−05	3−	0.00128	8.56E−05	5−	0.00143	2.43E−05	2−	0.00131	9.81E−05	4−	**0.13048**	3.17E−02	1
Total				51			57			63			73			26
Final Rank				2			3			4			5			1

Table 7
Comparison of test results of meta-heuristics and PAPHH on MaOP1-MaOP6 with 3, 5, and 10 objectives using Hypervolume from 30 trials. The variance is in brackets bellow. The best average value on each test problem is highlight in bold.

Pro.	Obj.	**Hypervolume metric value**															
		LMEA[65]		MaOEA/IGD[34]		NMPSO[2]		PPS[66]		WOF-SMPSO[37]		AR-MOEA[27]		CMOPSO[38]		PAPHH	
		AVG	Rank	AVG	Rank	AVG	Rank	AVG	Rank	AVG	Rank	AVG	Rank	AVG	Rank	AVG	Rank
MaOP1	3	0.12841	4−	7.57e−07	8−	0.12535	5−	0.11632	7−	0.12207	6−	0.15485	3−	0.17745	2−	**0.22923**	**1**
		(5.33e−4)		(5.00e−7)		(1.21e−3)		(1.87e−3)		(1.81e−3)		(5.39e−2)		(4.84e−2)		(4.94e−4)	
	5	0.00143	4−	1.09e−14	8−	0.00025	7−	0.00137	6−	0.00143	4−	0.00448	3−	0.00768	2−	**0.19064**	**1**
		(7.74e−5)		(1.82e−14)		(2.30e−4)		(1.34e−4)		(1.10e−4)		(3.94e−3)		(6.09e−3)		(3.92e−2)	
	10	2.77E−26	4−	3.04E−44	5−	2.35E−17	3−	0.00E+00	6−	0.00E+00	6−	1.07E−07	2−	0.00E+00	6−	**0.03109**	**1**
		(1.07e−25)		(1.04e−43)		(8.52e−17)		(0.00e+0)		(0.00e+0)		(2.06e−7)		(9.15e−7)		(2.83e−3)	
MaOP2	3	0.94203	3+	0.70407	8−	0.88670	6+	0.94168	4+	0.93439	5+	0.95211	2+	**0.95674**	**1+**	0.86329	7
		(1.20e−3)		(5.87e−3)		(2.66e−2)		(2.17e−3)		(7.29e−3)		(7.14e−3)		(6.42e−3)		(9.11e−3)	
	5	0.99548	4+	0.94656	8−	0.97572	7−	0.99568	3+	0.99056	5+	0.99773	2+	**0.99834**	**1+**	0.97967	6
		(1.80e−3)		(1.04e−2)		(7.65e−3)		(3.74e−4)		(2.31e−3)		(5.96e−4)		(3.79e−4)		(2.48e−2)	
	10	0.99823	6−	0.93778	8−	0.98282	7−	0.99996	3+	0.99984	4+	**1.00000**	**1+**	0.99999	2+	0.99943	5
		(1.54e−3)		(1.69e−2)		(1.22e−2)		(1.46e−5)		(3.70e−5)		(2.58e−6)		(7.23e−6)		(2.54e−2)	
MaOP3	3	0.41140	5−	0.39191	6−	0.41511	4−	0.38760	7−	0.37860	8−	**0.49159**	1≈	0.49140	2≈	0.41951	3
		(1.99e−3)		(1.81e−4)		(1.46e−3)		(7.36e−3)		(6.70e−3)		(7.53e−2)		(7.46e−2)		(2.84e−4)	
	5	0.68390	6−	0.69828	4−	0.69823	5−	0.65768	7−	0.56896	8−	0.75922	2−	0.74760	3−	**1.00000**	**1**
		(2.37e−3)		(4.13e−4)		(2.16e−3)		(7.56e−3)		(1.06e−2)		(5.90e−2)		(6.13e−2)		(5.01e−21)	
	10	0.86721	6−	0.92259	3−	0.91681	4−	0.80784	7−	0.70823	8−	0.94817	2−	0.88105	5−	**1.00000**	**1**
		(6.29e−2)		(3.47e−4)		(1.96e−3)		(1.03e−2)		(7.36e−3)		(2.45e−2)		(3.75e−2)		(9.83e−14)	
MaOP4	3	0.41153	5−	0.39189	6−	0.41611	4−	0.38842	7−	0.38020	8−	0.49159	2≈	**0.49168**	1≈	0.41948	3
		(2.06e−3)		(1.27e−4)		(8.32e−4)		(6.03e−3)		(3.67e−3)		(7.53e−2)		(7.53e−2)		(4.48e−4)	
	5	0.68381	6−	0.69822	5−	0.69907	4−	0.65844	7−	0.56783	8-	0.75943	2−	0.74583	3−	**1.00000**	**1**
		(2.41e−3)		(4.82e−4)		(3.10e−3)		(5.88e−3)		(9.12e−3)		(5.91e−2)		(6.00e−2)		(1.25e−10)	
	10	0.88555	5−	0.92263	3−	0.91657	4−	0.80309	7−	0.71041	8−	0.94810	2−	0.87631	6−	**1.00000**	**1**
		(1.16e−2)		(2.71e−4)		(3.96e−3)		(1.19e−2)		(1.10e−2)		(2.46e−2)		(3.91e−2)		(4.74e−13)	
MaOP5	3	0.31492	5−	0.26940	8−	0.31524	4−	0.30119	6−	0.29434	7−	0.40194	2≈	**0.40627**	1≈	0.31920	3
		(1.17e−3)		(5.73e−3)		(2.27e−3)		(4.29e−3)		(3.80e−3)		(8.86e−2)		(8.73e−2)		(4.83e−4)	
	5	0.00164	4−	0.00077	7−	0.00E+00	8−	0.00159	6−	0.00164	4−	0.01824	3−	0.01850	2−	**0.30480**	**1**
		(5.11e−5)		(8.70e−6)		(0.00e+0)		(1.03e−4)		(7.71e−5)		(1.63e−2)		(1.65e−2)		(1.46e−2)	
	10	2.07E−09	5−	1.94E−19	7−	0.00E+00	8−	2.66E−09	4−	1.78E−09	6−	1.30E−05	3−	1.33E−05	2−	**0.04749**	**1**
		(3.71e−10)		(8.64e−20)		(0.00e+0)		(2.48e−10)		(5.51e−10)		(1.26e−5)		(1.29e−5)		(1.03e−2)	
MaOP6	3	0.77042	5−	0.77393	4−	0.74961	7−	0.74966	6−	0.74748	8−	0.80272	2≈	**0.80395**	1≈	0.77447	3
		(1.08e−3)		(6.94e−5)		(7.49e−3)		(3.92e−3)		(5.70e−3)		(2.92e−2)		(2.91e−2)		(3.59e−4)	
	5	0.16274	5−	0.15048	7−	0.14437	8−	0.15727	6−	0.19184	3−	0.19253	2−	0.19082	4−	**0.49729**	**1**
		(4.52e−3)		(1.65e−4)		(4.58e−3)		(1.61e−3)		(3.45e−2)		(3.67e−2)		(3.65e−2)		(1.39e−1)	
	10	0.00121	2−	0.00077	8−	0.00119	3−	0.00106	5−	0.00108	4−	0.00094	6−	0.00089	7−	**0.13048**	**1**
		(5.86e−4)		(3.52e−4)		(5.07e−4)		(5.16e−4)		(4.87e−4)		(5.64e−4)		(5.48e−4)		(3.17e−2)	
Total		84		113		98		104		110		42		51		41	
Final Rank		4		8		5		6		7		2		3		1	

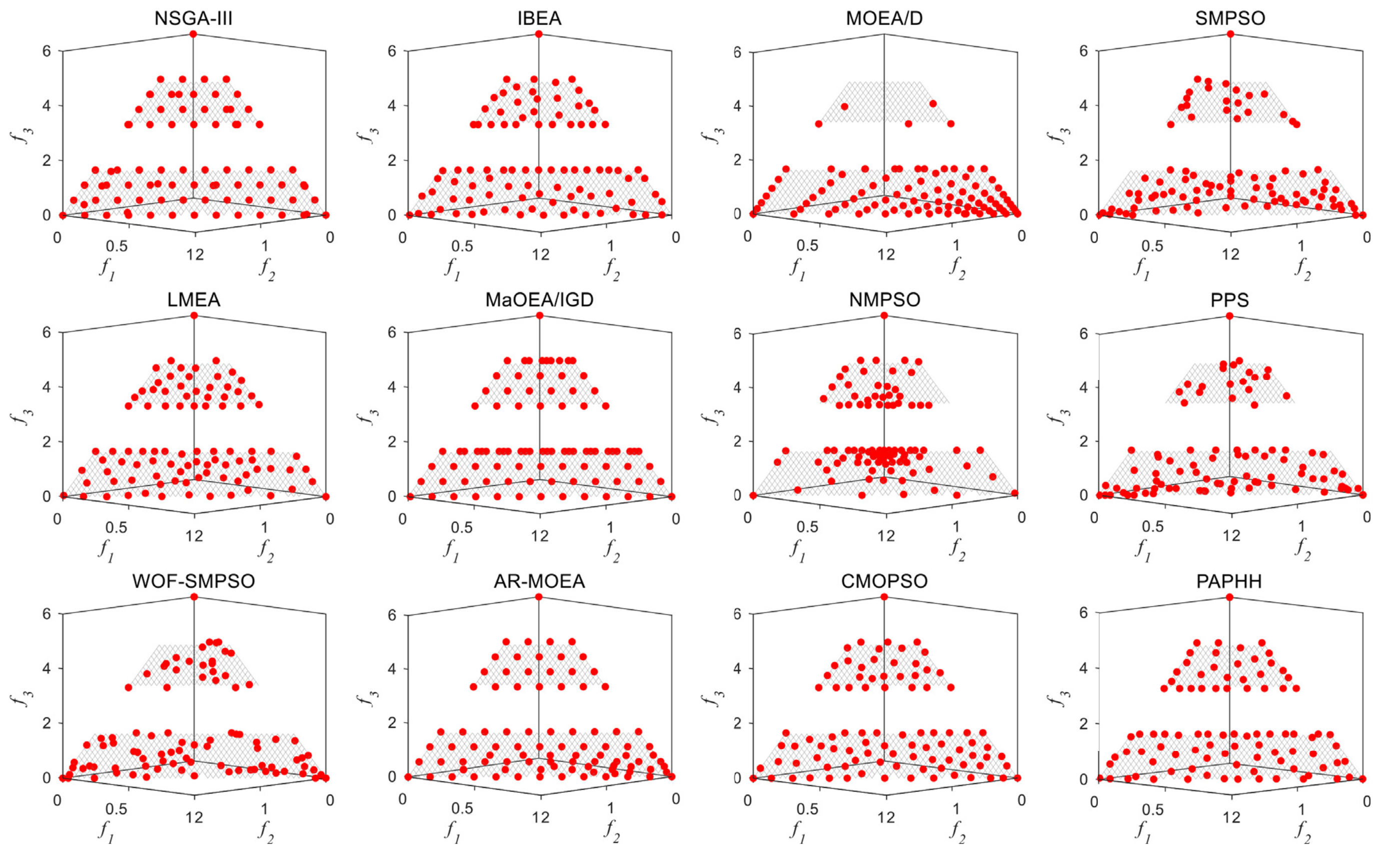

Fig. 11. Plots of the final approximation sets with the best Hypervolume metric values found by comparing algorithms and PAPHH from 30 trials in the objective space on MaOP6.

i) Because each low-level heuristic has its own mechanisms to evolve population and control diversity, we remove the evolutionary process (Steps 3,4, and 9) from Algorithm 1.
ii) Every selected low-level heuristic in PAPHH is run for several (e.g. 10) consecutive iterations.
iii) Hypervolume metric is employed as the fitness function instead of Spacing metric, for Spacing metric is not suitable for evaluating problems with high-dimensional objectives. Besides, Hypervolume is a maximization metric, we modify the reward value function in formula (7) as $r_{a,t} = 1 + (f_{a,t} - f_{best})/f_{best}$ if $f_{a,t} \geq f_{best}$, otherwise, $r_{a,t} = 0$.

6.1. Experimental settings

In our experiments, the following settings are considered.

Test problems: A novel test suite MaOP with more challenging features, such as multimodality, complicated Pareto-optimal set, bias, degeneracy and disconnection are used in the experiments [64]. Six test problems (MaOP1-6) with 3, 5, and 10 objectives are chosen. The parameter values in MaOP are the same with [64].

Comparing algorithms: Eleven comparing algorithms are chosen for comparison. The first four are the low-level heuristics run on their own, that is NSGA-III[29], IBEA [16], MOEA/D [15], and SMPSO [36]. The four low-level heuristics are used to test the effectiveness of PAPHH in combining low-level heuristics. The rest seven are LMEA[65], MaOEA/IGD [34], NMPSO [2], PPS [66], WOF-SMPSO [37], AR-MOEA [27], and CMOPSO [38]. They are recently proposed and competitive meta-heuristics.

Parameter settings: PAPHH needs to incorporate the four low-level heuristics together. Due to the reference points in NSGA-III and weight vectors in MOEA/D, the population size in all the algorithms is set to $N =$ 92, 212, and 276 for problems with 3-, 5-, and 10-objectives, respectively, as in Ref. [29]. The max iteration is set to 500. If needed, the crossover operator and mutation operator in all the algorithms use SBX crossover operator and polynomial mutation operator, s.t. crossover probability $pc = 0.9$, distribution index for SBX is 20, mutation probability $pm = 1/n$, distribution index for polynomial mutation is 20. The number of divisions in NSGA-III are set to 12 and 6 for 3- and 5-objective MOPs, respectively. Two layers of divisions are used for 10-objective MOPs. On the boundary layer, the number of divisions is set to 3, while on the inside layer, the number of divisions is set to 2 as [29]. The archive size in IBEA is set to N. In MOEA/D, the neighborhood selection probability and the neighbor size are set to 0.9 and 20, respectively. In SMPSO, the neighborhood selection probability is set to 0.9. The archive size and swarm population size of SMPSO are both set to N. The parameter K in PAPHH is set to 10. The parameter values of other comparative algorithms are the same as their original papers.

6.2. Experimental results of comparing meta-heuristics and PAPHH

In this subsection, we first run the low-level heuristics (NSGA-III, IBEA, MOEA/D, and SMPSO) as standalone algorithms, and compare them with PAPHH using Hypervolume values averaged from 30 trials (as shown in Table 6). Next, to test the effectiveness of PAPHH against newly proposed and effective meta-heuristics of different types, PAPHH is compared with LMEA, MaOEA/IGD, NMPSO, PPS, WOF-SMPSO, AR-MOEA, and CMOPSO (as shown in Table 7). Wilcoxon's rank sum test at a 0.05 significance level is adopted here to conduct pair comparison.

The test results in Table 6 can be concluded as follows. PAPHH performs the best on 13 out of 18 test problems, except for MaOP1 and MaOP5 with 3 objectives (performs the second best) and MaOP2 with 3, 5, and 10 objectives (performs the third best). For pair comparison, PAPHH performs better than NSGA-III on 15 test problems except for MaOP2 with 3, 5, and 10 objectives. PAPHH outperforms IBEA and MOEA/D on 16 test problems (except for MaOP1 and MaOP5 with 3 objectives for IBEA, while except for MaOP2 with 3 and 5 objectives for MOEA/D). PAPHH is superior to SMPSO on 17 test problems except MaOP2 with 10 objectives. It's also interesting to note that, after combining NSGA-III, IBEA, MOEA/D and SMPSO into the PAPHH framework, PAPHH is always better than each low-level heuristic no matter on problems with low-dimensional objectives or high-dimensional objectives. Over all the test problems, PAPHH never performs the worst among the algorithms.

According to Table 7, PAPHH performs the best on 11 out of 18 test problems comparing with the other seven newly proposed meta-heuristics, including one 3-objective test problem (MaOP1), five 5- and 10-objective test problems (MaOP1 and MaOP3-6). For pair comparison, PAPHH is better than MaOEA/IGD on all the test problems, and is superior to NMPSO except for 3-objective MaOP2. PAPHH is better than LMEA on 16 out of 18 test problems except for MaOP2 with 3 and 5 objectives, while PAPHH is superior to PPS and WOF-SMPSO on 15 test problems except for 3-, 5-, and 10-objective MaOP2. PAPHH excels AR-MOEA and CMOPSO on 11 test problems, including one 3-objective test problem (MaOP1), five 5- and 10-objective test problems (MaOP1, MaOP3-6). Interestingly, PAPHH becomes more competitive in high-dimensional objective MaOPs than low-dimensional ones when compared with the seven meta-heuristics. Meanwhile, based on the final rank, the overall performance of PAPHH is outstanding, ranking first. AR-MOEA and CMOPSO get the second and third place, respectively.

The plots of the final solution set of MaOP6 with the highest Hypervolume value obtained by the comparative algorithms and PAPHH from 30 runs are shown in Fig. 11. On one hand, MOEA/D loses part of Pareto-optimal front, and are prone to converge to some area (bottom and right in this figure). MaOEA/IGD prefers the top border front. SMPSO, NMPSO, PPS, and WOF-SMPSO obtain poorly distributed solution sets. On the other hand, the solution sets achieved by PAPHH and other comparative algorithms are distributed well.

7. Conclusion

In this study, we propose a new selection hyper-heuristic, PAPHH, to improve the performance of the obtained solution sets for MOPs. Firstly, PAPHH operates over three diversity mechanisms (the crowded-comparison strategy, the hyper-grid strategy and the density estimation strategy) as low-level heuristics. Secondly, PAPHH incorporates four classical meta-heuristics (NSGA-III, IBEA, MOEA/D, and SMPSO) as low-level heuristics. The framework of PAPHH is with the purpose to combine the merits of every low-level heuristic and get a well-distributed approximation set for the MOP at hand. At each decision point, one low-level heuristic is selected and applied.

Since choosing a suitable low-level heuristic at each decision point is a crucial part in a selection hyper-heuristic. We also propose a new learning strategy, the PAP strategy, to automatically learn the performance of each low-level heuristic in an online manner. The PAP strategy consists of two main components, the adaptive pursuit strategy and the perturbation strategy. PAPHH utilizes the PAP strategy to improve the decision-making process during the heuristic selection process with the reward value as the feedback after applying a low-level heuristic. In PAPHH, the reward value is measured as the fitness improvement rate.

Experimental results show strong empirical evidences that the performance of PAPHH has highly competitive performance in terms of Spacing metric and Hypervolume metric. The results also demonstrate that PAPHH can make claim for using on many-objective problems.

The PAPHH offers a unified high-level framework, in which other diversity mechanisms or meta-heuristics can also be easily included. For the future work, we are interested in investigating components in hyper-heuristics that affect both the diversity and the convergence aspects of approximation sets. Another promising future research topic is how to select the most suitable low-level heuristics. One of the possible and interesting way is to design new hyper-heuristic framework by incorporating preference information, such as priori or interactive approaches. For example, mining priori knowledge, and design data-driven selection hyper-heuristic is an interesting topic. Also, it would be promising to incorporate domain experts' knowledge to interactively and dynamically achieve better low-level heuristic selection mechanism.

Declaration of competing interest

None.

Acknowledgement

This work was supported by the National Natural Science Foundation of China [grant number 61772107]; and Technological Research Projects in Henan Province [grant number 172102210003, 192102210107].

References

[1] B.Y. Qu, Y.S. Zhu, Y.C. Jiao, M.Y. Wu, P.N. Suganthan, J.J. Liang, A survey on multi-objective evolutionary algorithms for the solution of the environmental/economic dispatch problems, Swarm Evol. Comput. 38 (2018) 1–11, https://doi.org/10.1016/j.swevo.2017.06.002.

[2] Q.Z. Lin, S.B. Liu, Q.L. Zhu, C.Y. Tang, R.Z. Song, J.Y. Chen, C.A. Coello, K.C. Wong, J. Zhang, Particle swarm optimization with a balanceable fitness estimation for many-objective optimization problems, IEEE Trans. Evol. Comput. 22 (1) (2018) 32–46, https://doi.org/10.1109/TEVC.2016.2631279.

[3] C.A. Coello, G.B. Lamont, D.A. Van Veldhuizen, Evolutionary Algorithms for Solving Multi-Objective Problems, second ed., Springer, Boston, MA, 2007 https://doi.org/10.1007/978-0-387-36797-2.

[4] N. Nedjah, L.M. Mourelle, Evolutionary multi-objective optimisation: a survey, Int. J. Bio-Inspired Comput. 7 (1) (2015) 1–25, https://doi.org/10.1504/IJBIC.2015.067991.

[5] J. Luo, Y. Yang, X. Li, Q. Liu, M. Chen, K. Gao, A decomposition-based multi-objective evolutionary algorithm with quality indicator, Swarm Evol. Comput. 39 (2018) 339–355, https://doi.org/10.1016/j.swevo.2017.11.004.

[6] J. Hajek, A. Szollos, J. Sistek, A new mechanism for maintaining diversity of Pareto archive in multi-objective optimization, Adv. Eng. Software 41 (7–8) (2010) 1031–1057, https://doi.org/10.1016/j.advengsoft.2010.03.003.

[7] H. Seada, M. Abouhawwash, K. Deb, Multi-phase balance of diversity and convergence in multiobjective optimization, IEEE Transactions on Evolutionary Computation (2018), https://doi.org/10.1109/TEVC.2018.2871362.

[8] K. Tahernezhad, K.B. Lari, A. Hamzeh, S. Hashemi, H.C.- Moea, A hierarchical clustering approach for increasing the solution's diversity in multiobjective evolutionary algorithms, Intell. Data Anal. 19 (1) (2015) 187–208, https://doi.org/10.3233/IDA-140703.

[9] E. Zitzler, L. Thiele, J. Bader, On set-based multiobjective optimization, IEEE Trans. Evol. Comput. 14 (1) (2010) 58–79, https://doi.org/10.1109/TEVC.2009.2016569.

[10] S.F. Adra, P.J. Fleming, Diversity management in evolutionary many-objective optimization, IEEE Trans. Evol. Comput. 15 (2) (2011) 183–195, https://doi.org/10.1109/TEVC.2010.2058117.

[11] A. García-Nájera, A. López-Jaimes, An investigation into many-objective optimization on combinatorial problems: analyzing the pickup and delivery problem, Swarm Evol. Comput. 38 (2018) 218–230, https://doi.org/10.1016/j.swevo.2017.08.001.

[12] D.W. Corne, N.R. Jerram, J.D. Knowles, M.J. Oates, Pesa-II: region-based selection in evolutionary multiobjective optimization, in: Proceedings of the 2001 Genetic and Evolutionary Computation Conference, San Francisco, California, USA, 2001, pp. 283–290, http://dl.acm.org/citation.cfm?id≈2955239.2955289.

[13] E. Zitzler, M. Laumanns, L. Thiele, SPEA2: improving the strength pareto evolutionary algorithm, in: Proceedings of Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems, Athens, Greece, 2001, pp. 95–100, https://doi.org/10.3929/ethz-a-004284029.

[14] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multi-objective genetic algorithm: NSGA-II, IEEE Trans. Evol. Comput. 6 (2) (2002) 182–197, https://doi.org/10.1109/4235.996017.

[15] Q. Zhang, H. Li, MOEA/D: a multiobjective evolutionary algorithm based on decomposition, IEEE Trans. Evol. Comput. 11 (6) (2007) 712–731, https://doi.org/10.1109/TEVC.2007.892759.

[16] E. Zitzler, S. Künzli, Indicator-based selection in multiobjective search, Lecture Notes in Computer Science vol. 3242, Springer, Berlin, Heidelberg, 2004, pp. 832–842, https://doi.org/10.1007/978-3-540-30217-9_84.

[17] V. Khare, X. Yao, K. Deb, Performance scaling of multi-objective evolutionary algorithms, in: Proceedings of Second International Conference on Evolutionary Multi-Criterion Optimization, Faro, Portugal, 2003, pp. 376–390, https://doi.org/10.1007/3-540-36970-8_27.

[18] L. Bradstreet, L. Barone, L. While, S. Huband, P. Hingston, Use of the wfg toolkit and pisa for comparison of MOEAs, in: Proceedings of IEEE Symposium on Computational Intelligence in Multi-Criteria Decision-Making, Honolulu, Hawaii, USA, 2007, pp. 382–389, https://doi.org/10.1109/MCDM.2007.369117.

[19] M. Gong, L. Jiao, H. Du, L. Bo, Multiobjective immune algorithm with nondominated neighbor-based selection, Evol. Comput. 16 (2) (2008) 225–255, https://doi.org/10.1162/evco.2008.16.2.225.

[20] E.K. Burke, M. Gendreau, M. Hyde, G. Kendall, G. Ochoa, E. Özcan, R. Qu, Hyper-heuristics: a survey of the state of the art, J. Oper. Res. Soc. 64 (12) (2013) 1695–1724, https://doi.org/10.1057/jors.2013.71.
[21] E.K. Burke, M.R. Hyde, G. Kendall, G. Ochoa, E. Özcan, J.R. Woodward, A classification of hyper-heuristic approaches: revisited, Handbook of Metaheuristics, vol. 272, 2018, pp. 453–477, https://doi.org/10.1007/978-3-319-91086-4_14.
[22] J. Li, G. Kendall, A hyper-heuristic methodology to generate adaptive strategies for games, IEEE Trans. Comput. Intell. AI Gam. 9 (1) (2017) 1–10, https://doi.org/10.1109/TCIAIG.2015.2394780.
[23] R.A. Gonçalves, J.N. Kuk, C.P. Almeida, S.M. Venske, MOEA/D-HH: a hyper-heuristic for multi-objective problems, in: International Conference on Evolutionary Multi-Criterion Optimization, Guimarães, Portugal, 2015, pp. 94–108, https://doi.org/10.1007/978-3-319-15934-8_7.
[24] W. Li, E. Özcan, R. John, A learning automata based multiobjective hyper-heuristic, IEEE Trans. Evol. Comput. 23 (1) (2019) 59–73, https://doi.org/10.1109/TEVC.2017.2785346.
[25] M. Maashi, E. Özcan, G. Kendall, A multi-objective hyper-heuristic based on choice function, Expert Syst. Appl. 41 (9) (2014) 4475–4493, https://doi.org/10.1016/j.eswa.2013.12.050.
[26] M. Maashi, G. Kendall, E. Özcan, Choice function based hyper-heuristics for multi-objective optimization, Appl. Soft Comput. 28 (2015) 312–326, https://doi.org/10.1016/j.asoc.2014.12.012.
[27] Y. Tian, R. Cheng, X.Y. Zhang, F. Cheng, Y.C. Jin, An indicator-based multiobjective evolutionary algorithm with reference point Adaptation for better versatility, IEEE Trans. Evol. Comput. 22 (4) (2018) 609–622, https://doi.org/10.1109/TEVC.2017.2749619.
[28] A. Trivedi, D. Srinivasan, K. Sanyal, A. Ghosh, A survey of multiobjective evolutionary algorithms based on decomposition, IEEE Trans. Evol. Comput. 21 (3) (2017) 440–462, https://doi.org/10.1109/TEVC.2016.2608507.
[29] K. Deb, H. Jain, An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, Part I: solving problems with box constraints, IEEE Trans. Evol. Comput. 18 (4) (2014) 577–601, https://doi.org/10.1109/TEVC.2013.2281535.
[30] H. Ishibuchi, T. Murata, A multi-objective genetic local search algorithm and its application to flowshop scheduling, IEEE Trans. Syst. Man Cybern. C Appl. Rev. 28 (3) (1998) 392–403, https://doi.org/10.1109/5326.704576.
[31] K. Li, K. Deb, Q.F. Zhang, S. Kwong, An evolutionary many-objective optimization algorithm based on dominance and decomposition, IEEE Trans. Evol. Comput. 19 (5) (2015) 694–716, https://doi.org/10.1109/TEVC.2014.2373386.
[32] E. Zitzler, L. Thiele, Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach, IEEE Trans. Evol. Comput. 3 (4) (1999) 257–271, https://doi.org/10.1109/4235.797969.
[33] R.H. Gómez, C.A. Coello, Improved metaheuristic based on the R2 indicator for many-objective optimization, in: GECCO '15 Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation, 2015, pp. 679–686, https://doi.org/10.1145/2739480.2754776.
[34] Y. Sun, G.G. Yen, Z. Yi, IGD indicator-based evolutionary algorithm for many-objective optimization problems, IEEE Trans. Evol. Comput. 23 (2) (2019) 173–187, https://doi.org/10.1109/tevc.2018.2791283.
[35] Q.Z. Lin, J.Q. Li, Z.H. Du, J.Y. Chen, Z. Ming, A novel multi-objective particle swarm optimization with multiple search strategies, Eur. J. Oper. Res. 247 (3) (2015) 732–744, https://doi.org/10.1016/j.ejor.2015.06.071.
[36] A.J. Nebro, J.J. Durillo, J. Garcia-Nieto, C.A. Coello, F. Luns, E. Alba, SMPSO: A New PSO-Based Metaheuristic for Multi-Objective Optimization, in: 2009 IEEE Symposium on Computational Intelligence in Multi-Criteria Decision-Making(MCDM), 2009, pp. 66–73, https://doi.org/10.1109/MCDM.2009.4938830.
[37] H. Zille, H. Ishibuchi, S. Mostaghim, Y. Nojima, A framework for large-scale multiobjective optimization based on problem transformation, IEEE Trans. Evol. Comput. 22 (2) (2018) 260–275, https://doi.org/10.1109/tevc.2017.2704782.
[38] X.Y. Zhang, X.T. Zheng, R. Cheng, J.F. Qiu, Y.C. Jin, A competitive mechanism based multi-objective particle swarm optimizer with fast convergence, Inf. Sci. 427 (2018) 63–76, https://doi.org/10.1016/j.ins.2017.10.037.
[39] H. Ishibuchi, Y. Setoguchi, H. Masuda, Y. Nojima, Performance of decomposition-based many-objective algorithms strongly depends on pareto front shapes, IEEE Trans. Evol. Comput. 21 (2) (2017) 169–190, https://doi.org/10.1109/TEVC.2016.2587749.
[40] M.Q. Li, S.X. Yang, X.H. Liu, Pareto or non-pareto: Bi-criterion evolution in multiobjective optimization, IEEE Trans. Evol. Comput. 20 (5) (2016) 645–665, https://doi.org/10.1109/TEVC.2015.2504730.
[41] S. Asta, E. Özcan, T. Curtois, A tensor based hyper-heuristic for nurse rostering, Knowl. Base Syst. 98 (15) (2016) 185–199, https://doi.org/10.1016/j.knosys.2016.01.031.
[42] T. Wauters, K. Verbeeck, P. De Causmaecker, G. Vanden Berghe, Boosting Metaheuristic Search Using Reinforcement Learning, Hybrid Metaheuristics, Springer, Berlin, Heidelberg, 2013, pp. 433–452, https://doi.org/10.1007/978-3-642-30671-6_17.
[43] E.K. Burke, J.D. Landa Silva, E. Soubeiga, multi-objective hyper-heuristic approaches for space allocation and timetabling, in: T. Ibaraki, K. Nonobe, M. Yagiura (Eds.), Metaheuristics: Progress as Real Problem Solvers, Springer, New York, 2005, pp. 129–158, https://doi.org/10.1007/0-387-25383-1_6.
[44] J.A. Vrugt, B.A. Robinson, Improved evolutionary optimization from genetically adaptive multimethod search, Proc. Natl. Acad. Sci. Unit. States Am. 104 (3) (2007) 708–711, https://doi.org/10.1073/pnas.0610471104.
[45] K. McClymont, E.C. Keedwell, Markov chain hyper-heuristic (mchh): an online selective hyper-heuristic for multi-objective continuous problems, in: Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation, Dublin, Ireland, 2011, pp. 2003–2010, https://doi.org/10.1145/2001576.2001845.
[46] A.C. Kumari, K. Srinivas, Scheduling and inspection planning in software development projects using multi-objective hyper-heuristic evolutionary algorithm, Int. J. Sci. Eng. Appl. 4 (3) (2013) 45–57, https://doi.org/10.5121/ijsea.2013.4304.
[47] Z. Ren, H. Jiang, J. Xuan, Z. Luo, Hyper-heuristics with low level parameter adaptation, Evol. Comput. 20 (2) (2012) 189–227, https://doi.org/10.1162/EVCO_a_00063.
[48] S.A.G. van der Stockt, A.P. Engelbrecht, Analysis of selection hyper-heuristics for population-based meta-heuristics in real-valued dynamic optimization, Swarm Evol. Comput. 43 (2018) 127–146, https://doi.org/10.1016/j.swevo.2018.03.012.
[49] P. Garrido, M.C. Riff, DVRP: a hard dynamic combinatorial optimisation problem tackled by an evolutionary hyper-heuristic, J. Heuristics 16 (6) (2010) 795–834, https://doi.org/10.1007/s10732-010-9126-2. Springer US.
[50] D. Thierens, U. University, T. Netherlands, An adaptive pursuit strategy for allocating operator probabilities, in: Proceedings of the 2005 Conference on Genetic and Evolutionary Computation, Washington, USA, 2005, pp. 1539–1546, https://doi.org/10.1145/1068009.1068251.
[51] Á. Fialho, L. Da Costa, M. Schoenauer, M. Sebag, Analyzing bandit-based adaptive operator selection mechanisms, Ann. Math. Artif. Intell. 60 (1–2) (2010) 25–64, https://doi.org/10.1007/s10472-010-9213-y.
[52] G. Guizzo, G.M. Fritsche, S.R. Vergilio, A.T.R. Pozo, A hyperheuristic for the multi-objective integration and test order problem, in: Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation, Madrid, Spain, 2015, pp. 1343–1350, https://doi.org/10.1145/2739480.2754725.
[53] N. Hitomi, D. Selva, A hyperheuristic approach to leveraging domain knowledge in multi-objective evolutionary algorithms, in: ASME 2016 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, American Society of Mechanical Engineers, 2016, https://doi.org/10.1115/DETC2016-59870. V02BT03A030–V02BT03A030.
[54] K.C. Tan, T. H Lee, E.F. Khor, Evolutionary algorithms for multi-objective optimization: performance assessments and comparisons, Artif. Intell. Rev. 17 (4) (2002) 253–290, https://doi.org/10.1023/A:1015516501242.
[55] J.R. Schott, Fault Tolerant Design Using Single and Multi-Criteria Genetic Algorithm Optimization, Master's Thesis, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, USA, 1995.
[56] S. Jiang, S. Yang, Evolutionary dynamic multiobjective optimization: benchmarks and algorithm comparisons, IEEE Trans. Cyber. 47 (1) (2017) 198–211, https://doi.org/10.1109/TCYB.2015.2510698.
[57] K. Li, Á. Fialho, S. Kwong, Q. Zhang, Adaptive operator selection with bandits for a multiobjective evolutionary algorithm based on decomposition, IEEE Trans. Evol. Comput. 18 (1) (2014) 114–130, https://doi.org/10.1109/TEVC.2013.2239648.
[58] E. Zitzler, K. Deb, L. Thiele, Comparison of multiobjective evolutionary algorithms: empirical results, Evol. Comput. 8 (2) (2000) 173–195, https://doi.org/10.1162/106365600568202.
[59] K. Deb, Multi-objective genetic algorithms: problem difficulties and construction of test problems, Evol. Comput. 7 (3) (1999) 205–230, https://doi.org/10.1162/evco.1999.7.3.205.
[60] S. Huband, P. Hingston, L. Barone, L. While, A review of multiobjective test problems and a scalable test problem toolkit, IEEE Trans. Evol. Comput. 10 (5) (2006) 477–506, https://doi.org/10.1109/tevc.2005.861417.
[61] H. Jiang, S. Zhang, Z. Ren, Solving multiobjective optimization problem by constraint optimization, in: Proceedings of 11th International Conference on Parallel Problem Solving from Nature, Krakow, Poland, 2010, pp. 637–646, https://doi.org/10.1007/978-3-642-15844-5_64.
[62] K. Deb, S. Jain, Running performance metrics for evolutionary multi-objective optimization, in: Proceedings of the 4th Asia-Pacific Conference on Simulated Evolution and Learning, Orchid Country Club, Singapore, 2002, pp. 13–20.
[63] S. Adriaensen, G. Ochoa, A. Nowé, A benchmark set extension and comparative study for the hyflex framework, in: IEEE Congress on Evolutionary Computation, Sendai, Japan, 2015, pp. 784–791, https://doi.org/10.1109/CEC.2015.7256971.
[64] H. Li, K. Deb, Q. Zhang, P.N. Suganthan, L. Chen, Comparison between MOEA/D and NSGA-III on a set of novel many and multi-objective benchmark problems with challenging difficulties, Swarm Evol. Comput. 46 (2019) 104–117, https://doi.org/10.1016/j.swevo.2019.02.003.
[65] X. Zhang, Y. Tian, R. Cheng, Y. Jin, A decision variable clustering based evolutionary algorithm for large-scale many-objective optimization, IEEE Trans. Evol. Comput. 22 (1) (2018) 97–112, https://doi.org/10.1109/tevc.2016.2600642.
[66] Z. Fan, W. Li, X. Cai, H. Li, C. Wei, Q. Zhang, K. Deb, E. Goodman, Push and pull search for solving constrained multi-objective optimization problems, Swarm Evol. Comput. 44 (2) (2019) 665–679, https://doi.org/10.1016/j.swevo.2018.08.017.