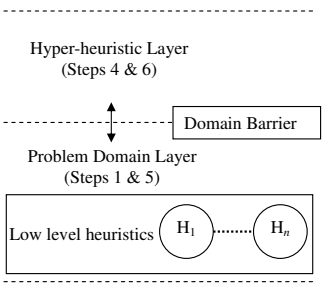
Group Decision Making Hyper-heuristics for Function Optimisation

Ender Özcan*, Mustafa Mısır†, Ahmed Kheiri*
*Automated Scheduling, Optimisation and Planning Research Group
School of Computer Science, University of Nottingham, UK
Email: {exo,axk}@cs.nott.ac.uk
†CODeS research group, Department of Computer Science
KU Leuven, Etienne Sabbelaan 53, 8500, Kortrijk, Belgium
Email: mustafa.misir@kahosl.be

***Abstract*—A hyper-heuristic is a high level methodology which performs search over the space of heuristics each operating on the space of solutions to solve hard computational problems. This search process is based on either *generation* or *selection* of low level heuristics. The latter approach is used in selection hyper-heuristics. A generic selection hyper-heuristic has two main components which operate successively: heuristic selection and move acceptance methods. An initially generated solution is improved iteratively using these methods. At a given step, the most appropriate heuristic is selected from a fixed set of low level heuristics and applied to a candidate solution producing a new one. Then, a decision is made whether to accept or reject the new solution. This process is repeated until the termination criterion is satisfied. There is strong empirical evidence that the choice of selection hyper-heuristic influences its overall performance. This is one of the first studies to the best of our knowledge that suggests and explores the use of group decision making methods for move acceptance in selection hyper-heuristics. The acceptance decision for a move is performed by multiple methods instead of a single one. The performance of four such group decision making move acceptance methods are analysed within different hyper-heuristics over a set of benchmark functions. The experimental results show that the group decision making strategies have potential to improve the overall performance of selection hyper-heuristics.**

I. Introduction

Hyper-heuristics are high level methodologies that search the heuristics-space rather than the solutions, directly in problem solving. One of the earliest studies on hyper-heuristics aiming at exploiting the strengths of multiple neighbourhood operators is provided in [1]. Since then there is a growing interest in hyper-heuristics [2], [3], [4]. There are two main classes of hyper-heuristics in the literature: methodologies to *generate* heuristics and methodologies to *select* heuristics. This study focusses on the latter type of methodologies. A *perturbative* heuristic processes and returns a complete solution. Figure 1 illustrates a selection hyper-heuristic framework performing a single point search based on perturbative low level heuristics [5]. In this framework, the hyper-heuristic layer interacts with the problem domain and heuristic layers through problem independent measures, such as the quality change in a candidate solution when the selected heuristic is employed. [6], [7] identify two key stages in the hyper-heuristic layer: *heuristic selection* and *move acceptance*. An instance of a selection hyper-heuristic will be denoted by a pair as "heuristic selection method"_"move acceptance criterion" from this point forward. A selection hyper-heuristic selects a heuristic from a set of n perturbative low level heuristics $\{H_1, H_2, ..., H_n\}$, then applies the chosen heuristic to the candidate solution. Afterwards, it decides whether to accept or reject the new solution at each step. An initially generated solution goes through this process repetitively until a set of termination criteria is satisfied. Finally, the best solution at hand is returned for a given problem.




```
1.  generate initial candidate solution p
2.  best=p
3.  while (termination criteria not satisfied){
4.    select a heuristic h from {H1, ..., Hn}
5.    generate a new solution s=h(p)
        by applying h to p
6.    decide whether to accept s or not
7.    if (s is accepted) then
8.      p=s;
9.    if (p is better than best) then
10.     best=p;
11. }
12. return best
```

Fig. 1. Layers in a generic hyper-heuristic framework and pseudocode of a selection hyper-heuristic

Most of the selection hyper-heuristics in the literature feature a filter, referred to as *domain barrier* which does not allow any domain specific information to be passed from the problem domain layer to the hyper-heuristic layer [5], [4]. Hence, selection hyper-heuristics, once implemented, are reusable general methods, applicable to different unseen instances from a specific domain as well as different domains. Even the heuristic selection and move acceptance components of hyper-heuristics are reusable. Although [8] implied that hyper-heuristics are problem independent approaches, [7], [6] showed that different combinations of hyper-heuristic components, namely heuristic selection and move acceptance methods might yield different performances even for different problem instances. This observation is crucial, since it implies that another level can be introduced on top of the hyper-heuristics for managing them. Then the question arises: "How are we going to end this hierarchical growth in the levels".

There are successful approaches that adapt group decision making models in different ways for solving engineering and optimisation problems [9], [10]. In [11], ant colony optimisation approach for global optimisation was proposed. In the approach, the ants leave pheromones on the trails that they pass and strengthen link between two objects in the representation.

This process can be considered a voting mechanism. [12] emphasised the role of combining decision rules within the scatter search to derive additional trial solutions. Scatter search is a different optimisation method using a single heuristic as a neighbourhood operator and exploits adaptive memory. In this study, four different move acceptance methods which are derived from well known group decision making models are investigated within the selection hyper-heuristics for the first time in the literature. The proposed strategies combine the decisions of a group of independent move acceptance mechanisms while deciding to accept or reject a single solution at each step. The use of a group decision making strategy allows all member mechanisms to operate at the same level and flattens the hierarchical growth in the hyper-heuristic levels for the move acceptance.

Hyper-heuristics that combine seven heuristic selection methods with the proposed move acceptance strategies are tested over fourteen benchmark functions. Section II provides an overview of the selection hyper-heuristics related to this study. Section III explains group decision making and the proposed hyper-heuristics. The benchmark functions and design of selection hyper-heuristics are overviewed in Section IV. The computational results are discussed in Section V. Section VI concludes the study with a summary and discussion.

II. Selection Hyper-heuristics

Many researchers and practitioners have been progressively involved in hyper-heuristic studies for solving difficult real world combinatorial optimisation problems. There is a growing number of empirical studies indicating the effectiveness of hyper-heuristics as general solvers [4]. A recent theoretical study also shows that mixing heuristics (operators) could perform exponentially faster search than the stand-alone heuristics on some problems [13].

[5] investigated most of the selection hyper-heuristic components. Simple Random (SR) randomly chooses a low level heuristic based on a uniform probability distribution at each step. Random Descent (RD) selects the heuristic in the same manner as SR, but applies it repeatedly until no improvement is achieved. Random Permutation (RP) generates a random initial permutation of the low level heuristics and at each step applies a low level heuristic in the provided order sequentially. Random Permutation Descent (RPD) processes the low level heuristics in the same manner as RP, but proceeds in the same manner as RD without changing the order of heuristics. The Greedy (GR) method applies all heuristics to a given candidate solution and selects the one that generates the most improved solution. Choice Function (CF) uses a learning mechanism that scores low level heuristics based on their individual and pair-wise performances. The heuristic having the best score is selected at each step and applied to the candidate solution. Two naive acceptance criteria were used to combine with the aforementioned heuristic selection mechanisms [5]. The All Moves (AM) acceptance criterion accepts all the generated solutions, while Only Improving (OI) accepts only better quality solutions. Improving and Equal (IE) accepts non-worsening moves. The experimental results showed the superior performance of the CF_AM hyper-heuristic. [14] presented a variant of CF.

[15] compared different Monte Carlo based move acceptance criteria which allow the acceptance of non-improving moves using different probability formula. These strategies are similar to the simulated annealing move acceptance [16] yet without a cooling schedule. The authors reported the success of the Exponential Monte Carlo with Counter (EMCQ) with Simple Random heuristic selection. EMCQ uses the probability of $e^{-\Delta f \times m/Q}$ for accepting non-improving moves, where Δf is the fitness change at a given step, m is the duration of the selected heuristic execution and Q is the number of successive worsening moves. Q is reset whenever there is an improvement.

[17], [7] compared the performances of different selection hyper-heuristics based on four different frameworks. It has been observed that different combinations of heuristic selection and move acceptance components yield different performances. A selection hyper-heuristic using the simulated annealing move acceptance with a linear cooling rate, denoted as MC (Equation 1) performed the best when Choice Function is used as the heuristic selection method.

$$e^{-\frac{\Delta f}{\Delta F(1-\frac{t}{T})}} \tag{1}$$

where Δf is the fitness change at step t, T is the maximum number of steps and ΔF is an expected range for the maximum fitness change.

[2] proposed a hyper-heuristic combining tabu search and ranking as a heuristic selection mechanism (TABU_IE). Tabu search was used to avoid selecting poor performing heuristics by maintaining a tabu list of heuristics. A reinforcement learning based ranking strategy was employed to calculate and update the heuristics' ranks based on their performance. The resulting ranks were then used to choose heuristics.

[18] experimented with a hyper-heuristic with the SR heuristic selection method and the Great Deluge (GD) acceptance criterion stochastic acceptance mechanism. GD is based on a stochastic framework which allows improving moves by default. Non-improving moves are accepted if the objective value of the candidate solution is better or equal to an expected objective value, named as level at each step. The objective value of the first generated candidate solution is used as the initial level. The level is updated at a linear rate towards a final objective value as shown in Equation 2.

$$\tau_t = f_0 + \Delta f \times (1 - \frac{t}{T}) \tag{2}$$

where τ_t is the threshold level at step t in a minimisation problem, T is the maximum number of steps, Δf is an expected range for the maximum fitness change and f_0 is the final objective value. Almost all move acceptance methods in the literature, accept improving moves and they differ in how they handle worsening moves. There is a growing number of studies emphasising the influence of move acceptance methods in hyper-heuristics. [19] uses *late acceptance* strategy which makes the acceptance decision comparing the quality of the current solution to the quality of another solution, produced a fixed number of steps earlier. The threshold move acceptance methods, such as great deluge and adaptive iteration limited list-based threshold accepting (AILLA) [20] showed success

within the selection hyper-heuristic framework. More on selection hyper-heuristics can be found in [21], [22], [7], [2], [4].

III. Group decision making hyper-heuristics

A. Group decision making

Group decision-making is defined as "the process by which a collective of individuals attempt to reach a required level of consensus on a given issue" [23]. This process contains two main phases; discussion between group members and reaching a single group decision. The final outcome requires an agreement based on a specified strategy as decision criteria such as voting.

A decision making process [24] consists of a set of steps to reach a choice. At first, the problem is identified. Then, the factors expected to be influential on the decision are listed. Each member of this list should be associated with a specific weight according to its importance, that is, some kind of priority should be established. After that, the alternatives that can meet the requirements are considered. The effect and performance of each alternative strategy are analysed. Among all the alternatives, the best one is chosen and performed on the given issue. During this process, three main circumstances can be encountered; certainty, uncertainty and risk. From the certainty perspective, all the possible effects of the decisions are known. For uncertainty, information about the results of alternatives is incomplete. Thus, a risk must be taken to get rid of this uncertainty by associating some probabilistic values.

During the group decision making process, one of four main decision making strategies as classified by [25] should be chosen and applied depending on the characteristics of a problem. One of them is the plop method. It works by providing different ideas about a subject and arguing them, then accepting one of them. It is very simple and commonly used approach, but it is not appropriate for all types of group decisions. The other one is group decision making under an authority rule. It is a straightforward strategy depending on the power. For instance, in a company, everyone provides some ideas about a subject and discusses to reach a decision. The final decision is made by an authorised person, such as, a chairman. Another model for group decision is the minority rule. Unlike the previous case, the discussion is rather shallow. An authorised person asks whether the idea is accepted or not and the silence of group members is considered the acceptance of the proposed idea. It is also possible that everyone states an allowed to state an opposing idea, but the final decision can be given by a small group of people, such as, the shareholders of a company without other board of members. The last and the most known one is the majority rule. It can be exemplified with two different approaches. One of them is a well known system, i.e. voting. Everyone votes for a decision, and then the decision received the majority of the votes is the final decision. The other majority rule is pooling. In this case, voting is performed twice. A discussion session is arranged in between them. If the general opinion is the same as before the discussions, the idea is accepted.

B. Group decision making move acceptance methods

Four different group decision making strategies are proposed as a hyper-heuristic move acceptance mechanism: G-AND, G-OR, G-VOT, G-PVO. Each one of these move acceptance mechanisms provides a decision whether a new candidate solution is accepted or not by evaluating the decisions of their member move acceptance mechanisms. Generally speaking, improvements are always accepted and a worsening move subject to the group decision criteria. G-OR and G-AND are biased strategies. G-OR makes an acceptance oriented decision. If the members willing to admit the new solution are in the minority, still, it is accepted. Even if there is a single member that admits the new solution, that member acts as an authority and makes the final decision. On the other hand, G-AND makes a rejection oriented decision. All the member move acceptance mechanisms must be in agreement so that the new solution gets accepted. Even if the members that reject the new solution are in the minority, it is rejected. G-VOT and G-PVO are based on the majority rule. G-VOT is based on the traditional voting scheme. If the majority of members accept the new solution, it is accepted, otherwise it is rejected. G-AND, G-OR and G-VOT act under certainty, whereas G-PVO is modelled favouring uncertainty to a degree using a probabilistic framework while making the final decision. The probability of acceptance of a new solution dynamically changes proportional to the number of members that vote for acceptance within the group at each step in G-PVO. For example, assuming that there are ten members in the group and six of them accept the new solution at a step, then this solution is accepted by G-PVO with a probability of 0.6. None of the group decision making move acceptance criteria requires odd number of members, but it is preferable by G-VOT.

The proposed group decision making move acceptance criteria can be represented by means of a more general model. In this model, given k move acceptance methods, a move is accepted if the inequality is satisfied by the Equation 3, otherwise it is rejected. The contribution of each member move acceptance mechanism towards a final decision for the acceptance can be adjusted through a weight, referred to as $strength$ (s_i). Assuming that all s_i values are 1, the method turns out to be G-AND for $\alpha = k$ and G-OR for $\alpha = 0.5$. If $\alpha = k/2$ and all s_i values are 1, then the method becomes G-VOT. If $\alpha = k \times r$, where r is a uniform random number in [0,1] and all s_i values are $1/k$, then the method becomes G-PVO. More static, dynamic and adaptive group decision making move acceptance strategies can be generated based on this model, which is out of the scope of this paper.

$$\sum_{i=1}^{k} s_i \times D(M_i) \geq \alpha \tag{3}$$

where M_i denotes the i^{th} group member (a move acceptance mechanism), $D(x)$ returns 1, if the strategy x accepts the new solution and 0, otherwise, s_i is the strength of the decision made by the i^{th} member move acceptance mechanism and α denotes a threshold value.

A move acceptance criterion used in a selection hyper-heuristic is categorised as *non-deterministic* if the acceptance decision depends on current time (iteration). If the acceptance

decision is the same for a given new and current solution at any point during the search process, the move acceptance criterion is considered as a *deterministic* criterion. Additionally, an acceptance mechanism can be characterised as *stochastic* (non-stochastic) if a probabilistic framework is (not) utilised while accepting or rejecting a move. Existing move acceptance fall in one of the three categories presented in Table I. A selection hyper-heuristic using a move acceptance method based on a group decision making model will be referred to as a *group decision making hyper-heuristic* from this point on.

IV. Group decision making hyper-heuristics for benchmark function optimisation

Seven heuristic selection methods {SR, RD, RP, RPD, CF, GR, TABU} are combined with four group decision making move acceptance mechanisms {G-AND, G-OR, G-VOT, G-PVO}, generating twenty eight hyper-heuristics. The move acceptance mechanisms embed M_1=IE (improving and equal), M_2=MC (simulated annealing) and M_3=GD (great deluge) as group members. These group members were selected due to their high performance reported in [7]. It should be also noted that each member falls into a different category, previously mentioned in Table I.

A. Benchmark function test suite

Benchmark functions provide an excellent controlled environment for evaluating a new approach and comparing its performance to the other approaches. Fourteen well known selected benchmark functions including the De Jong's test suite [26] are used to investigate the performance of group decision making hyper-heuristics. Table II provides the characteristics of these functions. There are eleven continuous (F1-F11) and three discrete functions (F12-F14) in the test set. The discrete functions are $deceptive$ functions because of the large hamming distance between the local optima and the global optimum. A unimodal benchmark function contains a single optimum, whereas a multimodal benchmark function has at least one local optimum that may cause a search method getting trapped. Separability property determines the dependency between the dimensional encoding of solutions and ease of evaluating candidate solution. In a separable function, the evaluation process can be divided into a set of independent evaluations for each dimensional encoding. This process allows delta evaluation in the case of a localised change within a dimension eliminating the need for decoding all dimensions to evaluate a given candidate solution.

B. Frameworks, low level heuristics and encoding for benchmark function optimisation

A hyper-heuristic framework (F_A) without differentiating the low level heuristics is presented in [2]. On the other hand, [17] separate $mutational\ heuristics$ and $hill\text{-}climbers$ and propose three additional hyper-heuristic frameworks (F_B, F_C, F_D). An improved or equal quality solution is expected from a hill climber as a local search component, while a mutational heuristic is a methodological random perturbation. The best performing hyper-heuristic framework, F_C, applies a predetermined hill climber right after a mutational heuristic. Memetic algorithms [36] that hybridise genetic algorithms and hill climbers and iterated local search utilise similar ideas [37]. The results in [7] show that a hyper-heuristic based on the F_C framework can compete with the performance of a memetic algorithm.

The low level heuristics for benchmark function optimisation includes three mutational heuristics and three hill climbers [7]:

- **H1:** $mutation$ (MUTN) flips a randomly selected bit.
- **H2:** $dimensional\ mutation$ (DIMM) randomly selects a dimension and flips a random bit.
- **H3:** $swap\ dimension$ (SWPD) swaps all bits between two randomly selected dimensions.
- **H4:** $random\ mutation\ hill\ climber$ (RMHC) flips a randomly selected bit. If the change improves the solution, it is accepted (otherwise rejected). This process is repeated n times, where n is the length of a candidate solution (number of bits).
- **H5:** $next\ gradient\ hill\ climber$ (NGHC) starts with the first bit and flips it. If this change improves the solution, it is accepted (otherwise rejected), and then the next bit is considered until all the bits are processed.
- **H6:** $Davis's\ bit\ hill\ climber$ (DBHC) operates similar to NGHC. The only difference is that DBHC uses a random ordering of bits instead of starting from the first bit and moving to the next one.

In this study, the traditional hyper-heuristic framework is investigated as well as the F_C framework for benchmark function optimisation. Within the F_C framework, a hyper-heuristic controls three mutational heuristics (H1, H2 and H3). DBHC (H6) is employed after applying a mutational heuristic. Gray encoding is used to represent the candidate solutions for the continuous functions.

V. Computational Results

Pentium IV 3 GHz LINUX machines having 2.00GB memories are used for the benchmark function optimisation experiments. Each experiment on a benchmark function is repeated for fifty times. The experiments are terminated if the execution time exceeds 600 CPU seconds or the expected global optimum (see Table II) is achieved. The performance of algorithms are evaluated based on $success\ rate$, $s.r.$, denoting the ratio of the number of successful trials for which the expected fitness (optimum) is achieved to the total number of trials: $s.r.$ = [number of trials in which optimum is obtained]/[50 (trials)]. Two sets of experiments are performed using different selection hyper-heuristic frameworks.

All twenty eight group decision making hyper-heuristics combining {SR, RD, RP, RPD, CF, GR, TABU} with {G-AND, G-OR, G-VOT, G-PVO} are applied to the benchmark function optimisation problems using the traditional hyper-heuristic framework (F_A) in the first set of experiments. The experimental results show that G-VOT as a group decision making move acceptance mechanism performs the best considering the average success rate over all test cases as illustrated in Figure 2(a). G-PVO, G-AND and G-OR follow G-VOT performance-wise in that order. The heuristic selection

TABLE I. CATEGORISATION OF SOME EXISTING MOVE ACCEPTANCE METHODS USED WITHIN THE SELECTION HYPER-HEURISTICS.

	deterministic	*non-deterministic*
stochastic	-	Simulated Annealing variants, EMCQ
non-stochastic	Accept all, Improving and Equal, Only Improving	Great Deluge, Late Acceptance, AILTA

TABLE II. CHARACTERISTICS OF THE BENCHMARK FUNCTIONS

Label	Function Name	[lb, ub]	Dimension	optimum	isContinuous	isMultimodal	isSeparable	Source
F1	Sphere	[-5.12,5.12]	10	0	✓	x	✓	[26]
F2	Rosenbrock	[-2.048,2.048]	10	0	✓	x	✓	[26]
F3	Step	[-5.12,5.12]	10	0	✓	x	✓	[26]
F4	Quartic with noise	[-1.28,1.28]	10	1	✓	✓	✓	[26]
F5	Foxhole	[-65.536,65.536]	2	1	✓	✓	x	[26]
F6	Rastrigin	[-5.12,5.12]	10	0	✓	✓	✓	[27]
F7	Schwefel	[-500,500]	10	0	✓	✓	✓	[28]
F8	Griewangk	[-600,600]	10	0	✓	✓	x	[29]
F9	Ackley	[-32.768,32.768]	10	0	✓	✓	x	[30]
F10	Easom	[-100,100]	6	-1	✓	x	x	[31]
F11	Schwefels Double Sum	[-65.536,65.536]	10	0	✓	x	x	[28]
F12	Royal Road	n/a	8	0	x	n/a	✓	[32]
F13	Goldberg	n/a	30	0	x	n/a	✓	[33], [34]
F14	Whitley	n/a	6	0	x	n/a	✓	[35]

methods deliver similar performances in the overall. CF as a heuristic selection method delivers a $slightly$ better performance when compared to the other heuristic selection methods with an average success rate of 0.78 over all test cases as illustrated in Figure 2(b).

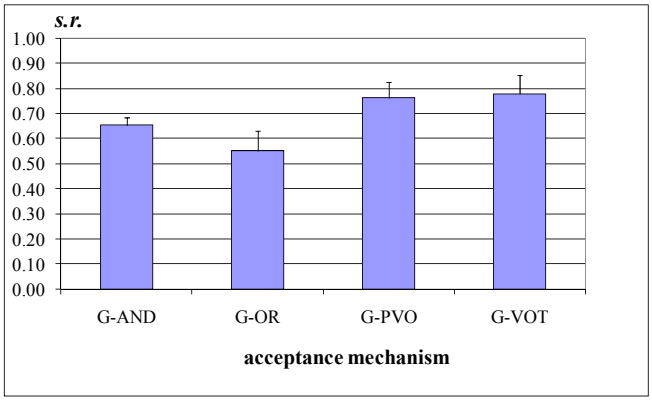
Table III provides the performance comparison of hyper-heuristics combining CF with a different group decision making move acceptance method with respect to success rate. The CF_G-VOT hyper-heuristic performs the best with an average success rate of 0.92 over all benchmark functions. CF_G-VOT achieves a success rate that is greater or equal to 0.96 for F4, F6 and F10 functions as shown in Table III. The full success is obtained in locating the global optimum for all functions, excluding F13. This hyper-heuristic is obviously susceptible to deception. The global optimum is not found for Goldberg's deceptive function (F13) in none of the runs. On the other hand, hyper-heuristics using G-AND locates the global optimum for F13 at least for once when combined with any heuristic selection method, other than CF.

TABLE III. PERFORMANCE COMPARISON OF DECISION MAKING HYPER-HEURISTICS USING CHOICE FUNCTION, CF, AS A HEURISTIC SELECTION COMPONENT OVER BENCHMARK FUNCTIONS BASED ON SUCCESS RATE.

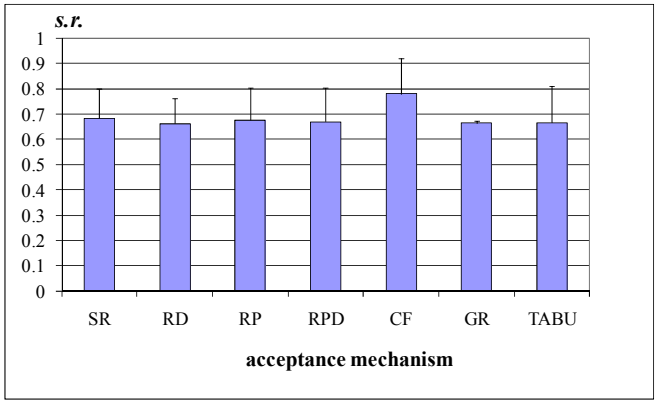
label	G-VOT	G-PVO	G-AND	G-OR
F1	1.00	1.00	1.00	1.00
F2	1.00	1.00	0.00	1.00
F3	1.00	1.00	1.00	0.78
F4	0.96	0.92	0.54	0.64
F5	1.00	1.00	1.00	1.00
F6	0.96	0.48	1.00	0.04
F7	1.00	1.00	1.00	0.50
F8	1.00	1.00	0.04	1.00
F9	1.00	1.00	1.00	1.00
F10	0.98	0.92	1.00	0.96
F11	1.00	1.00	0.02	1.00
F12	1.00	1.00	1.00	0.00
F13	0.00	0.00	0.00	0.00
F14	1.00	1.00	1.00	0.00
$avr.$	0.92	0.88	0.69	0.64

The experiments are repeated in the same environment with the same settings using hyper-heuristics that combine CF with each member move acceptance method from the group as a stand-alone (individual) move acceptance method. CF_IE, CF_GD and CF_MC generate an average success rate of 0.69, 0.88 and 0.91 over all benchmark functions, respectively. CF_G-VOT performs slightly better than CF_G-PVO, CF_GD and CF_MC. Its performance is better than CF_IE, CF_G-AND, and CF_G-OR and this performance difference is statistically significant within a confidence interval of 97% based on a two-tailed pairwise student's t-test of success rates over all benchmark functions.

The second set of experiments are performed using the same experimental setting with all the decision making hyper-heuristics based on the F_C framework as described in section IV. The results show that almost in all cases, the performance of the group decision making hyper-heuristics improves. Figure 3 illustrates the overall evaluation. Although G-AND turns out to be the best choice compared to the other group decision making move acceptance methods. The pair-wise performance difference between G-AND and each approach in {G-PVO, G-VOT} is not statistically significant. The performance of G-OR worsens when the F_C framework is used instead of the traditional one. As a result, it is observed that majority towards an agreement of acceptance is more valuable among the group members. The heuristic selection methods starting from the one having the best performance to the worst are GR, CF, TABU, RPD, SR, RP and RD, respectively, considering average success rates for heuristics selection methods on all test cases. The GR_G-PVO and GR_G-VOT hyper-heuristics are the best performing hyper-heuristics when the underlying framework is of type F_C generating the perfect success rate of 1.00 locating the optimum for each test function in the



(a)



(b)

Fig. 2. Average success rate (and the associated standard deviation) for (a) each heuristic selection method, (b) each group decision making acceptance mechanism over all benchmark functions when used based on the generic selection hyper-heuristic framework of F_A

benchmark across all runs. This performance is the same as the performance of the best memetic algorithm (bMA) as described in [7]. Changing from the underlying traditional framework to the F_C framework improves the performance of the CF_G-VOT hyper-heuristic from an average success rate of 0.92 to 0.99 over all benchmark functions. The pairwise performance difference between CF_G-VOT and each approach in {GR_G-PVO, GR_G-VOT, bMA} is not statistically significant.

VI. Conclusion and Future Work

A class of hyper-heuristics contains methodologies that combine two consecutive processes, namely heuristic selection and move acceptance in a single point based search framework. An initial solution is improved iteratively by applying a set of perturbative low level heuristics until termination. It has already been empirically observed that the choice of move acceptance method for a hyper-heuristic is critical. Different combinations of a heuristic selection method and a move acceptance criterion in a hyper-heuristic might yield different performance across different problem domains [6], [17], [7]. This study investigates the effectiveness of group decision making for move acceptance in hyper-heuristics. The decisions of a group of independent move acceptance criteria

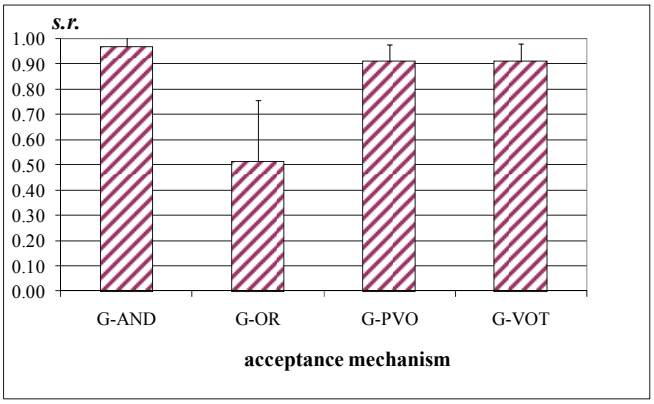


Fig. 3. Average success rate (and the associated standard deviation) for each group decision making acceptance mechanism over all benchmark function experiments when used based on the selection hyper-heuristic framework of F_C

are combined to make a single decision whether to accept or reject a new solution during the search process. The experimental results show that the majority rule based group decision making move acceptance methods can significantly improve the performance of a selection hyper-heuristic. The traditional AND-operator and probabilistic voting schemes which dynamically compute the acceptance probability of a move based on the votes from the group members are the most successful mechanisms to be used within the selection hyper-heuristics. If the mutational and hill climbing heuristics can be distinguished and implemented separately for a given problem, an additional improvement can be obtained by using the memetic hyper-heuristic framework (F_C) [7].

The use of multiple move acceptance methods based on group decision making models within selection hyper-heuristics appears to produce a synergy among the members yielding a better performance when compared to the performance of hyper-heuristics using each member method stand-alone. The effectiveness and generality level that the group decision making hyper-heuristics achieve will be further investigated on other problem domains.

References

[1] H. Fisher and G. L. Thompson, "Probabilistic learning combinations of local job-shop scheduling rules," in *Industrial Scheduling*, J. F. Muth and G. L. Thompson, Eds. New Jersey: Prentice-Hall, Inc, 1963, pp. 225–251.

[2] E. K. Burke, G. Kendall, and E. Soubeiga, "A tabu-search hyperheuristic for timetabling and rostering," *Journal of Heuristics*, vol. 9, no. 6, pp. 451–470, 2003.

[3] P. Ross, "Hyper-heuristics," in *Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques*, E. K. Burke and G. Kendall, Eds. Springer, 2005, ch. 17, pp. 529–556.

[4] E. K. Burke, M. Gendreau, M. Hyde, G. Kendall, G. Ochoa, E. Özcan, and R. Qu, "Hyper-heuristics: A survey of the state of the art," *Journal of the Operational Research Society*, 2013.

[5] P. Cowling, G. Kendall, and E. Soubeiga, "A hyperheuristic approach to scheduling a sales summit," in *Selected papers from the Third International Conference on Practice and Theory of Automated Timetabling*. London, UK: Springer-Verlag, 2001, pp. 176–190.

[6] B. Bilgin, E. Özcan, and E. Korkmaz, "An experimental study on hyper-heuristics and exam timetabling," in *Practice and Theory of Automated Timetabling VI*, ser. Lecture Notes in Computer Science, E. Burke and

H. Rudová, Eds. Springer Berlin / Heidelberg, 2007, vol. 3867, pp. 394–412.

[7] E. Özcan, B. Bilgin, and E. E. Korkmaz, "A comprehensive analysis of hyper-heuristics," *Intelligent Data Analysis*, vol. 12, no. 1, pp. 3–23, 2008.

[8] P. Cowling, G. Kendall, and E. Soubeiga, "A hyperheuristic approach for scheduling a sales summit," in *Selected Papers of the Third International Conference on the Practice And Theory of Automated Timetabling, PATAT 2000*, ser. Lecture Notes in Computer Science. Konstanz, Germany: Springer, August 2000, pp. 176–190.

[9] J. Mottl, "Excavator optimization using the voting method," *Computer Methods in Applied Mechanics and Engineering*, vol. 98, no. 2, pp. 227 – 250, 1992.

[10] M.-S. Wang and W.-C. Chen, "A majority-voting based watermarking scheme for color image tamper detection and recovery," *Computer Standards & Interfaces*, vol. 29, no. 5, pp. 561 – 570, 2007.

[11] M. Dorigo and T. Stützle, *Ant Colony Optimization*. MIT Press, 2004.

[12] F. Glover, M. Laguna, and R. Mart, "Fundamentals of scatter search and path relinking," *Control and Cybernetics*, vol. 39, pp. 653–684, 2000.

[13] P. K. Lehre and E. Özcan, "A runtime analysis of simple hyper-heuristics: To mix or not to mix operators," 2013, pp. 97–104.

[14] J. H. Drake, E. Özcan, and E. K. Burke, "An improved choice function heuristic selection for cross domain heuristic search," in *PPSN (2)*, ser. Lecture Notes in Computer Science, C. A. C. Coello, V. Cutello, K. Deb, S. Forrest, G. Nicosia, and M. Pavone, Eds., vol. 7492, 2012, pp. 307–316.

[15] M. Ayob and G. Kendall, "A monte carlo hyper-heuristic to optimise component placement sequencing for multi head placement machine," in *Proceedings of the International Conference on Intelligent Technologies (InTech'03)*, Chiang Mai, Thailand, December 17-19 2003, pp. 132–141.

[16] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, pp. 671–680, 1983.

[17] E. Özcan, B. Bilgin, and E. E. Korkmaz, "Hill climbers and mutational heuristics in hyperheuristics," in *Proceedings of the 9th international conference on Parallel Problem Solving from Nature*, ser. PPSN'06. Berlin, Heidelberg: Springer-Verlag, 2006, pp. 202–211.

[18] G. Kendall and M. Mohamad, "Channel assignment optimisation using a hyper-heuristic," in *Proceedings of the 2004 IEEE Conference on Cybernetic and Intelligent Systems (CIS2004)*, Singapore, 1-3 December 2004, pp. 790–795.

[19] E. Özcan, Y. Bykov, M. Birben, and E. K. Burke, "Examination timetabling using late acceptance hyper-heuristics," in *Proc. of the IEEE Congress on Evolutionary Computation*. IEEE Press, 2009, pp. 997–1004.

[20] M. Misir, K. Verbeeck, P. Causmaecker, and G. Berghe, "The effect of the set of low-level heuristics on the performance of selection hyper-heuristics," in *Parallel Problem Solving from Nature - PPSN XII*, ser. Lecture Notes in Computer Science, C. Coello, V. Cutello, K. Deb, S. Forrest, G. Nicosia, and M. Pavone, Eds. Springer Berlin Heidelberg, 2012, vol. 7492, pp. 408–417.

[21] E. K. Burke, M. R. Hyde, G. Kendall, G. Ochoa, E. Özcan, and J. R. Woodward, "Exploring hyper-heuristic methodologies with genetic programming," in *Computational Intelligence*, ser. Intelligent Systems Reference Library, J. Kacprzyk, L. C. Jain, C. L. Mumford, and L. C. Jain, Eds. Springer Berlin Heidelberg, 2009, vol. 1, pp. 177–201.

[22] K. Chakhlevitch and P. Cowling, "Hyperheuristics: Recent developments," 2008, pp. 3–29.

[23] K. Eliaz, D. Ray, and R. Razin, "Group decision-making in the shadow of disagreement," *J. Economic Theory*, vol. 132, no. 1, pp. 236–273, 2007.

[24] S. P. Robbins and D. A. DeCenzo, *Fundamentals of Management*, 4th ed. Prentice Hall, 2003.

[25] A. E. Schwartz, "Group decision-making," *The CPA Journal*, vol. 64, no. 8, pp. 60–62, 1994.

[26] K. De Jong, "An analysis of the behavior of a class of genetic adaptive systems," Ph.D. dissertation, University of Michigan, 1975.

[27] L. A. Rastrigin, "Extremal control system," in *Theoretical Foundations of Engineering Cybernetics Series*, Nauka, Moscow, 1974.

[28] H.-P. Schwefel, *Numerical Optimization of Computer Models*. New York, NY, USA: John Wiley & Sons, Inc., 1981.

[29] A. O. Griewangk, "Generalized descent of global optimization," *Journal of Optimization Theory and Applications*, vol. 34, no. 1, pp. 11 – 39, 1981.

[30] D. H. Ackley, "An empirical study of bit vector function optimization," in *Davis, L. (ed.) Genetic Algorithms and Simulated Annealing*, vol. 1. London: Pitman Publishing, 1987, pp. 170–215.

[31] E. E. Easom, "A survey of global optimization techniques," Master's thesis, University of Louisville, USA, 1990.

[32] M. Mitchell and S. Forrest, "Fitness landscapes: Royal road functions," in *Handbook of Evolutionary Computation*, T. Bäck, D. Fogel, and Z. Michalewicz, Eds. Institute of Physics Publishing and Oxford University, 1997, pp. 1–25.

[33] D. E. Goldberg, "Genetic algorithms and walsh functions: Part i, a gentle introduction," *Complex Systems*, vol. 3, no. 2, pp. 129–152, 1989.

[34] D. E. Goldberg, "Genetic algorithms and walsh functions: Part ii, deception and its analysis," *Complex Systems*, vol. 3, no. 2, pp. 153–171, 1989.

[35] L. D. Whitley, "Fundamental principles of deception in genetic search," in *Foundations of Genetic Algorithms*, G. J. E. Rawlins, Ed. Morgan Kaufmann, 1991, pp. 221–241.

[36] P. Moscato and M. Normam, "A memetic approach for the travelling salesman problem implementation of a computational ecology for combinatorial optimization on message-passing systems," in *Proceedings of the International Conference on Parallel Computing and Transportation Applications*, 1992, pp. 177–186.

[37] H. R. Lourenço, O. Martin, and T. Stützle, "Iterated local search," in *Handbook of Metaheuristics*, ser. International Series in Operations Research & Management Science, F. W. Glover and G. A. Kochenberger, Eds. Springer, 2003, vol. 57, ch. 11, pp. 320–353.