A Multi-objective Hyper-heuristic based on Choice Function

Mashael Maashi[1], Ender Özcan[2], Graham Kendall[3]

[1 2 3]*Automated Scheduling, Optimisation and Planning Research Group, University of Nottingham, Department of Computer Science, Jubilee Campus,Wollaton Road, Nottingham, NG8 1BB ,UK.*

[1]*email: psxmm3@exmail.nottingham.ac.uk*

[2]*email: ender.ozcan@nottingham.ac.uk*

[3]*The University of Nottingham Malaysia Campus*

email:graham.kendall@nottingham.edu.my

Abstract

Hyper-heuristics are emerging methodologies that perform a search over the space of heuristics in an attempt to solve difficult computational optimization problems. We present a learning selection choice function based hyper-heuristic to solve multi-objective optimization problems. This high level approach controls and combines the strengths of three well-known multi-objective evolutionary algorithms (i.e. NSGAII, SPEA2 and MOGA), utilizing them as the low level heuristics. The performance of the proposed learning hyper-heuristic is investigated on the Walking Fish Group test suite which is a common benchmark for multi-objective optimization. Additionally, the proposed hyper-heuristic is applied to the vehicle crashworthiness design problem as a real-world multi-objective problem. The experimental results demonstrate the effectiveness of the hyper-heuristic approach when compared to the performance of each low level heuristic run on its own, as well as being compared to other approaches including an adaptive multi-method search, namely AMALGAM.

Keywords: Hyper-heuristic, metaheuristic, evolutionary algorithm, multi-objective optimization.

1. Introduction

Most real-world problems are complex. Due to their (often) NP-hard nature, researchers and practitioners frequently resort to problem tailored heuristics to obtain a reasonable solution in a reasonable time. Generally, there are two recognized types of hyper-heuristics (Burke et al., 2013): (i) heuristic *selection* methodologies: (meta-)heuristics to choose (meta-)heuristics, and (ii) heuristic *generation* methodologies: (meta-)heuristics to generate new (meta-)heuristics from given components. A selection hyper-heuristic framework manages a set of low level heuristics and chooses one to be applied at any given time using a performance measure for each low level heuristic (Burke et al., 2013). The interest in selection hyper-heuristics has been growing in the recent years. However, the majority of research in this area has been limited to single-objective optimization.

A limited number of studies on selection hyper-heuristics have been introduced for multi-objective problems (see Table 1). Burke et al. (2003b) presented a multi-objective hyper-heuristic based on tabu search (TSRoulette Wheel), applying it to space allocation and timetabling problems.Veerapen et al. (2009) described another hyper-heuristic approach comprising two phases, applying it to the multi-objective traveling salesman problems. McClymont & Keedwell (2011) used a Markov chain-based learning selection hyper-heuristic (MCHH) for solving a real-world water distribution networks design problem. A new hyper-heuristic approach based on a multi-objective evolutionary algorithm i.e. NSGAII (Deb & Goel, 2001) was proposed in (Gomez & Terashima-Marìn, 2010,2012). NSGAII learned to choose from a set of rules representing a constructive heuristic for 2D irregular stock cutting. In (Furtuna et al., 2012) a multi-objective hyper-heuristic for the design and optimization of a stacked neural network is proposed. The proposed approach is based on NSGAII combined with a local search algorithm (Quasi-Newton algorithm). Rafique (2012) presented a multi-objective hyper-heuristic optimization scheme for engineering system design problems. A genetic algorithm, simulated annealing and particle swarm optimization are used as low-level heuristics. de Armas et al. (2011) and Miranda et al. (2010) described a representation scheme to be used in hyper-heuristics for multi-objective packing problems. Kumari et al. (2013) presented a multi-objective hyper-heuristic genetic algorithm (MHypGA) for the solution of a multi-objective software module clustering problem. In MHypGA, different methods of selection, crossover and mutation operations of genetic algorithms incorporated as a low-level heuristics. Vázquez-Rodríguez & Petrovic (2013) proposed a multi-indicator hyper-heuristic for multi-objective optimization. This was approach based on multiple rank indicators that taken from NSGAII (Deb & Goel, 2001), IBEA (Zitzler & Künzli, 2004) and SPEA2 (Zitzler et al., 2001). Len et al. (2009) proposed a hypervolume-based hyper-heuristic for a dynamic-mapped multi-objective island-based model. Bai et al. (2013) proposed a multiple neighbourhood hyper-heuristic for two-dimensional shelf space allocation problem. The proposed hyper-heuristic was based on a simulated annealing algorithm.

Different frameworks have been proposed for mixing a set of existing algorithms applied to different problems, such as an adaptive multi-method search (AMALGAM) (Vrugt & Robinson, 2007; Raad et al., 2010; Zhang et al., 2010) and multi-strategy ensemble multi-objective evolutionary algorithm (Wang & Li, 2010).

None of the above have used multi-objective evolutionary algorithms (MOEAs), with the exception of (Gomez & Terashima-Marín, 2010; Vrugt & Robinson, 2007; Rafique, 2012) and none of the standard multi-objective test problems are studied, except in (McClymont & Keedwell, 2011; Vrugt & Robinson, 2007; Len et al., 2009; Vázquez-Rodríguez & Petrovic, 2013). Moreover, none of the previous hyper-heuristics make use of the components specifically designed for multi-objective optimization that we introduce. This paper highlights the need for scientific study in the research area of multi-objective evolutionary algorithms and hyper-heuristics. We focus on an online learning selection choice function based hyper-heuristic, to solve continuous multi-objective optimization problems, and their hybridization with multi-objective evolutionary algorithms which controls and combines the strengths of three well-known multi-objective evolutionary algorithms (NSGAII (Deb & Goel, 2001), SPEA2 (Zitzler et al., 2001), and MOGA (Fonseca & Fleming, 1998)). The choice function was successful when used as a selection method for single-objective optimization (Cowling et al., 2002; Kendall et al., 2002). To the best of our knowledge, no work been reported in the literature that utilizes the choice function as selection method within a hyper-heuristic framework for multi-objective optimization.

Table 1: Heuristic components and application domains of hyper-heuristics for multi-objective optimization

Component Name	Application Domain \Test Problems	Reference(s)
Tabu Search	Space allocation, Timetabling	Burke et al. (2003b)
	Travelling salesman problems	Veerapen et al. (2009)
Markov Chain, Evolution Strategy	Real-world water distribution networks design \DTLZ, WFG	McClymont and Keedwell (2011)
NSGAII	Irregular 2D cutting stock	Gomez and Terashima-Marìn (2010, 2012)
	Strip packing and Cutting stock	de Armas et al. (2011), Miranda et al. (2010)
NSGAII, Quasi-Newton algorithm	Stacked neural network	Furtuna et al. (2012)
Number of Operations from NSGAII, SPEA2 and IBEA	A number of continuous multi-objective test problems	Vázquez-Rodríguez & Petrovic (2013)
Number of Selection, Crossover and Mutation Operations of Evolutionary Algorithms	Software Module Clustering	Kumari et al. (2013)
Hypervolume	Dynamic-mapped island-based model	Len et al. (2009)
Particle Swarm Optimization, Adaptive Metropolis Algorithm, Differential Evolution	Water resource problems \A number of continuous multi-objective test problems	Vrugt and Robinson (2007), Raad et al. (2010), Zhang et al. (2010)
Memory Strategy, Genetic and Differential Operators	Dynamic optimization problems \A Number of continuous multi-objective test problems	Wang and Li (2010)
Genetic Algorithm, Simulated Annealing, Particle Swarm Optimization	Engineering system design problems \A number of classical multi-objective test problems	Rafique (2012)
Simulated Annealing	Shelf space allocation	Bai et al. (2013)

Our hyper-heuristic for multi-objective optimization addresses the research areas of multi-objective evolutionary algorithms and hyper-heuristics. Section 2 discusses each one of these areas. The rest of the paper is organized as follows. Section 3 provides the details of the proposed hyper-heuristic framework for multi-objective optimization. The empirical results comparing our approach to the well known multi-objective evolutionary algorithms that are used as the low level heuristics are presented in Section 4. The comparison of our multi-objective hyper-heuristic to other approaches over benchmark test problems and a real-world problem are presented in Section 5 and 6 respectively. Section 7 summarizes and discusses possible future research directions.

2. Multi-objective Optimization

A multi-objective optimization problem (MOP) comprises several objectives, which need to be minimized or maximized depending on the problem. In the literature, many similar techniques are presented for multi-objective optimization. An example is a posteriori search is conducted to find solutions for the objective functions. Following this, a decision process selects the most appropriate solutions often involving a trade off. Examples of this methodology are multi-objective evolutionary optimization (MOEA) methods, whether non Pareto-based or Pareto-based methods. The Pareto-based evaluation is a method used to evaluate the quality of MOP solutions. In Pareto-based methods, all objectives are simultaneously optimized by applying Pareto dominance concepts. The idea behind the dominance concept is to generate a preference between MOP solutions since there is no information regarding the objective preference provided by the decision maker. Tan et al. (2002) and Coello et al. (2007) present a more formal definition of Pareto dominance.

$Definition$ 1: A vector $u = (u_1, ..., u_k)$ is said to dominate another vector $v = (v_1, ..., v_k)$ (denoted by $u \preceq v$) according to k objectives, if and only if, u is partially less than v, i.e., $\forall\, i \in \{1, ..., k\}$, $u_i \leq v_i \wedge \exists\, i \in \{1, ..., k\}$: $u_i < v_i$.

In the literature, various features for multi-objective optimization test problems are presented. Those features are designed to make the problems difficult enough to examine algorithmic performance. Examples of these features are deception (Goldberg, 1987; Whitley, 1991), multimodality (Horn & Goldberg, 1995), noise (Kargupta, 1995), and epistasis (Davidor, 1991). Moreover, other features of test problems are suggested in (Deb, 1999), such as multi-modality, deceptive, isolated optimum and collateral noise. These features can cause difficulties for evolutionary optimizers in terms of converging to the Pareto optimal front (POF) and maintaining the population diversity. Furthermore, some characteristics of the POF such as convexity or nonconvexity, discreteness, and nonuniformity could cause difficulties in term of the population diversity (Zitzler et al., 2000).

Typically, a test suite should include different test problems which consist of a wide range of characteristics and features as mentioned perviously. However, it is impractical to have a test suite that incorporates all possible combinations of features. The test suites most commonly employed as benchmark multi-objective problems in the MO/EA literature are the ZDT test suite (Zitzler et al., 2000), the DTLZ test suite (Deb et al., 2002), the WFG (Huband et al., 2006) and more recently LZ09 (Li & Zhang, 2009).

The Walking Fish Group's (WFG) test suite (Huband et al., 2006) consists of nine test problems (see Table 2). The benchmark problems fully satisfy the earlier recommendations. The WFG is designed only for real valued parameters with no side constraints. They make the problems easy to analyze and implement. The features of the WFG dataset are seen as the common choice for most MO/EA researchers (Huband et al., 2006). Unlike most of the multi-objective test suites such as ZDT (Zitzler et al., 2000) and DTLZ (Deb et al., 2002), the WFG test suite has powerful functionality since it has instances with distinct features compared to other test suites. Therefore, the WFG test suite is used as our benchmark dataset in this study.

2.1. Multi-objective Evolutionary Algorithms (MOEAs)

Many EA researchers would argue that evolutionary algorithm(s) are more suitable methods to deal with multi-objective optimization problems (Coello et al., 2007; Deb, 2001; Anderson et al., 2007; Bäck, 1996; Deb & Goldberg, 1989; Fonseca & Fleming, 1998; Zhang & Li, 2007; Miranda et al., 2010) because of their population-based nature, which means they can find Pareto-optimal sets (trade-off solutions) in a single run which allow a decision maker to select a suitable compromise solution.

This subsection focuses on the well known, efficient and effective Pareto-based approaches that we have used in our study, namely Multi-objective Genetic Algorithm (MOGA) (Fonseca & Fleming, 1998), Non-dominated Sorting Genetic Algorithm (NGSAII) (Deb & Goel, 2001), and Strength

Pareto Evolutionary Algorithm (SPEA2) (Zitzler et al., 2001). A survey of MOEAs can be found in (Zhou et al., 2011).

MOGA was proposed by Fonseca & Fleming (1993). The Pareto ranking scheme is used i.e. each solution in the current population is given a rank based on their dominance (Veldhuizen & Lamont, 2000; Landa-Silva, 2003). A modified version of this algorithm has been proposed in (Fonseca & Fleming, 1998). This version employed restricted sharing between solutions that have the same rank and the distance between two solutions is computed and compared to the key sharing parameter σ_{share}. While MOGA is efficient and easy to implement, its fitness sharing method prevents two vectors that have the same value in the objective space existing simultaneously unless the fitness sharing is genotypic-based.

NSGAII (Deb & Goel, 2001) is a non-explicit building block MOEA that incorporates the concept of elitism (Coello et al., 2007; Deb, 2005). The solutions compete, then each solution is are ranked and sorted based on its Pareto-optimal level. Genetic operators are applied to generate a new group of children who are then merged with parents in the population (Coello et al., 2007). Furthermore, a niching method based on crowding distance is used during the selection process in order to maintain a diverse Pareto front (Zhang & Li, 2007). Although NSGAII is efficient, it still has some drawbacks. It is unable to generate a Pareto optimal set in some regions of the search space, particularly in unpopulated regions (Coello & Pulido, 2001). In addition, its search bias strongly appears as the number of objectives rises (Jaszkiewicz, 2001). In other words, the algorithm seems to have bias towards some regions in the search space.

SPEA2 (Zitzler et al., 2001) incorporates a fine-grained fitness assignment strategy which considers the number of individuals for each solution that dominates it and which it is dominated by. It uses a nearest neighbor density estimation technique in order to increase the efficiency of the search. SPEA2 improves the archive truncation method that guarantees the preservation of boundary points by replacing the average linkage method used in the earlier version SPEA (Zitzler & Thiele, 1999). Experimental results show that SPEA2 performs well in terms of diversity and distribution as the number of objectives increases. In addition, it significantly outperforms its predecessor.

2.2. Studies on the Comparison of MOEAs

Most MOEAs have common strategies that are employed in their search process. However, they differ in the way that they apply these strategies. MOGA classifies the solutions based on the ranking scheme using linear or exponential interpolation and applies the sharing scheme in the objective space. NSGA uses dummy fitness values assigned to the solutions and applies the sharing scheme in the decision variable space (Veldhuizen & Lamont, 2000). NSGA with elitism performs as well as a SPEA (Zitzler et al., 2000).

In the literature, some studies have compared MOEAs' performance and quality against each other. A comparison study for SPEA2, NSGAII and MOGA on ZDT4 and ZDT6 problems (Zitzler et al., 2000) was presented in (Watanabe et al., 2002). With respect to the ratio of the non-dominated individual metric (RNI), NSGAII has better performance than the others on ZDT4. However, SPEA2 outperforms MOGA and NSGAII for the same metric on ZDT6. The authors concluded this study by stating that SPEA2 has an advantage in its accuracy over NSGAII. While NSGAII is superior to SPEA2 in finding wide spread solutions. In (Khare et al., 2003), another a comparative study for NSGAII, SPAE2 and PEAS on four test problems (DTLZ1, DTLZ2, DTLZ3 and DTLZ6) (Deb et al., 2002) with 2-8 objectives was carried out. Three performance metrics were used for convergence and diversity of the obtained non-dominated set and the running time that a MOEA requires to execute. SPEA2 performs better than NSGAII in terms of convergence for a small number of objectives. However, both perform similarly for a higher number of objectives. SPEA2 and NSGAII have good performance with respect to the diversity, but they have some difficulties in the closeness of the obtained non-dominated set to the POF. In comparison, PEAS (Liu et al., 2007) performs very well in converging to the true front but it fails in diversity and it requires a longer computational time as the number of objectives increases. However, NSGAII requires less run time to run as the number of objectives increases. Another comparative study between NSGAII and SPEA2 on the WFG test problems (Huband et al., 2006) with 24 real values and a different scale of objectives was presented in (Bradstreet et al., 2007). For two objectives, NSGAII is superior to SPEA2 on the WFG test problems with respect to the epsilon metric and the hypervolume (SSC). In contrast, SPEA2 outperforms NSGAII on all WFG problems expect WFG3 in three objectives with respect to the same two metrics. We can note from two last studies that the number of objectives can affect the performance of an algorithm. SPEA2 works well with a high number of objectives for WFG and a low number of objectives for DTLZ. The opposite is true for NSGAII. We can also observe from these comparative studies that an algorithm can perform better than other algorithms with respect to a specific metric on a certain problem, while another algorithm performs better than another algorithm with respect to another metric for the same problems. Moreover, the performance of a multi-objective algorithm could vary with respect to the number of objectives, it is able to effectively cope with. All these observations could be an advantage when combining different algorithms under a hyper-heuristic framework for multi-objective optimization to derive the strengths of the algorithms and avoid their weaknesses.

2.3. Selection Hyper-heuristics

In a hyper-heuristic approach, different heuristics or heuristic components can be selected, generated or combined to solve a given computationally difficult optimization problem in an efficient and effective way. The task of the high level strategy is to guide the search intelligently and adapt considering the success/failure of the low level heuristics or combinations of heuristic components during the search process. Hyper-heuristics are sufficiently general and modular search methods enabling reuse of their components for solving problems from different domains (Qu & Burke, 2009). The focus of this study is selection hyper-heuristics which perform a search using two successive stages (Burke et al., 2013; Özcan et al., 2008): *(meta-)heuristic selection* and *acceptance*. An initial solution or (a set of initial solutions) is iteratively improved using the low level (meta-)heuristics until some termination criteria are satisfied. During each iteration, the (meta-)heuristic selection decides which low level (meta-)heuristic will be employed next. After the selected (meta-)heuristic is applied to the current solution/s, a decision is made whether to accept the new solution/s or not using an acceptance method. The low level (meta-)heuristics in a selection hyper-heuristic framework are in general human designed heuristics which are fixed before the search starts.

Usually, in a selection hyper-heuristic framework, there is a clear separation between the high level hyper-heuristic approach also referred to as a *strategy* and the set of low-level heuristics or heuristic components. It is assumed that there is a domain barrier between them (Burke et al., 2003a). The purpose of domain barrier is increase the level of the generality of hyper-heuristic by being able to apply it to a new of problem without changing the framework, only a new set of problem-related low-level heuristics need to be supplied. The barrier allows only problem domain independent information to flow from the low level to the high level, such as the fitness/cost/penalty value measured by an evaluation function, indicating the quality of a solution (Hussin, 2005). Low level heuristics, or heuristic components, are the problem domain specific elements of a hyper-heuristic framework. Hence they have access to any relevant information, such as candidate solution/s.

Most of the existing selection hyper-heuristics are based on perturbative low level heuristics, and favor single-point based search. Cowling et al. (2002) investigated the performance of different hyper-heuristics, combining different heuristic selection, with different move acceptance methods, on a real world scheduling problem. *Simple Random*, *Random Descent*, *Random Permutation*, *Random Permutation Descent*, *Greedy* and *Choice Function* were introduced as heuristic selection methods. The authors utilized the following deterministic acceptance methods: All-Moves accepted and Only Improving moves accepted. The hyper-heuristic combining Choice Function with All-Moves acceptance performed the best. In (Cowling et al., 2002; Kendall et al., 2002), the choice function heuristic selection method is adaptively ranks the low-level heuristics (h_i) using Eq. 1 .

$$f(h_i) = \alpha f_1(h_i) + \beta f_2(h_j, h_i) + \delta f_3(h_i) \tag{1}$$

where f_1 measures the individual performance of each low level heuristic, f_2 measures the performance of pairs of low level heuristics invoked consecutively, and finally, f_3 is the elapsed CPU time since the heuristic was last called. Both f_1 and f_2 support intensification while f_3 supports diversification. The parameter values for α, β and δ are changed adaptively based on a reinforcement learning strategy. In (Kendall et al., 2002), the choice function based hyper-heuristic was applied to nurse scheduling and sales summit scheduling. The study shows that the choice function hyper-heuristic is successful in making effective use of low level heuristics, due to its ability of learning the dynamics between the solution space and the low level heuristics to guide the search process towards better quality solutions. There are an increasing number of studies showing the success of choice function heuristic selection (Özcan et al., 2008; Gibbs et al., 2011;

Burke et al., 2012). In these studies, a choice function is used for solving single-objective problems. Unlike these studies, we introduce a selection hyper-heuristic framework based on choice function which is modified to deal with multi-objective optimization problems and a mechanism to rank low level heuristics for heuristic selection. More on selection hyper-heuristics including an overview of hyper-heuristic components, different framework, application areas can be found in (Burke et al., 2013).

3. A Selection Hyper-heuristic Framework for Multi-objective Optimization

The design of the framework for our multi-objective hyper-heuristic is motivated in two ways.. Firstly, there is no existing algorithm that excels across all types of problems. In the context of multi-objective optimization, no single MOEA algorithm has the best performance with respect to all performance measures on all types of multi-objective problems. Some comparison studies in MOEAs which emphasizes this idea are presented in Section 2.2. Secondly, the hybridization or combining different of (meta-)heuristics/algorithms into one framework could yield promising results compared to (meta)heuristics/algorithms when used alone. we are looking to gain an advantage of combining different algorithms in a hyper-heuristic framework for multi-objective optimization to get benefit from the strengths of the algorithms, whilst avoiding their weaknesses.

The idea of hybridizing a number of algorithms (heuristics) into a selection hyper-heuristic framework is straightforward. However, many design issues related to the development of hyper-heuristics for multi-objective optimization require more attention when designing such a framework to be applicable and effective. One of design issues choosing appropriate low-level heuristics, is a challenging task. Many questions arise. What is the heuristics (algorithms) are suitable to deal with multi-objective optimization problems, are priori approaches or a posteriori approaches more suitable etc?. As the aims of hyper-heuristic is to raise the level of generality, a posteriori approach is more suitable to achieve this aim. Unlike the priori approaches, there is no need to set objective preferences or weights prior to the search process in the posteriori approach such as MOEAs. We agree with many researchers (Coello et al., 2007; Deb, 2001; Anderson et al., 2007; Bäck, 1996; Deb & Goldberg, 1989; Fonseca & Fleming, 1998; Zhang & Li, 2007; Miranda et al., 2010) that evolutionary algorithms are more suitable methods to deal with multi-objective optimization problems because of their population-based nature. This means that they can find Pareto optimal sets (trade-off solutions) in a single run, which allow a decision maker to select a suitable compromise solution (with respect to the space of the solutions). In the context of multi-objective hyper-heuristics, a decision maker could be a selection method that decides which is the best low level heuristics to select at each decision point. According to some studies in MOEAs, that presented in Section 2.2, we choose three well-known multi-objective evolutionary algorithms (NSGAII (Deb & Goel, 2001), SPEA2 (Zitzler et al., 2001), and MOGA (Fonseca & Fleming, 1998)) to act as low level heuristics. Although NSGAII, SPEA2 and MOGA are no longer considered good, they are still viewed as a good baseline for MOEA research. They incorporate much of the known MOEA theory (Veldhuizen & Lamont, 2000) which make them applicable to investigate their combination under the multi-objective hyper-heuristic framework. As one of our hyper-heuristic aims is to benefit from the strengths of low level heuristics and avoid their weaknesses, the features of NSGAII, SPEA2 and MOGA enable us to investigate this aspect.

Another design issue related to the development of hyper-heuristics for multi-objective optimization is a selection mechanism. As a selection hyper-heuristic relies on an iterative process, the question arises what is an effective way to choose an appropriate heuristic at each decision point. In single-objective case, this criterion is easy to determine by measuring the quality of the solution such as the objective/cost value and time. However, this is more difficult when tackling a multi-objective problem. The quality of the solution is hard to assess as many different criteria be considered such as the number of non-dominated individuals and the distance between the non-dominated front and the POF. As we aim to keep the framework simple, we not employ information problem specific such as the number of objectives nor the nature of the solution space. We focus more on the performance of low level heuristics. Thus, high performing heuristics are chosen more frequently for intensification. Diversification is also taken into account, so that a balance between intensification and diversification is provided. The choice function heuristic selection gives a balance between intensification and diversification. As mentioned earlier, the measurement of the quality of the solution for multi-objective problems requires us to assess different aspects of the non-dominated set in the objective space. According to Tan et al. (2002), no single MOEA excels in all performance measures. We employ a learning mechanism based on different measures using a ranking scheme to provide a feedback about the quality of the solutions. We do not aim to choose a heuristic that performs well with respect to all measures. We aim to select a heuristic that performs well in most measures. The learning mechanism that is employed in our multi-objective hyper-heuristic are provided in Section 3.1.

In this study, we present a selection choice function based hyper-heuristic for multi-objective optimization denoted as HH_CF. The choice function heuristic selection acts as the high level strategy which adaptively ranks the performance of three low level meta-heuristics deciding which one to call at each decision point during the search process. All-Moves is employed as a deterministic acceptance strategy, meaning that we accept the output of each low level heuristic whether it improves the quality of the solution or not.

3.1. Choice Function and a Ranking Scheme for Multi-objective Optimization

The HH_CF framework imposes the domain barrier concept. No problem specific information is exchanged between the high level hyper-heuristic and low level meta-heuristics. However, the framework enables us to maintain relevant information on how each low level meta-heuristic performs. In order to provide some information as a feedback, regardless of the multi-objective problem, four performance metrics are maintained as shown in (Tan et al., 2002). The high level strategy selects one low level heuristic at each decision point according to the information obtained from the feedback mechanism. Note that the three low level heuristics operate in an encapsulated way. Each approach has its own characteristic as described in section 2.1, but they share the same population.

Four performance metrics are employed in the proposed HH_CF framework as a feedback mechanism:

- Algorithm effort (**AE**) (Tan et al., 2002): measures the computational effort of an algorithm to obtain the Pareto optimal set. It ranges from $[0,\infty)$. A smaller value of AE indicates better performance.

- Ratio of non-dominated individuals (**RNI**) (Tan et al., 2002): evaluates the fraction of non-dominated individuals in the population. It ranges from [0,1]. If RNI = 1, this indicates that all individuals for a given population are non-dominated .

- Size of space covered or so-called S_metric Hypervolume (**SSC**) (Zitzler & Thiele, 1999): evaluates the size of the objective functions space covered by the solutions around the POF . It ranges from [0,∞). A higher value of SSC indicates better performance.

- Uniform distribution of a non-dominated population (**UD**) (Srinivas & Deb, 1994): evaluates the distribution of non-dominated individuals over the POF. It ranges from [0,1]. A higher value of UD indicates better performance.

These metrics are chosen as they have been commonly used for MOEAs to measure different aspects of the final population (Tan et al., 2002). In addition, they do not require prior knowledge of the POF. This means that our framework is suitable for tackling any given problem in future studies.

The performance metrics (AE, RNI, SSC, UD) (Tan et al., 2002) are used as performance indicators for each low level heuristic. They serve the high level online learning mechanism which guides the search to determine which low level heuristic should be selected. Since the performance metrics provide values that are in different scalar units, it is not trivial to combine those values into a single score which can be used to select the best heuristic. Therefore, we use a ranking scheme in order to select the best performing heuristic at each step. This ranking scheme is simple and flexible. It enables us to incorporate any number of heuristics and even performance indicators.

We propose the following choice function heuristic selection based on two stage ranking scheme (f_1) to be used as a part of the selection hyper-heuristic for multi-objective optimization.

$$CF(h) = \alpha f_1(h) + f_2(h) \tag{2}$$

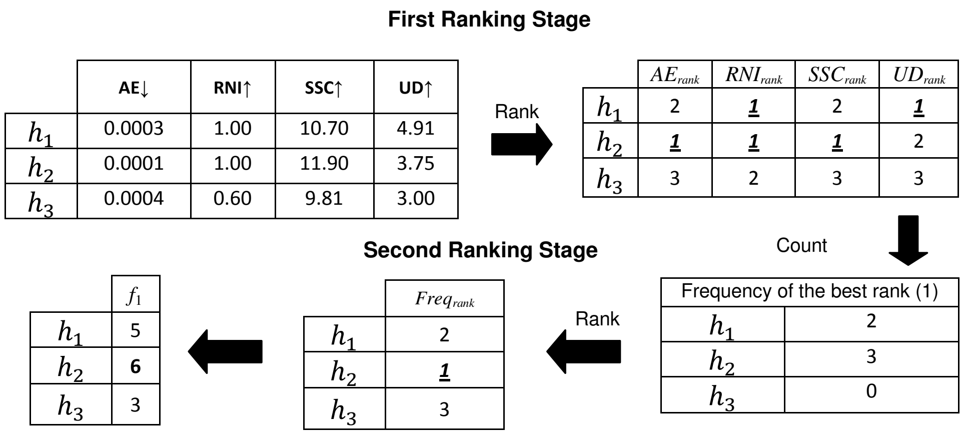
Eq. 2 differs from Eq. 1 as it is adjusted to deal with a given multi-objective optimization problem, but their goal is the same, measuring the overall performance of a low level heuristic. In Eq. 2, $CF(h)$ is an overall score of each heuristic h. $f_1(h)$ embeds the ranking scheme and is used for intensification. $f_2(h)$ is the number of CPU seconds elapsed since the heuristic was last called. This provides an element of diversification, by favoring those low level heuristics that have not been called recently. α is a large positive value (e.g. 100). It is important to strike a balance between $f_1(h)$ and $f_2(h)$ values, so that they are in the same scalar unit. The low level heuristic with the highest value of $CF(h)$ is the heuristic that is applied. In the case of multiple heuristics having the same scores, then we choose one of them randomly.

Unlike the ranking scheme that was used in (Vázquez-Rodríguez & Petrovic, 2013) which ranked heuristics based on their probabilities against the performance indicators using a mixture of experiments, our ranking scheme operates in two successive stages. Fig. 1 illustrates how the ranking scheme works based on four performance metrics to rank three heuristics, denoted as $h1$-$h3$. In the first stage, each heuristic is ranked with respect to each performance indicator, separately. For this purpose, a matrix of $N \times M$ is used, where N and M is the number of heuristics and performance metrics, respectively. All heuristics are ranked according to their performances against each metric. The rankings of heuristics for each metric get recorded as a column of the matrix, the best and worst rank being 1 and N, respectively. If two heuristics have the same performance, both heuristics are assigned to the same rank. In the second stage, all heuristics are ranked according to their frequency count of the best rank, that is 1, from the first stage of ranking. This ranking is denoted as $Freq_{rank}(h)$ for each heuristic. Finally, each heuristic is ranked according to its frequency of count of the best rank. In Fig. 1, h_2 has the best final rank, after the second stage of ranking.

As we do not only look for the heuristic that has the best performance, but we also aim to have a large number of non-dominated individuals. For each heuristic, the rank of the frequency count of the best rank is added to its RNI rank using the following equation:

$$f_1(h) = 2 \times (N + 1) - \{Freq_{rank}(h) + RNI_{rank}(h)\} \tag{3}$$

$f_1(h)$ represents the performance of an individual heuristic h. It is structured to favor the best performing low level heuristic with respect to as many metrics as possible used in the system. For example, h_2 has the highest $f_1(h)$ in Fig. 1 and if durations of all heuristics being invoked last time are the same, then h_2 will be selected, since it will have the highest score.



	AE↓	RNI↑	SSC↑	UD↑
h_1	0.0003	1.00	10.70	4.91
h_2	0.0001	1.00	11.90	3.75
h_3	0.0004	0.60	9.81	3.00

	AE_{rank}	RNI_{rank}	SSC_{rank}	UD_{rank}
h_1	2	***1***	2	***1***
h_2	***1***	***1***	***1***	2
h_3	3	2	3	3

Frequency of the best rank (1)	
h_1	2
h_2	3
h_3	0

	$Freq_{rank}$
h_1	2
h_2	***1***
h_3	3

	f_1
h_1	5
h_2	**6**
h_3	3

Figure 1: An example of how f_1 is computed based on the two stage ranking process, given three low level heuristics, denoted as h_1, h_2 and h_3. Those heuristics are ranked using four performance metrics of AE, RNI, SSC, and UD in the first stage. The $\downarrow$ and $\uparrow$ show whether heuristics are ranked in decreasing or increasing order for the associated metric

3.2. The Multi-objective Hyper-heuristic

Algorithm 1 shows how a greedy algorithm is applied initially to determine the best low level heuristic to apply in the first iteration (steps 2-6). All three low level heuristics are run (step 3). Then, the three low level heuristics scored and ranked using Eq. 3 and their choice function values are computed by using Eq. 2 (steps 4&5). The low level heuristic with the highest choice function value is selected (step 6) to be applied as an initial heuristic (step 8). Then, for all low level heuristics, the ranking mechanism is updated (step 9). The choice function values are also computed and updated (step 10). According to the updated choice function values, the low level heuristic with the highest choice function value is selected to be applied in the next iteration (step 11). This process is repeated until the stopping condition is met (steps 7-12). Note that the greedy algorithm is applied only once at the beginning of the search, in order to determine which low level heuristic to apply first. Then, only one low level heuristic is selected at each iteration.

Algorithm 1 Multi-Objective Hyper-heuristic Algorithm

```
1:  procedure HH_CF($H$) where $H$ is a set of the low level heuristics.
2:      Initialization
3:      Run $h$,$\forall\ h \in H$
4:      Rank $h$, $\forall\ h \in H$ based on the ranking scheme
5:      Get $CF(h)$, $\forall\ h \in H$
6:      Select $h$ with the highest $CF(h)$ as an initial heuristic
7:      repeat
8:          Execute the selected $h$
9:          Update the rank of $h$, $\forall\ h \in H$ based on the ranking scheme
10:         Update $CF(h)$, $\forall\ h \in H$
11:         Select $h$ with the highest $CF(h)$, $\forall\ h \in H$
12:     until (termination criteria are satisfied)
13: end procedure
```

4. Performance Comparison of HH_CF and Low level Meta-heuristics

A set of experiments using the WFG test suite (Huband et al., 2006) is conducted to see the performance difference when using each individual multi-objective meta-heuristic (NSGAII, SPEA2, and MOGA) run on its own and the proposed HH_CF selection hyper-heuristic approach, that combines the three multi-objective meta-heuristics. Although NSGAII and SPEA2 have previously been applied to the WFG test suite in (Bradstreet et al., 2007), we repeat the experiments, including MOGA, under our own experimental settings.

4.1. Experimental Settings and Performance Evaluation Criteria

Nine test problems for the WFG suite (WFG1-WFG9) have 24 real parameters including four position parameters, 20 distance parameters and two objectives. We agree with (Zitzler et al., 2000) that two objectives are enough to represent the essential features of multi-objective optimization problems to demonstrate the significance of the proposed approach. All settings for the test suit are fixed using the same settings proposed in the previous studies (Zitzler et al., 2000; Huband et al., 2006). According to (Voutchkov & Keane, 2010; Chow & Regan, 2012) an algorithm could reach better convergence by 6,250 generations. Therefore, the HH_CF was terminated after 6,250 generations. That is, HH_CF runs for a total of 25 iterations. In each iteration, one low level heuristic is applied and this is executed for 250 generations, with a population size of 100. The secondary population of SPEA2 is set to 100. The execution time takes about 10-30 minutes depending on the given problem. In order to make a fair comparison, each low level heuristic is used in isolation and is terminated after 6,250 generations. For all three low level heuristics, the simulated binary crossover (SBX) operator is used for recombination and a polynomial distribution for mutation (Deb & Agrawal, 1995).

For the WFG problems, 30 independent trials were run for each algorithm with a different random seed. The crossover and mutation probability were set to 0.9 and 1/24 respectively. The distribution indices for crossover and mutation were set to 10 and 20 respectively. In the measure of SSC, the reference points for WFG problems with k objectives was set $r_i = (0, i * 2)$, $i = 1, ..., k$ (Huband et al., 2006) . The distance sharing σ for the UD metric and MOGA was set to 0.01 in the normalized space. These settings were used for SSC and UD as a feedback indicator in the ranking scheme of HH_CF and as a performance measure for the comparison. All algorithms were implemented with the same common sub-functions using Microsoft Visual C++ 2008 on an Intel Core2 Duo 3GHz\2G\250G computer.

The comparison of the quality of solutions for multi-objective optimization is more complex than single-objective problems. The number of non-dominated individuals should be maximized, the distance of the non-dominated front should minimized, i.e. the resulting non-dominated set should be distributed as uniformly possible and converge well toward the POF. Because of that,we use three performance metrics RNI, SSC, and UD, to assess the quality of approximation sets in different aspects. These performance metrics are also used to measure the effectiveness of the ranking scheme in the HH_CF as they are employed as feedback indicators for low level heuristics. In addition, we used a *t*-test as the statistical test while comparing the average performance of a pair of algorithms with respect to a metric averaged over 30 trials. The null hypothesis is as follows:

$$\begin{cases} H_0 \text{ the performance of a pair of algorithms have same means} \\ H_1 \text{ the performance of a pair of algorithms have different means} \end{cases}$$

We assume two independent samples, unequal variance and one-tailed distribution with a 95% confidence level. We aim to reject the null hypothesis and accept the alternative hypothesis and demonstrate that our proposed choice function based hyper-heuristics HH_CF is statistically different from the three low level heuristics (NSGAII, SPEA2, and MOGA), when used in isolation.

We use the following notation. Given two algorithms P (left) and Q (right), P-Q $+$ $(-)$ indicates that P performs better than Q on average and this performance difference is statistically significant. The $\sim$ sign indicates that both algorithms deliver a similar performance. n/a means the *t*-test is not applicable for two samples since they are completely equal.

4.2. Results

NSGAII, SPEA2, MOGA and HH_CF are tested on the nine WFG test problems under the same experimental settings described in the previous section. Table 3 summarizes the average and standard deviation value pairs for each algorithm with respect to RNI, SSC, and UD over 30 trials. For all performance metrics, a higher value indicates a better performance. HH_CF has a higher RNI value than MOGA while it has a lower value than NSGAII and SPEA2 for WFG1. HH_CF has the highest value of SSC and UD metrics among the methods. We can put WFG5 and WFG6 in this category. For WFG2 and WFG3, HH_CF has a RNI value similar to MOGA and lower than the others. With respect to SSC, HH_CF has higher values than SPEA2 and MOGA and similar to NSGAII. However, HH_CF has the highest value among other methods in the measure of UD. For WFG4 and WGF7, HH_CF has the lowest (worst) RNI value and the highest UD value. HH_CF has a higher value than MOGA similar to NSGAII and SPEA2 with respect to the SSC metric. For WFG8 and WFG9, the HH_CF has the lowest value with respect to RNI and SSC metrics, and the highest value with respect to UD metric.

These performance results with respect to RNI, SSC and UD are also displayed as box plots in Figs. 2, 4 and 3 in order to provide a clear visualization of the distribution of the simulation data of the 30 independent runs. The statistical *t*-test comparing our proposed HH_CF and the three low level heuristics (NSGAII, SPEA2, and MOGA), when used in isolation for the three performance metrics (RNI, SSC, and UD) are given in Table 4. We note that HH_CF and other algorithms are statistically different in the majority of cases.

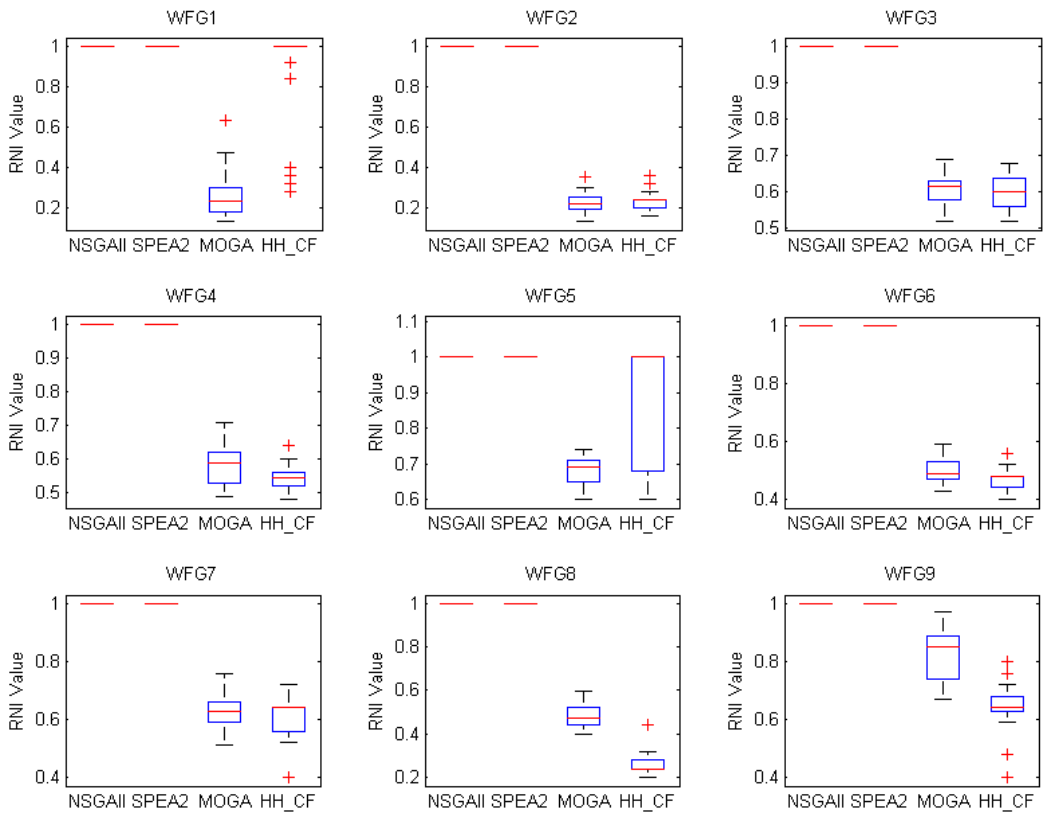


Figure 2: Box plots of NSGAII, SPEA2, MOGA and HH_CF, for the measure of ratio of non-dominated individuals (RNI) on the WFG test functions

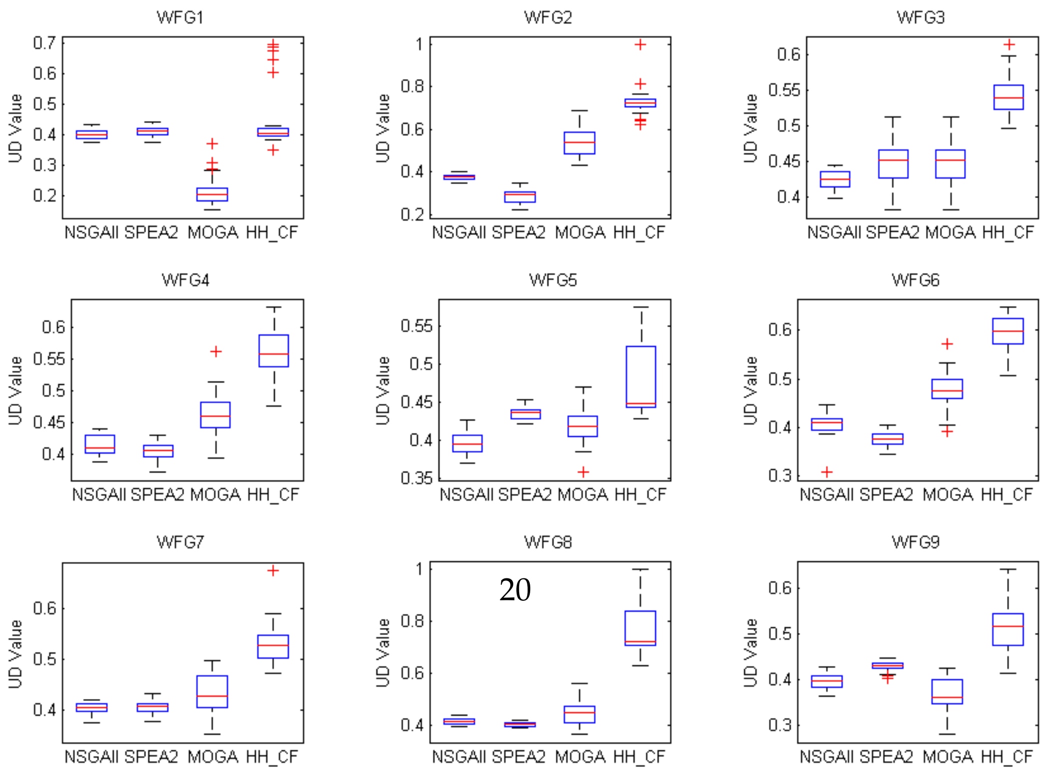


Figure 3: Box plots of NSGAII, SPEA2, MOGA and HH_CF for the uniform distribution (UD) of non-dominated population on the WFG test functions

In Fig. 2, NSGAII and SPEA2 perform better than the others and produce the highest value of RNI for all datasets. This performance variation is statistically significant as illustrated in Table 4. Moreover, NSGAII and SPEA2 performs the same across all benchmark with respect to RNI. However, HH_CF and MOGA produce relatively low values for this metric. HH_CF performs significantly better than MOGA on two instances of WFG1 and WFG5 and vice-versa for two instances of WFG8 and WFG9. For the rest of the instances, they deliver the same performance. This indicates that HH_CF performs badly according to the metric of RNI and produces a low number of non-dominated solutions than other algorithms, except for MOGA.

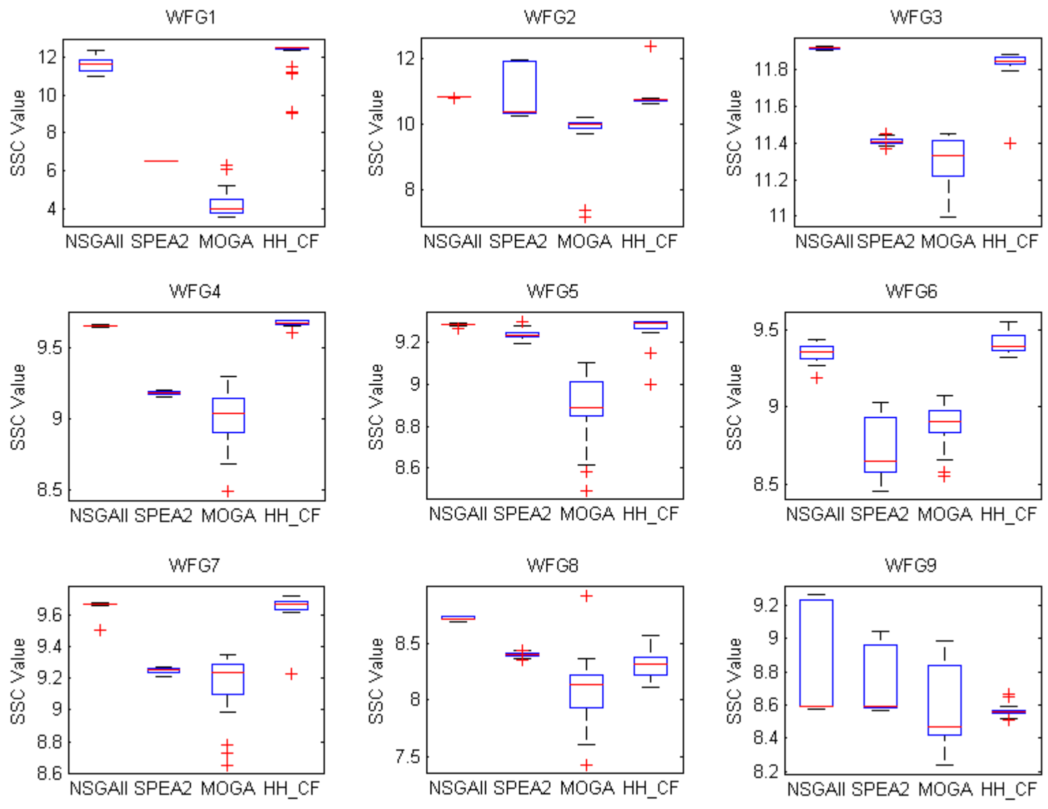


Figure 4: Box plots of NSGAII, SPEA2, MOGA and HH_CF for the measure of hypervolume (SSC) on the WFG test functions

In Fig. 3, it can be seen that HH_CF has the highest uniform distribution UD value across all test problems. This indicates that HH_CF is superior to the other algorithms on all WFG instances in terms of the distribution of non-dominated individuals over the POF. This performance variation is statistically significant as illustrated in Table 4. HH_CF performs significantly better than the other methods on all nine instances of WFG.

In Fig. 4, the performance of HH_CF for SSC is relatively better than SPEA2 and MOGA across all test problems except for WFG9. HH_CF performs significantly better than SPEA2 and MOGA on eight instances of WFG (see Table 4). HH_CF also performs better than NSGA2 in WFG1, WFG5 and WFG6. This performance variation is statistically significant as illustrated in Table 4. HH_CF performs significantly better than NSGAII on three instances (WFG1 and WFG5, WFG6).

Although HH_CF performs similarly to NSGAII on WFG2, WFG3, WFG4, and WFG7, HH_CF performs significantly slightly better than NSGAII on three instances (WFG2, WFG4 and WFG7) (see Tables 3 and4). For WFG8 and WFG9, HH_CF does not perform well compared to the others, except MOGA. HH_CF performs significantly worse than NSGAII and SPEA2 where HH_CF performs significantly better than MOGA as shown in Table 4.

We note from the above results that HH_CF performs worse than the low level heuristics when used in isolation with respect to the RNI metric, and it produces a low number of non-dominated solutions for most of the WFG problems. However, HH_CF performs very well and produces non-dominated solutions that are distributed uniformly well over the POF with respect to the UD metric when compared to the other methods. HH_CF also performs better than the others in most of the WFG problems and produces non-dominated solutions with high diversity that cover a larger proportion of objective space with respect to the SSC metric, except for WFG8 and WFG9 where it failed to converge towards the POF. As WFG8 and WFG9 has a significant bias feature, HH_CF may have difficulties coping with bias.

Generally, HH_CF produces competitive results across most of the WFG problems with respect to two performance metrics (SSC and UD) out of the three metrics. Although HH_CF obtains low number of solutions, it produces very good solutions in terms of diversity and convergence when compared to the low level heuristics when used in isolation. HH_CF can benefit from the strengths of the low level heuristics. Moreover, it has the ability to intelligently adapt to calling combinations of low level heuristics.

To understand how the HH_CF could obtain these results, we analyze the behavior of the low level heuristics in the next sub-section.

4.3. Behavior of Low Level Heuristics

We compute the average *heuristic utilization rate* which indicates how frequently a given low level heuristic is chosen and applied during the search process, across all runs, in order to see which low level heuristic is used more frequently. The results are presented in Fig. 5. The average heuristic utilization rate of NSGAII is at least 44% and is the highest among all low level heuristics for each problem, except WFG5 for which SPEA2 is chosen most frequently with a utilization rate of 55.72% during the search process. It explains why HH_CF has either a similar or relatively better convergence to the POF for most of the test problems when compared with NSGAII. It is indicates that NSGAII performs best among other low level heuristics on most of the WFG problems. The authors theorize that HH_CF, therefore, prefers NSGAII and it is chosen more frequently than the other low level heuristics. Our result is consistent with the result in (Bradstreet et al., 2007) that show the best performance is achieved by NSGAII on the WFG test functions with two objectives. The performance of MOGA is not that good on the WFG test, thus it is invoked relatively less frequently during the search process because of the diversification factor f_2, in equation 3. However, MOGA still influences the performance of HH_CF, negatively, in particular with respect to the RNI metric. This is due to that fact that MOGA does not have any archive mechanism or preserving strategy to maintain the non-dominated solutions during the search. The average utilization rate of MOGA is the highest for WFG8 (10.16%) and WFG9 (22.40%). This utilization rate explains why the performance of HH_CF is the worst performing approach in terms of RNI. HH_CF also faces some difficulty while solving WFG8 and WFG9 in terms of convergence.

In order to see the effectiveness of each chosen low level heuristic on the performance of HH_CF, we investigate the performance of the low level heuristics with respect to the RNI, SSC and UD metrics at twenty five decision points during the search process. We observe that some problems are following a specific pattern to invoke the low level heuristics during the search. Each problem has its own pattern. For example, for WFG3, NSGAII is invoked and executed for the first seven consecutive decision points. Then SPAE2 is invoked for the next four decision points,

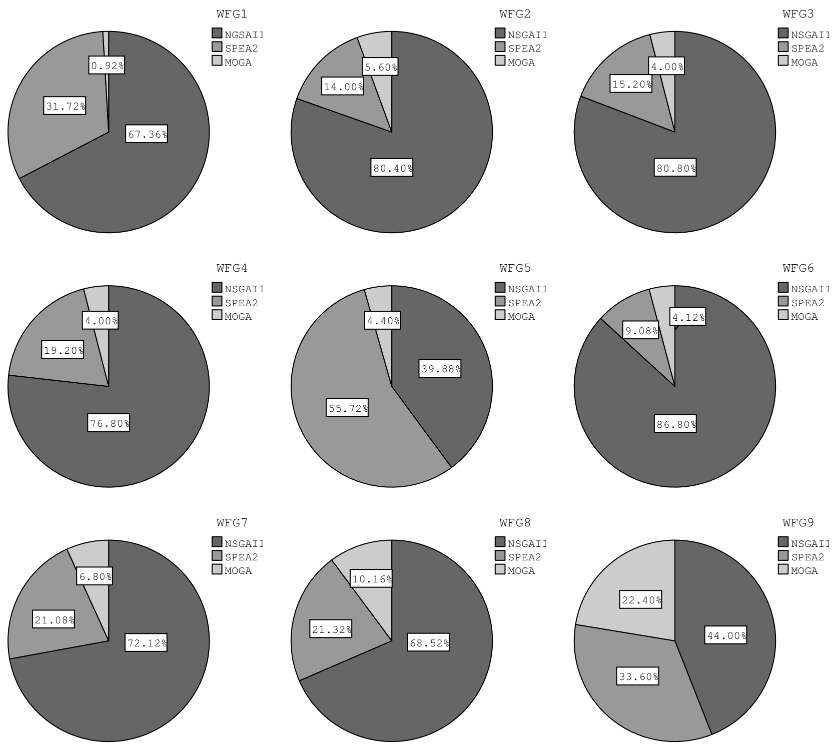


Figure 5: The average heuristic utilization rate for the low level heuristics (NSGAII, SPEA2 and MOGA) in HH_CF on the WFG test suite

followed by one iteration of MOGA. Then NSGAII is chosen for the rest of the search. More of these patterns are illustrated in Fig. 6.

In order to analyze these results, we divide the WFG instances into four categories based on the performance of HH_CF compared to the three low level heuristics being used in isolation with respect to RNI, SSC and UD as listed below:

1. WFG1,WFG5 and WFG6:
   - RNI: Better performance than MOGA and worse than NSGAII and SPEA2
   - SSC: The best performance among NSGAII, SPEA2 and MOGA
   - UD: The best performance among NSGAII, SPEA2 and MOGA
2. WFG2 and WFG3:

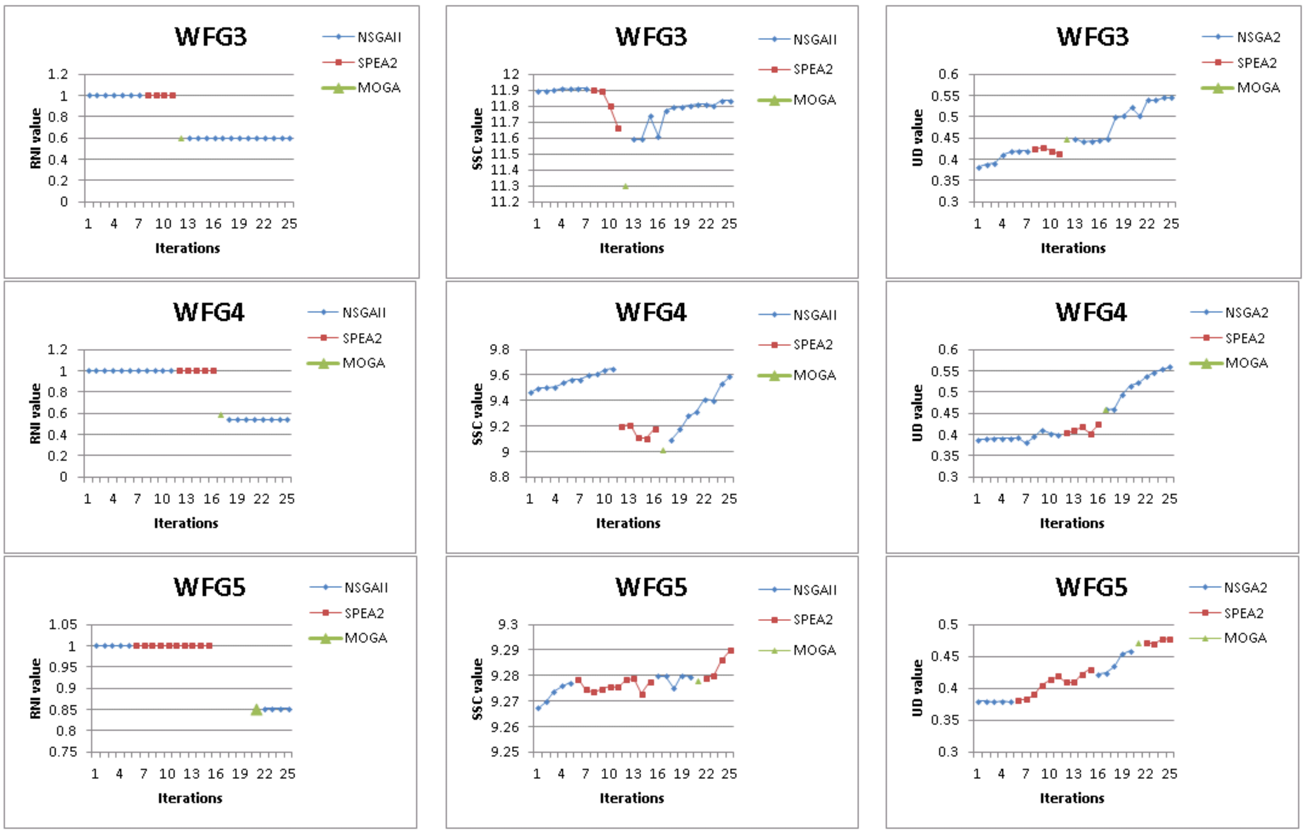


Figure 6: The average of RNI,SSC and UD values versus decision point steps plots across selected benchmark problems (the WFG3 ,WFG4 and WFG5). Each step in the plot is associated with the most frequently selected low level heuristics across 30 trials.

- RNI: Similar performance to MOGA and worse than NSGAII and SPEA2
- SSC: Better performance than SPEA2 and MOGA and similar to NSGAII
- UD: The best performance among NSGAII, SPEA2 and MOGA

3. WFG4 and WGF7:
    - RNI: The worst performance among NSGAII, SPEA2 and MOGA
    - SSC: Better performance than SPEA2 and MOGA and similar to NSGAII
    - UD: The best performance among NSGAII, SPEA2 and MOGA

4. WFG8 and WFG9:
    - RNI: The worst performance among NSGAII, SPEA2 and MOGA
    - SSC: The worst performance among NSGAII, SPEA2 and MOGA
    - UD: The best performance among NSGAII, SPEA2 and MOGA

For each category described above, except the last one, we have selected a sample problem to visualize the low level call patterns. WFG5 for the first category, WFG3 for the second category and WFG4 for the third category. For the last category, no specific pattern has been observed. The selected three problems have different problems features in terms of separability and modality Huband et al. (2006). The average of RNI, SSC and UD values versus decision point plots across selected benchmark problems (WFG3, WFG4 and WFG5) are shown in Fig. 6. Each step in the plot is associated with the most frequently selected low level heuristics across 30 trials. Since we employed All-Moves as an acceptance strategy, some moves are accepted even if it worsens the solution quality.

There is strong empirical evidence in the literature showing that the number of iterations is influential on the performance of a learning hyper-heuristic (Burke et al., 2013, 2012). Here, we observe from Fig. 6 that if the experiments were performed with shorter iterations, say a run was terminated at twelve instead of twenty five, then the algorithm would have ended up with worse SSC and UD values for the sample benchmark functions of WFG3, WFG4 and WFG5. It is clear from Fig. 6 that MOGA produces a worse solution with respect to RNI during the search and this solution is accepted influencing the performance of HH_CF. However, some worsening moves could still lead to better solutions at the end of the search process with respect to a certain metric, such as the performance of HH_CF with respect to the UD metric. SPEA2 produces low quality solutions in terms of the distribution along the POF, but this helps it to escape from the local optimum and obtain better solutions at the end. This is also true with respect to the SSC performance indicator. In addition, we note that HH_CF has an advantage over MOGA and outperforms the three MOEAs methods with respect to the distribution of non-dominated individuals over the Pareto optimal front. It also has an advantage over NSGAII in terms of convergence, in that it performs better than all other methods in some problems while performing better or similar to NSGAII on the other problems. However, HH_CF does not have an advantage over NSGAII and

SPEA2 with respect to the non-dominated individuals in the population. HH_CF performs poorly because of MOGA's effect. It worth noting that the fewer number of iterations is not sufficient for the learning heuristic selection method to distinguish the well performing low level heuristics from poorer ones. This is clear from Fig. 6 where increasing the number iterations improves the SSC and UD values for the selected problems.

It can be concluded that our choice function hyper-heuristic can benefit from the strengths of the low level heuristics. And it can avoid the weaknesses of them (partially), as the poor performance of MOGA affects the performance of HH_CF badly in the metric of RNI by producing low number of non-dominated solutions. We can avoid this by employing another acceptance move strategy instead of All-Moves. A non-deterministic acceptance strategy could accept worsening moves within a limited degree and help improve the quality of the solutions. However, HH_CF has the ability to intelligently adapt to calling combinations of low level heuristics.

5. Performance Comparison of HH_CF to the Other Multi-objective Approaches

The experiments are conducted to examine the performance of our proposed HH_CF compared to two other multi-objective approaches; a random hyper-heuristics (HH_RAND) and the adaptive multi-method search (AMALGAM) (Vrugt & Robinson, 2007). In a random hyper-heuristic (HH_RAND), we employ a simple random selection instead of the choice function selection that is used in HH_CF. No ranking scheme nor learning mechanism, is embedded into HH_RAND. In HH_RAND, we use the same three low level heuristics that are used in HH_CF. The hypervolume (SSC) (Zitzler & Thiele, 1999)and the generational distance (GD) (Van Veldhuizen & Lamont, 1998) metrics are used to compare the performance of the multi-objective hyper-heuristic for this set of experiments. The GD measures the distance (convergence) of the approximation non-dominated front to the POF. A smaller value of GD is more desirable and it indicates that the approximation non-dominated front is closer to the POF. We use *t*-test for the average performance comparison of algorithms and the results are discussed using the same notation as in Section 4.1.

In order to keep the computational costs of the experiments to an affordable level, all the methods were executed for 25,000 function evaluations with a population size of 100 and 250 generations in each run. Both

HH_CF and HH_RAND are executed for 2500 function evaluations at each iteration. Depending on the given problem, the execution time of HH_CF and HH_RAND for one run takes about 5-12 minutes. Other parameter settings of AMALGAM are identical to those used in (Vrugt & Robinson, 2007). We used the Matlab implementation of AMALGAM obtained from the authors via personal communication. We implemented a C++ interface between AMALGAM and the WFG test suite's C++ code. All other experimental settings are fixed the same in Section 4.1.

5.1. Results

The performance values of HH_CF and the other hyper-heuristics methods with respect to the performance metrics SSC and GD on the WFG problems are summarized in Table 5. For each performance metric, the average and standard deviation values are computed. These performance results with respect to SSC, GD are also displayed as box plots in Figs. 7 and 8 in order to provide a visualization of the distribution of the simulation data of the 30 independent runs. The statistical *t*-test comparing our proposed HH_CF and other multi-objective hyper-heuristics for the metrics (SSC and GD) are given in Table 6. The results show that the HH_CF performs better than the other algorithms in most cases. As expected, HH_CF achieves better coverage and diversity than HH_RAND according to two metrics. This is due to the learning mechanism that is used in HH_CF which adaptively guides the search towards the POF. Interestingly, HH_RAND performs better than AMALGAM according to the hypervolume metric except in WFG9. However, HH_RAND performs worse than AMALGAM according to the GD metric on all problems. This performance variation is statistically significant as illustrated in Table 6. HH_RAND performs significantly better than AMALGAM for the SSC metric on eight instances of WFG except in WFG9. HH_RAND also performs significantly better than AMALGAM for the GD metric on three instances (WFG1, WFG6 and WFG7) while it performs significantly similar to AMALGAM on one instance of WFG5 where it performs significantly worse than AMALGAM for the rest.

Compared to AMALGAM, HH_CF performs better with respect to the convergence and diversity for the most of the WFG problems. According to the SSC metric, HH_CF produced non-dominated solutions that covers a larger proportion of the objective space than AMALGAM on all

WFG problems except for WFG9. In Table 6, HH_CF performs significantly better than AMALGAM on eight instances of WFG except for WFG9 where AMALGAM performs significantly better than HH_CF on this instance. The superiority of HH_CF in SSC metric is due to the stronger selection mechanism and the effective ranking scheme that rely on choosing a heuristic with the best SSC value at the right time (decision point) to guide the search to move toward more spaces around the POF. This result is more reliable as shown in Fig. 7.

According to the metrics of GD, HH_CF is superior to AMALGAM on most of WFG problems as reported in Table 5 and displayed as box plots in Fig 8. In Table 6, HH_CF performs significantly better than AMALGAM on five instances out of nine including WFG1, WFG2, WFG5, WFG6, and WFG7 for the metric of GD. Again, this result is due to the online learning selection mechanism and the ranking scheme in HH_CF. The ranking scheme maintains the past performance of low level heuristics using a set of performance indicators that measure different aspects of the solutions. During the search process, the ranking scheme creates a balance between choosing the low level heuristics and their performance according to a particular metric. This balance enhances the algorithm performance to yield better solutions that converge toward the POF as well as distribute uniformly along the POF. However, AMALGAM performs significantly better than HH_CF on the other four instances for GD (see Tables 5 and 6). This might be because of the nature of the problems that present difficulties for HH_CF to converge toward the POF or might slow down the convergence speed such as the bias in WFG8,WFG9 and the multimodality of WFG4. It is good to report that AMALGAM has better performance according to the three metrics; SSC and GD in WFG9. This is shown in Table 6 that AMALGAM performs significantly better than others on one instance (WFG9).

For each problem, we computed the 50% attainment surface for each algorithm, from the 30 fronts after 25,000 evaluation functions. In Fig. 9 shows the POF and the 50% attainment surface of the algorithms. HH_CF shows good convergence and uniform distribution for most datasets. It seems clear that HH_CF has converged well on the POF in WFG1 and WFG2 when compared to other algorithms. Moreover, HH_CF produced solutions that covered larger proportions of the objective space compared to the other algorithms. AMALGAM has poor convergence on the most problems. It has fewer number of solutions with poor convergence on

WFG2. AMALGAM has no solutions over the middle-lower segments of the POF for WFG3, WFG5, WFG6, WFG7, and WFG8 and no solutions over the upper-middle segments of the POF for WFG4.

It can be concluded that all the above results demonstrate the effectiveness of HH_CF in terms of its ability to intelligently adapt to calling combinations of low level heuristics and outperforming other hyper-heuristics for multi-objective optimization.

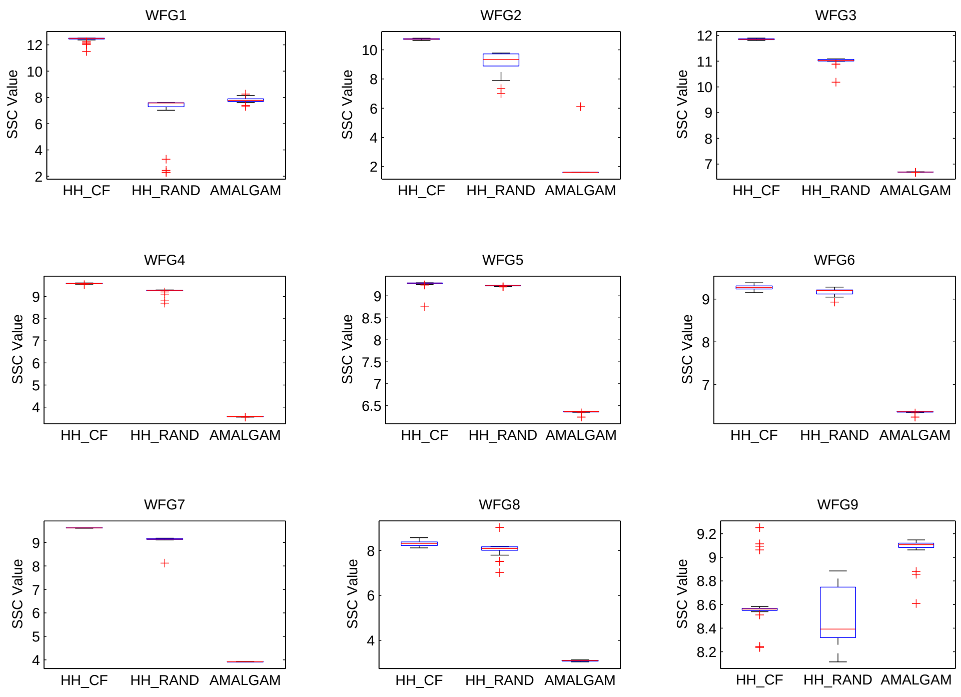


Figure 7: Box plots of HH_CF ,HH_RAND, and AMALGAM for the measure of hypervolume (SSC) on the WFG test functions

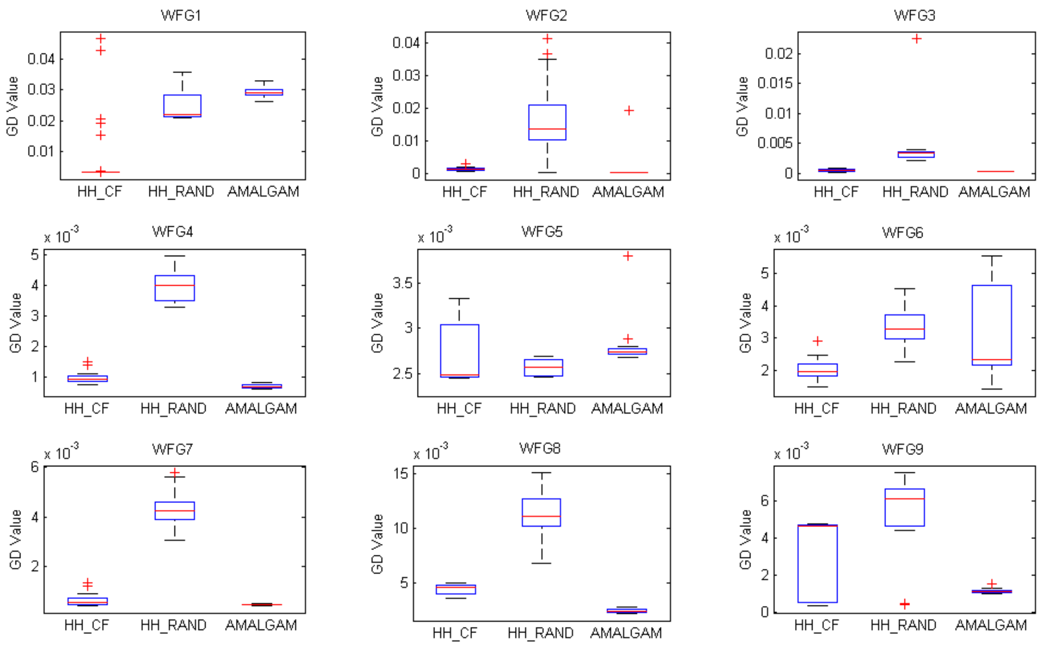


Figure 8: Box plots of HH_CF, HH_RAND, and AMALGAM for the measure of generational distance (GD) on the WFG test functions

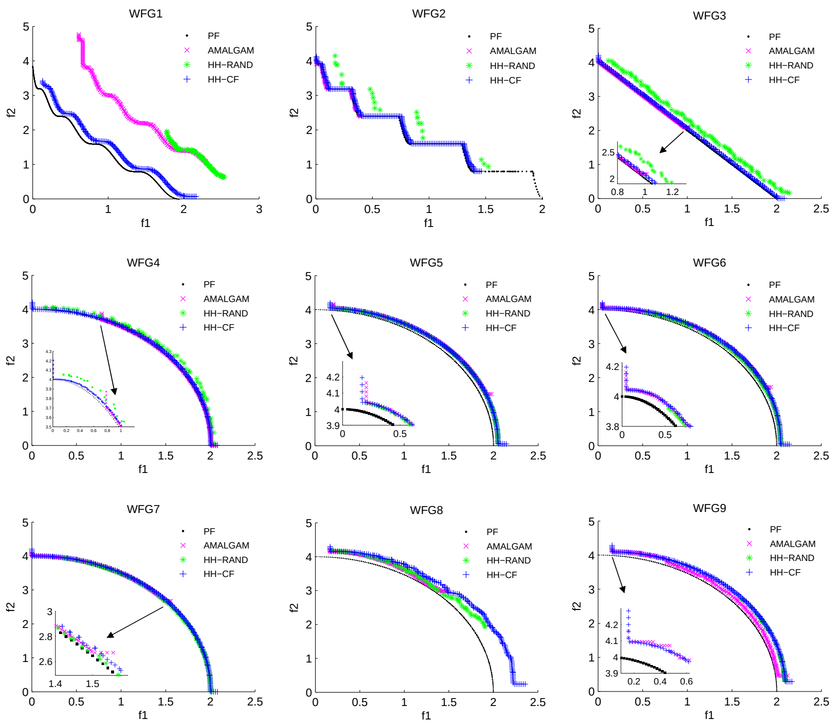


Figure 9: Pareto optimal front and 50% attainment surfaces for AMALGAM, HH_RAND and HH_CF on the WFG test functions

6. Performance HH_CF on the Multi-objective Design of Vehicle Crashworthiness

More experiments are conducted over a multi-objective real-world problem, namely the design of vehicle crashworthiness problem (Liao et al., 2008), to evaluate the performance of our choice function based hyper-heuristics (HH_CF). The same performance evaluation criteria and algorithms are used as described in the previous section. In addition, the performance of HH_CF is compared to NSGAII (Deb & Goel, 2001). The motivation behind applying HH_CF to this problem is to see its performance on a real-world problem and to measure the level of generality that can achieve.

6.1. Problem description and Formulation

In the automotive industry, crashworthiness is a very important issue when designing a vehicle. Crashworthiness design of real-world vehicles involves optimization of a number of objectives including the head, injury criterion, chest acceleration, chest deflection, etc. However, some of these objectives may be, and usually are, in conflict with each other, i.e. an improvement in one objective value leads to deterioration in the values of the other objectives.

Liao et al. (2008) presented a multi-objective design for the vehicle crashworthiness problem with three objectives considering the mass of the vehicle as the first design objective, while an integration of collision acceleration between $t_1 = 0.05s$ and $t_2 = 0.07s$ in the full frontal crash as the second objective function. The toe-board intrusion in the 40% offset-frontal crash is tackled as the third objective. The second and third objectives are constructed from the two crash conditions to reflect the extreme crashworthiness and formulated as quadratic basis functions while the vehicle mass is formulated as a linear function as follows:

$$\begin{aligned}\text{Mass} &= 1640.2823 + 2.3573285t_1 + 2.3220035t_2 \\ &+ 4.5688768t_3 + 7.7213633t_4 + 4.4559504t_5\end{aligned} \tag{4}$$

$$\begin{aligned}\text{Ain} &= 6.5856 + 1.15t_1 - 1.0427t_2 + 0.9738t_3 + 0.8364t_4 \\ &- 0.3695t_1t_4 + 0.0861t_1t_5 + 0.3628t_2t_4 - 0.1106t_1^2 \\ &- 0.3437t_3^2 + 0.1764t_4^2\end{aligned} \tag{5}$$

$$\begin{aligned}\text{Intrusion} &= -0.0551 + 0.0181t_1 + 0.1024t_2 + 0.0421t_3 \\ &- 0.0073t_1t_2 + 0.024t_2t_3 - 0.0118t_2t_4 - 0.0204t_3t_4 \\ &- 0.008t_3t_5 - 0.0241t_2^2 + 0.0109t_4^2\end{aligned} \tag{6}$$

So, the multi-objective design of vehicle crashworthiness problem is formulated as:

$$\begin{aligned}&\min \text{F(x)}= [Mass, Ain, Intrusion] \\ &\quad \text{s.t.} \\ &\qquad 1mm \leq x \leq 3mm \\ &\text{where x}=(t_1, t_2, t_3, t_4, t_5)^T\end{aligned} \tag{7}$$

6.2. *Experimental Settings*

We performed 30 independent runs for each comparison method using the same parameter settings as provided in (Liao et al., 2008) with a population size of 30 and running for 50 generations in each iteration. In order to make a fair comparison, we repeated NSGAII experiments conducted in (Liao et al., 2008) under the same environment. All methods were run for 75,000 function evaluations. The distance sharing σ for the UD metric and MOGA was arbitrarily set to 0.09 in the normalized space. These settings were used for the UD as a feedback indicator in the ranking scheme of the HH_CF and as a performance measure for the comparison. As the true Pareto front is unknown, we consider the best approximation found by means of combining results of all considered methods and used it instead of a true Pareto front for the metrics of GD. In the measure of SSC, the reference points in our experiments for k objectives can be set as $r_i = z^{nadir_i} + 0.5(z^{nadir_i} - z^{ideal_i})(0, i*2), i = 1, ..., k$ (Li & Landa-Silva, 2011). Other experimental settings are the same as those used in Section 4.1. All experiments are performed on an Intel Core2 Duo 3GHz/2G/250G computer.

6.3. Results

An initial set of experiments is performed to observe the performance variation of our approach with respect to the number of the decision points, denoted as NDP. NDP depends on the other parameters such as the number of function evaluations and the number of generations. During the initial experiments, the number of objective function evaluations is fixed as 1,500 for each stage which starts at each decision point. Each decision point is executed with the chosen MOEA for a fixed number of generations and population size, set to 50 and 30, respectively. HH_CF is tested using two different values for NDP; 25 and 50.

The performance of HH_CF for the different values of NDP with respect to the performance metrics (RNI, SSC, UD, GD) on the vehicle crashworthiness problem is summarized in Table 7. The average values across 30 trials are used. We observe that the choice of NDP does not have an influence on the performance of HH_CF based on RNI. The best SSC and UD average values are obtained when NDP is 50. However, HH_CF obtains the best average value for UD with an NDP of 25. Fifty decision points produces better solutions, hence these results are used for HH_CF while comparing its performance to the other approaches.

The mean performance comparison of HH_CF, HH_RAND, AMALGAM and NSGAII based on the performance metrics SSC and GD for solving the vehicle crashworthiness problem is provided in Table 8. The distribution of the simulation data of the 30 independent runs for the comparison methods with respect to these performance metrics are visualized in Fig. 10. For the metric of SSC, a higher value indicates a better performance while a lower value indicates a better performance for the metric of GD. Tables 8 and 9 show that HH_CF has the highest average value among HH_RAND and NSGAII with respect to the hypervolume (SSC), except AMALGAM where perform the best. With respect to the measures of GD, HH_CF is superior to all comparison methods. The performance difference of HH_CF from the other methods is statistically significant (see Table 9). HH_CF performs significantly better than NSGAII in both metrics. In addition, HH_CF achieves better coverage and diversity than HH_RAND according to both metrics. This results is consistent with the result in Section 4. The results demonstrate that effectiveness of the learning multi-objective hyper-heuristic approach when compared to one without a learning mechanism. This is understandable, as it has been observed that the learning mechanism successfully guides the search towards the POF. Interestingly, HH_RAND performs better than AMALGAM according to the GD metric. However, HH_RAND performs worse than AMALGAM according to the SSC metric.

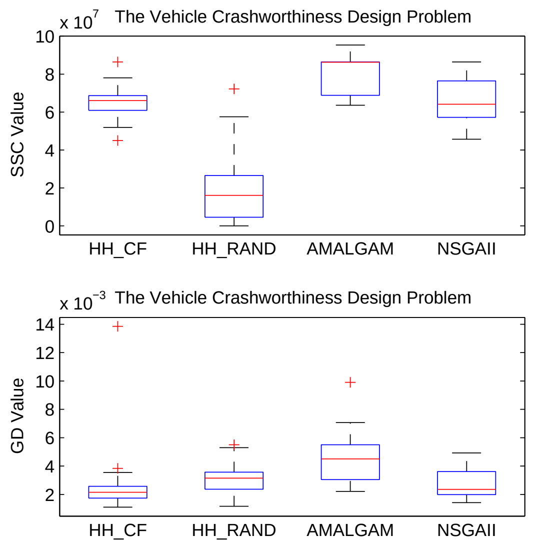


Figure 10: Box plots of HH_CF, HH_RAND, AMALGAM and NSGAII for the measure of hypervolume (SSC) and generational distance (GD) on the vehicle crashworthiness design problem

In summary, HH_CF performs the best considering convergence and diversity, producing better solutions that converge towards the POF compared to all comparison methods. Although HH_CF produces acceptable solutions with respect to the measure of hypervolume, it performs worse than AMALGAM for the same metric. Generally, the results demonstrate the potential of HH_CF for solving for solving this type of problem.

7. Remarks and Conclusion

studies on hyper-heuristics for multi-objective optimization are scarce. For the first time, a general selection hyper-heuristic framework for multi-objective optimization has been proposed. This framework is motivated by: (i) there is no existing algorithm which excels across all types of problems, and (ii) there is empirical evidence showing that hybridization or combining different (meta-)heuristics/algorithms could yield improved performance compared to (meta-)heuristics/algorithms run on their own. Hyper-heuristic frameworks, generally, impose a domain barrier which separates the hyper-heuristic from the domain implementation along with low level heuristics to provide a higher level of abstraction. The domain barrier does not allow any problem specific information to be passed to the hyper-heuristic during the search process. We designed our framework in this same modular manner. One of advantages of the proposed framework is its simplicity. The proposed framework is highly flexible and its components reusable. It is built on an interface which allows other researchers to write their own hyper-heuristic components easily. Even the low level heuristics can be easily changed if required. If new and better performing components are found in the future, the software can be easily modified to include those components for testing. A simple choice function, for the first time, is employed as a (high level heuristic) selection mechanism to deal with the multi-objective optimization problems. The choice function adaptively ranks the performance of three low-level meta-heuristics, deciding which one to call at each decision point. All-Moves is employed as an acceptance strategy, meaning that we accept the output of each low level heuristic whether it improves the quality of the solution or not. In our multi-objective hyper-heuristic framework, a learning process is an essential component for guiding the heuristic selection method while it decides on the most appropriate heuristic to apply at each step of the iterative approach. We employed four performance metrics (algorithm effort (AE), ratio of non-dominated individuals (RNI), size of space covered (SSC) and uniform distribution of a non-dominated population (UD) to act as an online learning mechanism to provide knowledge of the problem domain to the selection mechanism. These metrics do not require a prior knowledge of the POF, which means that our framework is suitable for tackling any given real-world problem in future.

The performance metrics are integrated into a ranking scheme that we introduced in this study. Our ranking scheme relies on sorting the low level heuristics in descending order based on the highest ranking among the other heuristics.

The ranking scheme is simple and flexible and enables us to incorporate any number of low level heuristics. Three well-known multi-objective evolutionary algorithms (NSGAII, SPEA2, and MOGA) are incorporated into the multi-objective choice function hyper-heuristic framework to act as the low level meta-heuristics. Although NSGAII, SPEA2 and MOGA are not considered state-of-the-art MOEA, they are still viewed as a baseline of MOEAs. They incorporate much of the known MOEA theory which make them applicable to investigate their hybridization within our multi-objective hyper-heuristic framework.

Our multi-objective choice function based hyper-heuristic (HH_CF) is tested over both benchmark test problems i.e the WFG test suite and real-world application i.e. the multi-objective design of vehicle crashworthiness with two and three objectives respectively. The experimental results demonstrate the effectiveness and potential of the proposed approach in solving continuous multi-objective optimization problems. HH_CF outperforms the low level heuristics, i.e. NSGAII, SPEA2 and MOGA, when used in isolation, and to two other multi-objective hyper-heuristics; a random hyper-heuristics HH_RAND and adaptive multi-method search AMALGAM on the WFG test suite. HH_CF performs well in terms of the distribution of non-dominated individuals along the POF and obtains competitive results in terms of converging towards the POF. Moreover, this observation further is supported by empirical evidence obtained from testing HH_CF against NSGAII, HH_RAND and AMALGAM over the multi-objective vehicle crashworthiness design problem as a real-world problem. HH_CF is superior to HH_RAND and NSGAII for solving this problem in terms of convergence and diversity. In addition, HH_CF outperforms AMALGAM according to the measures of generational distance. HH_CF still produces solutions with acceptable quality with respect to the metric of hypervolume (SSC). However, it could not perform better when compared to AMALGAM. This is could be due to the dimensionally of the problem, as the HH_CF beats AMALGAM in this metric over two objective problem, where the reverse is true with the three objective problem.

Generally, the results reported, in this study demonstrate the effectiveness of the learning multi-objective hyper-heuristic approach when compared to methodologies with no learning mechanism. This is understandable, as it has been observed that the learning mechanism successfully guides the search process towards the POF. Moreover, the experimental results show that our multi-objective choice function based hyper-heuristic can exploit and combine the strengths of multiple low level heuristics. The superiority of HH_CF is due to online learning heuristic selection mechanism and the effective ranking scheme. The ranking scheme maintains the past performance of low level heuristics using a set of performance indicators that measure different aspects of the solutions. During the search process, the ranking scheme creates a balance between choosing the low level heuristics and their performance according to a particular metric. This balance enhances the algorithm performance to yield better solutions that converge toward the POF as well as distribute uniformly along the POF. Unfortunately, HH_CF cannot avoid the weaknesses of the low level heuristics fully, as the poor performance of MOGA influences the performance of HH_CF with respect to the ratio of non-dominated individual (RNI) by causing the generation of lower numbers of non-dominated solutions as compared to NSGAII and SPEA2. We can overcome this by employing another acceptance move strategy instead of All-Moves. Future work needed to investigate the performance of our choice function based hyper-heuristic when employing the alternative move acceptance strategy, that accepts worsening moves within a limited degree and help improve the quality of the solutions. This process is not a trivial process. It requires elaboration of existing methods and their usefulness in a multi-objective setting. The framework in which HH_CF is used for managing a set of multi-objective meta-heuristics offers interesting potential research directions in multi-objective optimization. There is strong empirical evidence showing that different combinations of heuristic selection and acceptance methods in a selection hyper-heuristic framework yield different performances in single-objective optimization (Burke et al., 2013; Asta et al., 2013). More multi-objective optimizers and even heuristic selection can be adapted from previous research in single-objective optimization could incorporated with our multi-objective hyper-heuristic framework. The proposed framework tackled continuous multi-objective optimization problems. Our aim is to test the level of generality of our framework further over a wide number of multi-objective problems including combinatorial, discrete and dynamic problems.

Acknowledgement

The authors thank Shahriar Asta for his help and support.

References

Anderson, J. M., Sayers, T. M., & Bell, M. G. H. (2007). Optimisation of a fuzzy logic traffic signal controller by a multiobjective genetic algorithm. *IEEE Road Transport Information and Control*, *454*, 186–190.

de Armas, J., Miranda, G., & León, C. (2011). Hyperheuristic encoding scheme for multi-objective guillotine cutting problems. In *Proceedings of the 13th Annual Genetic and Evolutionary Computation Conference* (pp. 1683–1690).

Asta, S., Karapetyan, D., Kheiri, A., Özcan, E., & Parkes, A. (2013). Combining monte-carlo and hyper-heuristic methods for the multi-mode resource-constrained multi-project scheduling problem. *Journal of Scheduling, in review*, .

Bäck, T. (1996). *Evolutionary Algorithms in Theory and Practice*. Oxford University Press.

Bai, R., van Woensel, T., Kendall, G., & Burke, E. K. (2013). A new model and a hyper-heuristic approach for two-dimensional shelf space allocation. *Journal Operation Research*, *11*, 31–55.

Bradstreet, L., Barone, L., While, L., Huband, S., & Hingston, P. (2007). Use of the wfg toolkit and pisa for comparison of multi-objective evolutionary algorithms. In *Proceedings of IEEE Symposium on Computational Intelligence in Multi-criteria Decision-making* (pp. 382–389).

Burke, E., Kendall, G., Newall, J., Hart, E., Ross, P., & Schulenburg, S. (2003a). Handbook of meta-heuristics. chapter Hyper-Heuristics: An Emerging Direction in Modern Search Technology. (pp. 457–474). Kluwer Academic Publishers.

Burke, E., Landa-Silva, D., & Soubeiga, E. (2003b). Multi-objective hyper-heuristic approaches for space allocation and timetabling. In *MIC 2003-Meta-heuristics: Progress as Real Problem Solvers* (pp. 129–158).

Burke, E. K., Gendreau, M., Hyde, M., Kendall, G., Ochoa, G., Özcan, E., & Qu, R. (2013). Hyper-heuristics: A survey of the state of the art. *Journal of the Operational Research Society*, *64*, 1695–1724.

Burke, E. K., Kendall, G., Misir, M., & Özcan, E. (2012). Monte carlo hyper-heuristics for examination timetabling. *Annals of Operations Research, 196*, 73–90.

Chow, J., & Regan, A. (2012). A surrogate-based multiobjective meta-heuristic and network degradation simulation model for for robust toll pricing. Civil Engineering Working Papers.

Coello, C. C., & Pulido, G. (2001). Multiobjective optimization using a micro-genetic algoritm. In *Proceedings of Genetic and Evolutionary Computation Conference.* (pp. 274–282).

Coello, C. C., Veldhuizen, D. V., & Lamont, G. (2007). *Evolutionary Algorithms for Solving Multi-Objective Problems*. Kluwer Academic Publishers.

Cowling, P., Kendall, G., & Soubeiga, E. (2002). A hyper-heuristic approach to scheduling a sales summit. In *Proceedings of 3rd International Conference Practice and Theory of Automated Timetabling(PATAT 2000)*.

Davidor, Y. (1991). *Epistasis variance; A viewpoint on GA-hardness*. Rawlins.

Deb, K. (1999). Multi-objective genetic algorithms: Problem difficulties and construction of test problem. *Evolutionary Computation, 7*, 205–230.

Deb, K. (2001). *Multi-Objective Optimization Using Evolutionary Algorithms*. Wiley.

Deb, K. (2005). Introductory tutorials in optimization and decision support methodologies. chapter Multi-objective optimization. Search Methodologies. (pp. 273–316). Springer.

Deb, K., & Agrawal, R. B. (1995). Simulated binary crossover for continuous search space. *Complex Systems, 9*, 115–148.

Deb, K., & Goel, T. (2001). Controlled elitist nondominated sorting genetic algorithms for better convergence. In *Proceedings of Evolution Multi Criterion Optimization Conference* (pp. 67–81).

Deb, K., & Goldberg, D. (1989). An investigation on niche and species formation in genetic function optimization. In *Proceedings of 3rd International Conference on Genetic Algorithms* (pp. 42–50).

Deb, K., Thiele, L., Laumanns, M., & Zitzler, E. (2002). Scalable multi-objective optimization test problems. In *Proceedings of IEEE congress on evolutionary computation.* (pp. 825–830).

Fonseca, C., & Fleming, P. (1998). Multiobjective optimization and multiple constraint handling with evolutionary algorithms. *IEEE Transactions on Systems, Man, and Cybernetics.-Part A: Systems and Humans*, *28*, 26–37.

Fonseca, C. M., & Fleming, P. J. (1993). Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. In *Proceedings of Genetic Algorithms Conference* (pp. 416–423).

Furtuna, R., Curteanu, S., & Leon, F. (2012). Multi-objective optimization of a stacked neural network using an evolutionary hyper-heuristic. *Applied Soft Computing*, *12*.

Gibbs, J., Kendall, G., & Özcan, E. (2011). Scheduling english football fixtures over the holiday period using hyper-heuristics. In *Proceedings of the 11th International Conference on Parallel Problem Solving From Nature* (pp. 496–505). volume 6238.

Goldberg, D. E. (1987). Genetic algorithms and simulated annealing. chapter Simple genetic algorithms and the minimal, deceptive problem. (pp. 74–88). Morgan Kaufmann.

Gomez, J., & Terashima-Marín, H. (2010). Approximating multi-objective hyper-heuristics for solving 2d irregular cutting stock problems. In *Advances in Soft Computing* Lecture Notes in Computer Science (pp. 349–360).

Horn, J., & Goldberg, D. E. (1995). Genetic algorithm difficulty and the modality of fitness landscape. In *Proceedings of the 3rd Workshop on Foundation of Genetic Algorithms* (pp. 243–270).

Huband, S., Hingston, P., Barone, L., & While, L. (2006). A review of multiobjective test problems and a scalable test problem toolkit. *IEEE Transactions on Evolutionary Computation*, *10*, 477–506.

Hussin, N. (2005). *Tabu Search Based Hyper-heuristic Approaches for Examination Timetabling*. Ph.D. thesis The University of Nottingham, Nottingham, UK.

Jaszkiewicz, A. (2001). Comparison of local search-based metaheuristics on the multiple objective knapsack problem. *Foundations of Computing and Decision Sciences*, *26*, 99–120.

Kargupta, H. (1995). Signal-to-noise, crosstalk, and long range problem difficulty in genetic algorithms. In *Proceedings of the 6th Internationl Conference on Genetic Algorithms* (pp. 193–200).

Kendall, G., Cowling, P., & Soubeiga, E. (2002). Choice function and random hyperheuristics. In *Proceedings of the 4th Asia- Pacific Conference on Simulated Evolution And Learning* (pp. 667–671).

Khare, V., Yao, X., & Deb, K. (2003). Performance scaling of multi-objective evolutionary algorithms. In *Proceedings of 2nd International Conference on Evolutionary Multi-Criterion Optimization* (pp. 376–390).

Kumari, A., Srinivas, K., & Gupta, M. (2013). Software module clustering using a hyper-heuristic based multi-objective genetic algorithm. In *Advance Computing Conference (IACC), 2013 IEEE 3rd International* (pp. 813–818).

Landa-Silva, J. D. (2003). *Metaheuristic and Multiobjective Approaches for Space Allocation*. Ph.D. thesis University of Nottingham, UK.

Len, C., Miranda, G., & Segura, C. (2009). Hyperheuristics for a dynamic-mapped multi-objective island-based model. In *Distributed Computing, Artificial Intelligence, Bioinformatics, Soft Computing, and Ambient Assisted Living* (pp. 41–49). Springer Berlin Heidelberg volume 5518 of *Lecture Notes in Computer Science*.

Li, H., & Landa-Silva, D. (2011). An adaptive evolutionary multi-objective approach based on simulated annealing. *Evolutionary Computation*, *19*, 561–595.

Li, H., & Zhang, Q. (2009). Multiobjective optimization problems with complicated pareto sets, moea/d and nsga-ii. *IEEE Transactions on Evolutionary Computation*, *13*, 284–302.

Liao, X., Li, Q., Yang, X., Zhang, W., & Li, W. (2008). Multiobjective optimization for crash safety design of vehicles using stepwise regression model. *Struct Multidisc Optim*, *35*, 561–569.

Liu, D., Tan, K., Goh, C., & Ho, W. (2007). A multiobjective memetic algorithm based on particle swarm optimization. *IEEE Trans Syst Man Cybernet Part B Cybernet*, *37*, 42–50.

McClymont, K., & Keedwell, E. C. (2011). Markov chain hyperheuristic (mchh): an online selective hyper-heuristic for multiobjective continuous problems. In *Proceedings of GECCO11* (pp. 2003–2010).

Miranda, G., de Armas, J., Segura, C., & León, C. (2010). Hyperheuristic codification for the multi-objective 2d guillotine strip packing problem. In *Proceedings of IEEE Congress on Evolutionary Computation* (pp. 1–8).

Özcan, E., Bilgin, B., & Korkmaz, E. (2008). A comprehensive analysis of hyper-heuristics. *Intelligent Data Analysis*, *12*, 3–23.

Qu, R., & Burke, E. (2009). Hybridisations within a graph based hyper-heuristic framework for university timetabling problems. *Journal of the Operational Research Society*, *60*, 1273–1285.

Raad, D., Sinkse, A., & Vuuren, J. (2010). Multiobjective optimization for water distribution systemdesign using a hyperheuristic. *Journal of Water Resources Management*, *136*, 592–596.

Rafique, A. F. (2012). Multiobjective hyper heuristic scheme for system design and optimization. In *Proceedings of 9th International Conference on Mathematical Problems in Engineering, Aerospace and Science,AIP Conference 1493*. volume 764.

Srinivas, N., & Deb, K. (1994). Multiobjective optimization using non-dominated sorting in genetic algorithms. *Evolutionary Computation*, *2*, 221–248.

Tan, K. C., Lee, T. H., & Khor, E. F. (2002). Evolutionary algorithms for multi-objective optimization: Performance assessments and comparisons. *Artificial Intelligence Review*, *17*, 253–290.

Van Veldhuizen, D. A., & Lamont, G. B. (1998). Evolutionary computation and convergence to a pareto front. In *In Late Breaking Papers at the Genetic Programming 1998 Conference* (pp. 221–228).

Vázquez-Rodríguez, J., & Petrovic, S. (2013). A mixture experiments multi-objective hyper-heuristics. *Journal of the Operational Research Society*, *64*, 1664–1675.

Veerapen, N., Landa-Silva, D., & Gandibleux, X. (2009). Hyperheuristic as component of a multi-objective metaheuristic. In *Proceedings of the Doctoral Symposium on Engineering Stochastic Local Search Algorithms (SLS-DS 2009)*.

Veldhuizen, D. V., & Lamont, G. (2000). Multiobjective evolutionary algorithms: Analyzing the state-of-the-art. *Evolutionary Computation*, *8*, 125–147.

Voutchkov, I., & Keane, A. (2010). Computational intelligence in optimization: Adaptation, learning, and optimization. chapter Multi-objective optimization using surrogates. (pp. 155–175).

Vrugt, J., & Robinson, B. (2007). Improved evolutionary optimization from genetically adaptive multimethod search. *Proceedings of the National Academy of Sciences*, *104*, 708–711.

Wang, Y., & Li, B. (2010). Multi-strategy ensemble evolutionary optimization for dynamic multi-objective optimization. *Memetic Comput*, *2*, 3–24.

Watanabe, S., Hiroyasu, T., & Miki, M. (2002). Lcga: Local cultivation genetic algorithm for multi-objective optimization problem. In *Proceedings of 2002 Genetic and Evolutionary Computation Conference*.

Whitley, L. D. (1991). Foundations of genetic algorithms. chapter Fundamental principles of deception in genetic search. (pp. 221–241). Morgan Kaufmann.

Zhang, Q., & Li, H. (2007). Evolutionary algorithms for multi-objective optimization: Performance assessments and comparisons. *IEEE Transactions on Evolutionary Computation*, *11*, 712–731.

Zhang, X., Srinivasan, R., & Liew, M. V. (2010). On the use of multi-algorithm, genetically adaptive multi-objective method for multi-site calibration of the swat model. *Hydrological Processes*, *24*, 955–1094.

Zhou, A., Qu, B.-Y., Li, H., Zhao, S.-Z., Suganthan, P. N., & Zhang, Q. (2011). Multiobjective evolutionary algorithms: A survey of the state of the art. *Swarm and Evolutionary Computation*, *1*, 32 – 49.

Zitzler, E., Deb, K., & Thiele, L. (2000). Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation*, *8*, 173–195.

Zitzler, E., & Künzli, S. (2004). Indicator-based selection in multiobjective search. In *Lecture Notes in Computer Science* Parallel Problem Solving from Nature (PPSN VIII) (p. 832842). Springer.

Zitzler, E., Laumanns, M., & Thiele, L. (2001). Spea2: Improving the strength pareto evolutionary algorithm for multi-objective optimization. In *EUROGEN 2001-Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problem* (pp. 95–100).

Zitzler, E., & Thiele, L. (1999). Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach. *IEEE Transactions on Evolutionary Computation*, *3*, 253–290.

Table 2: The WFG Test Functions

WFG1	$h_{M=1} : M = convex_m$ $h_M = mixed_M \; (with \; \alpha = 1 \; and \; A = 5)$ $t^1_{i=1:k} = y_i$ $t^1_{i=k+1:n} = S\_linear(y_i, 0.35)$ $t^2_{i=1:k} = y_i$ $t^2_{i=k+1:n} = b\_flat(y_i, 0.8, 0.75, 0.85)$ $t^3_{i=1:n} = b\_poly(y_i, 0.02)$ $t^4_{i=1:M-1} = r\_sum(\{y_{(i-1)k/(M-1)} + 1, ..., y_{ik/(M-1)}\}, \{2((i - 1k/(M-1, ..., 2ik/(M-1)\})$ $t^4_M = r\_sum(\{y_{k+1}, ..., y_n\}, \{2(k+1), ..., 2n\})$
WFG2	$h_{M=1} : M = convex_m$ $h_M = disc_M \; (with \; \alpha = \; \beta = 1 \; and \; A = 5)$ $As \; t^1 \; from \; WFG1. \; (Linear \; shift.)$ $t^2_{i=1:k} = y_i$ $t^2_{i=k+1:k+l/2} = r\_nonsep(\{y_{k+2(i-k)-1}, y_{k+2(i-k)}\}$ $, 2)$ $t^3_{i=1:M-1} = r\_sum(\{y_{(i-1)k/(M-1)} + 1, ..., y_{ik/(M-1)}\}, \{1, ..., 1\})$ $t^3_M = r\_sum(\{y_{k+}, ..., y_{k+l/2}\}, \{1, ..., 1\})$
WFG3	$h_{M=1} : M = linear_m(degenerate)$ $As \; t^{1:3} \; from \; WFG2.(Linear \; shift, non \; separable$ $reduction$ $and \; weighted \; sum \; reduction.)$
WFG4	$h_{M=1} : M = concave_m$ $t^1_{i=1:n} = S\_multi(y_i, 30, 10, 0.35)$ $t_(i = 1 : M-1)^2 = r\_sum(\{y_{(i-1)k/(M-1)} + 1, ..., y_{ik/(M-1)}\}, \{1, ..., 1\})$ $t^2_M = r\_sum(\{y_{k+1}, ..., y_n\}, \{1, ..., 1\})$
WFG5	$h_{M=1} : M = concave_m$ $t^1_{i=1:n} = S\_decept(y_i, 0.35, 0.001, 0.05)$ $As \; t^2 \; from \; WFG4. \; (weighted \; sum \; reduction.)$
WFG6	$h_{M=1} : M = concave_m$ $As \; t^1 \; from \; WFG1. \; (Linear \; shift.)$ $t^2_{i=1:M-1} = r\_nonsep(\{y_{(i-1)k/(M-1)} + 1, ..., y_{ik/(M-1)}\}, k/(M-1))$ $t^2_M = r\_nonsep(\{y_{k+1}, ..., y_n\}, l)$
WFG7	$h_{M=1} : M = concave_m$ $t^2_{i=1:k} = b\_param(y_i, r\_sum(\{y_{(i-1)}, ..., y_n\}, \{1, ...$ $, 1\}), 0.98/49.98, 0.02, 50)$ $t^2_{i=k+1:n} = y_i$ $As \; t^1 \; from \; WFG1. \; (Linear \; shift.)$ $As \; t^2 \; from \; WFG4. \; (weighted \; sum \; reduction.)$
WFG8	$h_{M=1} : M = concave_m$ $t^1_{i=1:k} = y_i$ $t^1_{i=k+1:n} = b\_param(y_i, r\_sum(\{y_1, ..., y_{i-1}\}, \{1, ...$ $, 1\}), 0.98/49.98, 0.02, 50)$ $Ast^1 fromWFG1.(Linearshift.)$ $As \; t^2 \; from \; WFG4. \; (weighted \; sum \; reduction.)$
WFG9	$h_{M=1} : M = concave_m$ $t^1_{i=1:n-1} = b\_param(y_i, r\_sum(\{y_{i+1}, ..., y_n\}, \{1, ...$ $, 1\}, 0.98/49.98, 0.02, 50)$ $t^1_n = y_n$ $t^2_{i=1:k} = S\_decept(y_i, 0.35, 0.001, 0.05)$ $t^2_{i=k+1:n} = S\_multi(y_i, 30, 95, 0.35)$ $As \; t^2 \; from \; WFG6. \; (non \; separable \; reduction.)$

Table 3: The average performance of HH_CF compered to the low level heuristics on the WFG test problems with respect to the ratio of non-dominated individuals (RNI), the hypervolume (SSC) and the uniform distribution (UD)

		RNI		SSC		UD	
	Methods	AVG	STD	AVG	STD	AVG	STD
WFG1	NSGAII	**1.0000**	0.0000	11.6041	0.3880	0.4003	0.0140
	SPEA2	**1.0000**	0.0000	6.4931	0.0066	0.4099	0.0148
	MOGA	0.2650	0.1140	4.2184	0.6727	0.2117	0.0478
	HH_CF	0.8800	0.2539	**12.1386**	0.9101	**0.4428**	0.1007
WFG2	NSGAII	**1.0000**	0.0000	10.8199	0.0041	0.3747	0.0112
	SPEA2	**1.0000**	0.0000	10.7898	0.7935	0.2874	0.0305
	MOGA	**1.0000**	0.0000	9.7959	0.6978	0.5414	0.0597
	HH_CF	0.2293	0.0545	**11.0219**	0.3042	**0.7278**	0.0661
WFG3	NSGAII	**1.0000**	0.0000	**11.9185**	0.0063	0.4244	0.0120
	SPEA2	**1.0000**	0.0000	11.4062	0.0189	0.4289	0.0078
	MOGA	0.6070	0.0400	11.2921	0.1393	0.4468	0.0324
	HH_CF	0.6027	0.0445	11.8940	0.0853	**0.5450**	0.0289
WFG4	NSGAII	**1.0000**	0.0000	9.6460	0.0041	0.4132	0.0151
	SPEA2	**1.0000**	0.0000	9.1853	0.0133	0.4058	0.0133
	MOGA	0.5800	0.0540	8.9968	0.2056	0.4594	0.0387
	HH_CF	0.5443	0.0452	**9.6588**	0.0176	**0.5596**	0.0361
WFG5	NSGAII	**1.0000**	0.0000	9.2857	0.0043	0.3958	0.0129
	SPEA2	**1.0000**	0.0000	9.2860	0.0214	0.4360	0.0087
	MOGA	0.6820	0.0360	8.8946	0.4171	0.4184	0.0272
	HH_CF	0.8537	0.1723	**9.2899**	0.5744	**0.4779**	0.0468
WFG6	NSGAII	**1.0000**	0.0000	9.3503	0.0605	0.4082	0.0247
	SPEA2	**1.0000**	0.0000	8.7135	0.1851	0.3761	0.0158
	MOGA	0.4990	0.0420	8.8878	0.1345	0.4786	0.0367
	HH_CF	0.4720	0.0412	**9.3687**	0.0542	**0.5962**	0.0363
WFG7	NSGAII	**1.0000**	0.0000	9.6579	0.0294	0.4048	0.0117
	SPEA2	**1.0000**	0.0000	9.2481	0.0161	0.4082	0.0116
	MOGA	0.6300	0.0550	9.1685	0.1799	0.4331	0.0415
	HH_CF	0.6173	0.0653	**9.6606**	0.0926	**0.5289**	0.0416
WFG8	NSGAII	**1.0000**	0.0000	**8.7155**	0.0140	0.4178	0.0123
	SPEA2	**1.0000**	0.0000	8.3957	0.0199	0.4069	0.0083
	MOGA	0.4790	0.0460	8.0762	0.2777	0.4490	0.0450
	HH_CF	0.2627	0.0454	8.3033	0.1224	**0.7886**	0.1245
WFG9	NSGAII	**1.0000**	0.0000	**8.7650**	0.2960	0.3955	0.0163
	SPEA2	**1.0000**	0.0000	8.7091	0.1967	0.4303	0.0106
	MOGA	0.8260	0.0900	8.5723	0.2259	0.3693	0.0350
	HH_CF	0.6410	0.0896	8.6132	0.2236	**0.5142**	0.0525

Table 4: The *t*-test results of HH_CF and low level heurstics on the WFG test problems with respect to the ratio of non-dominated individuals (RNI), the hypervolume (SSC) and the uniform distribution (UD)

Problem	Methods	Metrics		
		RNI	SSC	UD
WFG1	HH_CF-NSGAII	-	+	+
	HH_CF-SPEA2	-	+	+
	HH_CF-MOGA	+	+	+
	NSGAII-SPEA2	n/a	+	-
	NSGAII-MOGA	+	+	+
	SPEA2-MOGA	+	+	+
WFG2	HH_CF-NSGAII	-	∼	+
	HH_CF-SPEA2	-	+	+
	HH_CF-MOGA	∼	+	+
	NSGAII-SPEA2	n/a	∼	+
	NSGAII-MOGA	+	+	-
	SPEA2-MOGA	+	+	-
WFG3	HH_CF-NSGAII	-	∼	+
	HH_CF-SPEA2	-	+	+
	HH_CF-MOGA	∼	+	+
	NSGAII-SPEA2	n/a	+	-
	NSGAII-MOGA	+	+	-
	SPEA2-MOGA	+	+	-
WFG4	HH_CF-NSGAII	-	∼	+
	HH_CF-SPEA2	-	+	+
	HH_CF-MOGA	-	+	+
	NSGAII-SPEA2	n/a	+	+
	NSGAII-MOGA	+	+	-
	SPEA2-MOGA	+	+	-
WFG5	HH_CF-NSGAII	-	+	+
	HH_CF-SPEA2	-	+	+
	HH_CF-MOGA	+	+	+
	NSGAII-SPEA2	n/a	+	-
	NSGAII-MOGA	+	+	-
	SPEA2-MOGA	+	+	+
WFG6	HH_CF-NSGAII	-	+	+
	HH_CF-SPEA2	-	+	+
	HH_CF-MOGA	∼	+	+
	NSGAII-SPEA2	n/a	+	+
	NSGAII-MOGA	+	+	-
	SPEA2-MOGA	+	-	-
WFG7	HH_CF-NSGAII	-	∼	+
	HH_CF-SPEA2	-	+	+
	HH_CF-MOGA	∼	-	+
	NSGAII-SPEA2	n/a	+	∼
	NSGAII-MOGA	+	+	-
	SPEA2-MOGA	+	+	-
WFG8	HH_CF-NSGAII	-	-	+
	HH_CF-SPEA2	-	-	+
	HH_CF-MOGA	-	+	+
	NSGAII-SPEA2	n/a	+	-
	NSGAII-MOGA	+	+	-
	SPEA2-MOGA	+	+	-
WFG9	HH_CF-NSGAII	-	-	+
	HH_CF-SPEA2	-	-	+
	HH_CF-MOGA	-	+	+
	NSGAII-SPEA2	n/a	+	-
	NSGAII-MOGA	+	+	+
	SPEA2-MOGA	+	+	+

Table 5: The performance of HH_CF compared to multi-objective hyper-heuristics on the WFG test problems with respect to the Hypervolume (SSC) and the generational distance (GD)

		SSC		GD	
	Methods	AVG	STD	AVG	STD
WFG1	HH_CF	**12.0044**	0.8301	**0.00774**	0.0111
	HH_RAND	7.0258	0.7877	0.0242	0.0014
	AMALGAM	7.7902	0.1941	0.0292	0.0016
WFG2	HH_CF	**11.0102**	0.2033	**0.00046**	0.0005
	HH_RAND	9.7547	0.5078	0.0168	0.0109
	AMALGAM	1.7582	0.821	0.001	0.0035
WFG3	HH_CF	**11.7550**	0.0743	0.0007	0.0005
	HH_RAND	11.029	0.149	0.0038	0.0036
	AMALGAM	6.689	0.0049	**0.00036**	0
WFG4	HH_CF	**9.5610**	0.0143	0.001	0.0002
	HH_RAND	9.2052	0.0145	0.0041	0.0005
	AMALGAM	3.5687	0.0075	**0.00081**	0.0001
WFG5	HH_CF	**9.2701**	0.5343	**0.00253**	0.0003
	HH_RAND	9.2577	0.0556	0.0028	0.0001
	AMALGAM	6.3554	0.0323	0.0028	0.0003
WFG6	HH_CF	**9.3579**	0.053	**0.00225**	0.0006
	HH_RAND	9.3119	0.0501	0.0033	0.0005
	AMALGAM	6.3554	0.0323	0.003	0.0012
WFG7	HH_CF	**9.6498**	0.0901	**0.00047**	0.0003
	HH_RAND	9.1184	0.3473	0.0043	0.0007
	AMALGAM	3.9171	0.0035	0.0007	0.0000
WFG8	HH_CF	**8.2843**	0.1451	0.0044	0.0004
	HH_RAND	8.1089	0.3867	0.0114	0.0021
	AMALGAM	3.0945	0.0213	**0.00241**	0.0002
WFG9	HH_CF	8.5981	0.2143	0.0053	0.0015
	HH_RAND	8.4697	0.3059	0.006	0.0017
	AMALGAM	**9.0676**	0.114	**0.00113**	0.0001

Table 6: The t-test results of HH_CF,HH_RAND and AMALGAM on the WFG test problems with respect to the hypervolume (SSC) and generational distance(GD)

Problem	Methods	Metrics	
		SSC	GD
WFG1	HH_CF-HH_RAND	+	+
	HH_CF-AMALGAM	+	+
	HH_RAND-AMALGAM	+	+
WFG2	HH_CF-HH_RAND	+	+
	HH_CF-AMALGAM	+	+
	HH_RAND-AMALGAM	+	-
WFG3	HH_CF-HH_RAND	+	+
	HH_CF-AMALGAM	+	-
	HH_RAND-AMALGAM	+	-
WFG4	HH_CF-HH_RAND	+	+
	HH_CF-AMALGAM	+	-
	HH_RAND-AMALGAM	+	-
WFG5	HH_CF-HH_RAND	+	+
	HH_CF-AMALGAM	+	+
	HH_RAND-AMALGAM	+	$\sim$
WFG6	HH_CF-HH_RAND	+	+
	HH_CF-AMALGAM	+	+
	HH_RAND-AMALGAM	+	+
WFG7	HH_CF-HH_RAND	+	+
	HH_CF-AMALGAM	+	+
	HH_RAND-AMALGAM	+	+
WFG8	HH_CF-HH_RAND	+	+
	HH_CF-AMALGAM	+	-
	HH_RAND-AMALGAM	+	-
WFG9	HH_CF-HH_RAND	+	+
	HH_CF-AMALGAM	-	-
	HH_RAND-AMALGAM	-	-

Table 7: The performance of HH_CF with different sizes of decision points on the multi-objective design of vehicle crashworthiness problem with respect to the metrics of ratio of non-dominated individuals (RNI), size of space covered (SSC),uniform distribution (UD) of non-dominated population and generational distance(GD). Bold values indicate the best NDP choice for each give metric.

Decision Points	RNI		SSC		UD		GD	
	AVG	STD	AVG	STD	AVG	STD	AVG	STD
25	**1.00**	0.000	6.045E+07	1.669E+07	**0.623**	0.044	3.41E-03	8.39E-04
50	**1.00**	0.000	**6.631E+07**	1.979E+07	0.480	0.140	**2.59E-03**	2.23E-03

Table 8: The performance of multi-objective hyper-heuristics and NSGAII on the vehicle crashworthiness design problem with respect to the Hypervolume (SSC) and the generational distance (GD)

	SSC		GD	
Methods	AVG	STD	AVG	STD
HH_CF	6.631E+07	1.979E+07	**2.59E-03**	2.23E-03
HH_RAND	2.062E+07	1.903E+07	3.06E-03	1.26E-03
AMALGAM	**8.122E+07**	1.029E+07	4.52E-03	1.65E-03
NSGAII	6.618E+07	1.336E+07	2.76E-03	1.01E-03

Table 9: The *t*-test results of multi-objective hyper-heuristics and NSGAII on the vehicle crashworthiness design problem with respect to the hypervolume (SSC) and generational distance(GD)

Methods	Metrics	
	SSC	GD
HH_CF-HH_RAND	+	+
HH_CF-NSGAII	+	+
HH_CF-AMALGAM	+	+
HH_RAND-AMALGAM	-	+
HH_RAND-NSGAII	-	+
AMALGAM-NSGAII	+	-