2013 IEEE Congress on Evolutionary Computation
June 20-23, Cancún, México

Multi-Method Algorithms: Investigating the Entity-to-Algorithm Allocation Problem

Jacomine Grobler
Department of Industrial and Systems Engineering
University of Pretoria.
Pretoria, South Africa
Email: jacomine.grobler@gmail.com

Andries P. Engelbrecht
Department of Computer Science
University of Pretoria
Pretoria, South Africa

Graham Kendall[1] and V.S.S. Yadavalli[2]
School of Computer Science
University of Nottingham, UK and Malaysia[1]
Department of Industrial and Systems Engineering
University of Pretoria
Pretoria, South Africa[2]

***Abstract*—This paper investigates the algorithm selection problem, otherwise referred to as the entity-to-algorithm allocation problem, within the context of three recent multi-method algorithm frameworks. A population-based algorithm portfolio, a meta-hyper-heuristic and a bandit based operator selection method are evaluated under similar conditions on a diverse set of floating-point benchmark problems. The meta-hyper heuristic is shown to outperform the other two algorithms.**

I. Introduction

Over the last five decades a large number of optimization algorithms have been developed to address numerous real world optimization problems. Unfortunately, it is not always easy, or even possible, to predict which one of the many algorithms already in existence will be the most suitable for solving a specific problem. This unpredictability is not only limited to different algorithms on different problem classes, but there may even be issues with respect to large variations in algorithm performance over different instances of the same problem. Within this context, the simultaneous use of more than one algorithm for solving optimization problems becomes an attractive alternative.

A multi-method algorithm can be described as consisting of one or more entities representing solutions which are evolved over time, a set of available algorithms or operators, referred to as constituent algorithms, and a high level strategy responsible for allocating the entities to the most suitable algorithms for optimization. This paper focuses on investigating various entity-to-algorithm allocation strategies.

The idea of multiple algorithms collaborating and competing to achieve a better result than each constituent algorithm can achieve in isolation, is not new. In fact, multi-method algorithms have started appearing in various different domains over the last couple of years. Examples include memetic computation [1], algorithm portfolios [2], algorithm ensembles [3], and hyper-heuristics [4].

Unfortunately, research within these areas is often performed in isolation with little interaction between the different research areas. An inspection of the multi-method literature, in general, uncovers similar concepts described by means of different terminologies. Algorithms are also not always compared on the same benchmark problem set and the selection of constituent algorithms and algorithm control parameters such as population size can also have a significant impact on the fairness with which algorithms are compared. It thus makes sense to be able to compare multi-method algorithms in a meaningful way, the same constituent algorithms and control parameters should be used for each investigated multi-method framework. Unfortunately, this is not common practice in literature making it exceedingly difficult to gauge the success of different multi-method algorithms from different research areas. The value that can be obtained from the comparison of different multi-method algorithms under strictly similar conditions, should thus not be underestimated.

After a brief analysis of the available literature, three multi-method algorithms were selected for evaluation. The population-based algorithm portfolio (PAP) of Peng *et al.* [5], the fitness based area under the curve bandit operator selection method (AUC-Bandit) [6], and the heterogeneous meta-hyper-heuristic (HMHH) [7] algorithm were implemented and compared with regards to the effectiveness of their entity-to-algorithm allocation strategy.

Algorithm performance was evaluated on a set of varied floating-point benchmark problems. The most promising results were obtained by the HMHH algorithm which outperformed the other two strategies on a large percentage of the problems. Good performance, when compared to its constituent algorithms, is also demonstrated.

The rest of the paper is organized as follows: Section II provides a brief overview of the multi-method algorithm literature. Section III describes the multi-method algorithms which were evaluated. The results are documented in Section IV before the paper is concluded in Section V.

II. A brief review of related literature

A number of research directions investigating the use of more than one optimization algorithm simultaneously have been developed in the last few years. This section provides a closer inspection of multi-method, portfolio algorithms and hyper-heuristics - all fields that have already considered the entity-to-algorithm allocation problem.

978-1-4799-0454-9/13/$31.00 ©2013 IEEE

Two main approaches can be identified from the literature for allocating entities to algorithms. Static entity-to-algorithm allocation assigns entities to algorithms at the start of the optimization run and this allocation remains static throughout the rest of the run. Dynamic entity-to-algorithm allocation continuously updates the allocation of entities to algorithms throughout the optimization run.

Memetic algorithms can be considered to be the first multi-method techniques applied to the field of computational intelligence [8]. Recently, more complex self-adaptive multi- method algorithms have been developed. The self-adaptive differential evolution algorithm of Qin and Suganthan [9] makes use of different differential evolution (DE) learning strategies which are weighted based on previous algorithm success.

The heterogeneous cooperative algorithm of Olorunda and Engelbrecht [10] makes use of different evolutionary algorithms to update each of the sub-populations in a cooperative algorithm framework, thereby combining the strengths and weaknesses of various optimization strategies within the same algorithm.

Peng *et al.* [5] developed the population based algorithm portfolio. This algorithm is based on the principle of multiple subpopulations each assigned to one algorithm from a portfolio of available algorithms. At pre-specified time intervals, entities are migrated between subpopulations to ensure effective information sharing between the different optimization algorithms.

Vrugt et als highly successful population-based genetic adaptive method for single objective optimization (AMALGAM-SO) [11] is one of the few examples of an algorithm which continually updates the allocation of algorithms to entities during the optimization run. AMALGAM-SO employs a self-adaptive learning strategy to determine the percentage of candidate solutions in a common population to be allocated to each of the three evolutionary algorithms. A restart strategy is used to update the percentages based on algorithm performance.

Another successful adaptive strategy selection mechanism was investigated by Fialho *et al.* [6]. Comparisons of alternative credit assignment methods [12] and strategy selection mechanisms within a differential evolution framework [13], highlighted the superior performance of the fitness-based area under curve bandit (AUC-Bandit) technique.

Various other ensemble strategies have also been developed in the differential evolution domain for the intelligent selection of mutation operators and control parameters during optimization [14] [15] [16].

Hyper-heuristics [17] promote the design of more generally applicable search methodologies and tend to focus on performing relatively well on a large set of different problems, in contrast to specialized algorithms which focus on outperforming the state-of-the-art for a single application. Most recent hyper-heuristic algorithms consist of a high level methodology which controls the selection or generation of a generic search strategy while using a set of low-level heuristics as input. This strategy facilitates the automatic design of several algorithmic aspects, thus the impact of hyper-heuristic research on recent optimization trends is significant.

Various hyper-heuristics have been developed over the last decade which could prove an effective means of addressing the algorithm selection problem. The tabu-search hyper-heuristic of Burke *et al.* [18] and the simulated annealing based hyper-heuristic of Dowsland *et al.* [19] are notable examples.

Based on the available literature, a number of state of the art algorithms can be identified from each of the above mentioned fields. Three algorithms, namely the population- based algorithm portfolio of Peng *et al.* [5], the AUC-Bandit method [13], and the heterogeneous meta-hyper-heuristic algorithm [7] will be investigated in this paper.

III. A SELECTION OF MULTI-METHOD ALGORITHMS

One of the main ideas behind multi-method algorithms is that the simultaneous use of more than one search algorithm during the optimization process allows the algorithms to exploit each other's strengths while also compensating for inherent weaknesses.

It has already been suggested that the effectiveness of the constituent algorithms, when applied in isolation, have a significant impact on the high level multi-method strategy's performance.

The constituent algorithms used in this paper are:

- A genetic algorithm (GA) with a floating-point representation, tournament selection, blend crossover [20] [10], and self-adaptive gaussian mutation [7].
- The guaranteed convergence particle swarm optimization algorithm (GCPSO) [21].
- The self-adaptive (SaNSDE) algorithm of [22].
- The covariance matrix adapting evolutionary strategy algorithm (CMAES) [23].

The composition of the set of available algorithms is an important consideration. One of the main ideas of using multiple algorithms in the same optimization run, is that the algorithms should complement each other: the set of available algorithms compensating for the strength and weaknesses of each individual algorithm. There are various methods to define the extent to which a set of algorithms complement each other. Hadka and Reed [24] based the selection of mutation strategies available to their algorithm on the distribution of offspring associated with various operators.

The set of constituent algorithms in this investigation were selected to demonstrate different algorithm diversity profiles. For example, as can be seen in Figure 1 of the first 2005 IEEE Congress of Evolutionary Computation benchmark problem set in 50 dimensions [25], the population diversity of the GCPSO algorithm decreases at a much slower rate than the population diversity of the CMAES algorithm. The idea is that, during different stages of the optimization process, different exploration and exploitation rates are necessary. By making sure that algorithms that address the exploration-exploitation trade-off differently are available, the chances of countering an improper population diversity management strategy by one

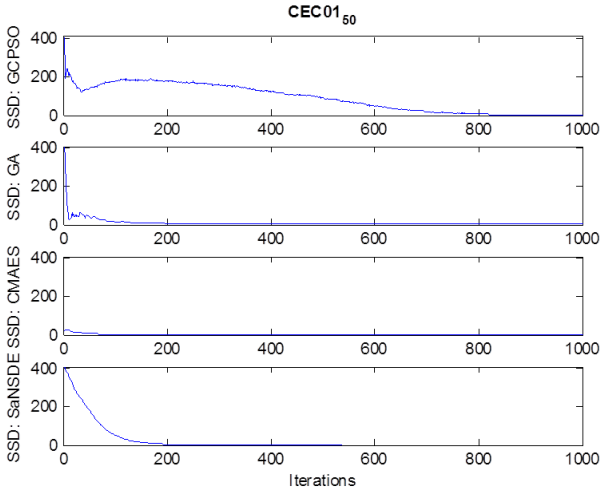

Fig. 1: Diversity profiles of constituent algorithms.

algorithm from the set, is greater. Population diversity, SSD in Figure 1, was defined as:

$$SSD = \frac{1}{n_s} \sum_{i=1}^{n_s} \sqrt{\sum_{j=1}^{n_x} (x_{ij}(t) - \overline{x}_j(t))^2} \quad (1)$$

where n_s is the number of entities in the population and n_x is the number of dimensions; $x_{ij}(t)$ is the position of the j^{th} dimension of the i^{th} entity at time t.

The three multi-method algorithm frameworks which were selected for comparison are described in the rest of this section.

A. Population based Algorithm Portfolio

Peng *et al.* [5] developed the population based algorithm portfolio (PAP). Their work focuses on reducing the risk of poor algorithm performance by dividing the number of allowable function evaluations between a portfolio of available algorithms. As described in Algorithm 1, entities are divided into subpopulations which are evolved in parallel by an assigned constituent algorithm, where one constituent algorithm is used per subpopulation. Each entity only has access to other entities within the same subpopulation in order to prevent the same genetic material from being evolved repeatedly. At prespecified time intervals referred to as $migrationintervals$, $migrationsize$ entities are migrated between subpopulations to ensure effective information sharing between the different optimization algorithms.

PAP can be considered a static algorithm selection mechanism since the allocation of entities to algorithms are only performed once at the start of the optimization process and the allocation remains the same throughout the optimization process.

B. Fitness area under the curve multi armed bandit

The second framework investigated is an online operator selection strategy, where constituent algorithm selection is performed before the generation of each new entity. The fitness-based AUC-Bandit algorithm [13] consists of a credit assignment mechanism and a strategy selection mechanism. The credit assignment mechanism evaluates the performance of each of the available constituent algorithms based on previous successes.

Algorithm 1: The PAP algorithm [5].

```
Initialize m subpopulations, pop_1, pop_2, ..., pop_m
while Stopping conditions are not satisfied do
    Evaluate all the entities in pop_1, pop_2, ..., pop_m
    for All algorithms m do
        Evolve pop_m using algorithm m
    end
    if Migration interval is reached then
        Activate migration procedure as follows:
        for All populations m do
            For pop_m, select migrationsize best
            individuals from the other m − 1
            subpopulations
            denoted as set emigrants_i
            Set pop'_m to pop_m ∪ emigrants_m
            Discard the migrationsize worst individuals
            in pop'_m
        end
        for All populations m do
            Set pop_m to pop'_m
        end
    end
end
```

More specifically, for a given time window W, a ranked list of entities is considered. A receiving operator curve (ROC) for each algorithm is plotted by scanning the ordered list of entities. Starting from the origin, a vertical segment is plotted for every entity that has been generated by the algorithm under consideration and a horizontal segment is drawn otherwise. Ties are broken by means of diagonal segments. A decay factor, D, is also used to bias the selection towards the higher ranked entities. If r is the rank position of an entity, the length of the current segment can be defined as $D^r(W - r)$ with $D \in [0, 1]$. Finally, a ROC such as the example in Figure 2 is obtained for each algorithm. The area under the ROC of algorithm m at time t, namely q_{mt} is then used as the credit associated with the algorithm under consideration.

The strategy selection mechanism utilizes the credit of each operator in a dynamic bandit-based mechanism as described in [13] . The maximization operator is:

$$q_{mt} + C\sqrt{\frac{2\log\sum_k n_{kt}}{n_{mt}}}, \quad (2)$$

where C is a user-defined constant balancing exploration and exploitation and n_{mt} refers to the number of times algorithm m has been used previously, is selected.

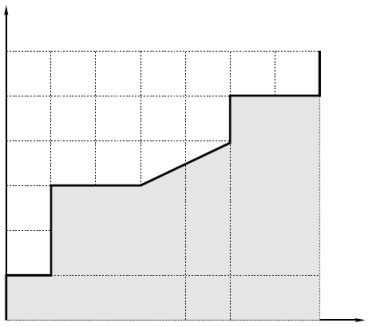

Fig. 2: ROC curve [13].

For the sake of completeness, high level pseudocode of the implementation is provided in Algorithm 2.

Algorithm 2: The AUC-Bandit algorithm [5].

Initialize the parent population $\boldsymbol{X}$
Randomly select the algorithm m_1 from the set of constituent algorithms for allocation to the first entity.
while *Stopping conditions are not satisfied* **do**
    Evolve entity i using algorithm m_i
    Calculate $Q_{\delta m}(t)$, the total improvement in fitness function value of entity i after application of algorithm m and add entity-algorithm allocation information to archive $\boldsymbol{A}$ of size W
    Sort all entries in $\boldsymbol{A}$ according to greatest improvement
    for *All algorithms* m **do**
        Calculate the area under the ROC curve, q_{mt}.
        Calculate the maximization value of algorithm m as defined in Equation 2.
    end
    Allocate entity $i+1$ to the algorithm m_{i+1} which maximizes Equation 2.
    $i = i + 1$
end

C. The heterogeneous meta-hyper-heuristic algorithm

Similar to PAP, the HMHH algorithm [7] also divides the population of entities into a number of subpopulations which are evolved in parallel by the constituent algorithms. However, in contrast to PAP, each entity is able to access the genetic material of other subpopulations, as if part of a common population of entities.

The HMHH algorithm also updates the allocation of entities to constituent algorithms on a dynamic basis throughout the optimization run. The idea is that an intelligent algorithm can be evolved which selects the appropriate constituent algorithm at each k^{th} iteration to be applied to each entity within the context of the common parent population, to ensure that the population of entities converge to a high quality solution. The constituent algorithm allocation is maintained for k iterations, while the common parent population is continuously updated with new information and better solutions. Throughout this process, the various constituent algorithms are ranked based on their previous performance as defined by $Q_{\delta m}(t)$ in Algorithm 3 . A tabu list is used to prevent the algorithm from repeatedly using the same poorly performing operators. The highest ranking non-tabu operator is then selected for each entity during re-allocation of entities to algorithms as described in [4] .

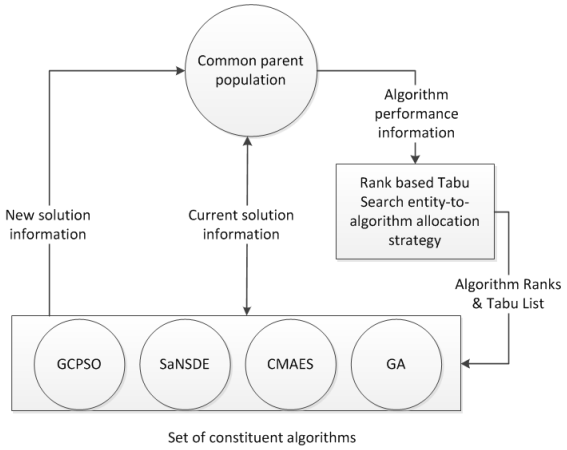

Fig. 3: The HMHH Algorithm

This strategy was considered promising since, similar to the bandit-based framework, each entity can use unique and continuously adapting meta-heuristic operators, helpful for dealing with the specific search space characteristics it is encountering at that specific stage of the optimization process and the role it is playing in the larger algorithm.

IV. Empirical evaluation

The three multi-method algorithm frameworks were evaluated on the first 14 problems of the 2005 IEEE Congress of Evolutionary Computation benchmark problem set in both 10 and 30 dimensions. This benchmark problem set enables algorithm performance evaluation on both unimodal and multimodal functions and includes various expanded and hybridized problems, some with noisy fitness functions.

The control parameters of the three multi-method algorithms are specified in Table I. With regards to the constituent algorithms, the control parameters specified in the original papers ([22],[23],[7]) were used.

The results of the multi-method algorithm framework comparison is presented in Table IV, where the results for each algorithm were recorded over 30 independent simulation runs. μ and σ denote the mean and standard deviation associated

Algorithm 3: The heterogeneous meta-hyper-heuristic.

Initialize the parent population $\boldsymbol{X}$
for *All entities* $i \in \boldsymbol{X}$ **do**
    Randomly select an initial algorithm $A_{im}(1)$ from the set of constituent algorithms to apply to entity i
end
$t = 1$
$k = 5$
while *The stopping conditions are not met* **do**
    for *All entities* i **do**
        Apply constituent algorithm $A_{im}(t)$ to entity i for k iterations
        Calculate $Q_{\delta m}(t)$, the total improvement in fitness function value of all entities assigned to algorithm m from iteration $t+1-k$ to iteration t.
    end
    for *All entities* i **do**
        Use $Q_{\delta m}(t)$ as input to select constituent algorithm $A_{im}(t+k)$ according to the rank based tabu search mechanism described in [4]
    end
    $t = t + k$
end

TABLE I: Algorithm parameters.

Parameter	Value used
Common algorithm parameters	
Population size (n_s)	100
maximum number of iterations (I_{max})	100n
AUC-Bandit	
Size of time window (W)	50
Decay factor (D)	0.5
Exploration-exploitation constant (C)	0.5
PAP	
Number of entities assigned to CMAES	14
Number of entities assigned to GCPSO	18
Number of entities assigned to GA	18
Number of entities assigned to SaNSDE	50
Migration interval	$I_{max}/20$
Number of entities involved in migration	1
HMHH	
Number of iterations between re-allocation (k)	5
Size of tabu list	3

with the corresponding performance measure and $\#FEs$ denotes the number of function evaluations which were needed to reach the global optimum within a specified accuracy. Where the global optimum could not be found within the maximum number of iterations, the final solution at I_{max}, denoted by FFV, was recorded. The best performing algorithm results for each problem is also highlighted in Table IV.

Mann-Whitney U tests were used to evaluate the various strategies according to the number of iterations required to obtain the final fitness function value, as well as the quality of the actual fitness function value. Each strategy was compared to each one of the other strategies, and the number of times the first strategy significantly outperformed the second strategy, performed similarly, or is outperformed by the second strategy, was recorded. The results in Table II are subsequently provided in the form: "Number of wins-number of draws-number of losses". To illustrate, (2-4-22) in row 1 column 2, indicates that the AUC-Bandit strategy outperformed PAP twice over the benchmark problem set. Four draws and 22 losses were recorded.

TABLE II: Hypotheses analysis of alternative multi-method algorithms.

	AUC-Bandit	**PAP**	**HMHH**
AUC-Bandit	NA	$2-4-22$	$2-4-22$
PAP	$22-4-2$	NA	$7-10-11$
HMHH	$22-4-2$	$11-10-7$	NA
TOTAL			
AUC-Bandit		$4-8-44$	
PAP		$29-14-13$	
HMHH		$33-14-9$	

From the results it is clear that the HMHH algorithm outperformed the other two evaluated algorithms. This is most probably due to the careful balance of exploration and exploitation in the rank based tabu search mechanism and the frequent, yet less computational intensive updating of the entity-to-algorithm allocation when compared to the AUC-Bandit appraoch. Since PAP does not make provision for any updating of the entity-to-algorithm allocation during the optimization run, the algorithm is stuck with its initial choices regardless of how the search space and the suitability of the algorithms change over time. The AUC-Bandit approach is extremely slow due to the frequent updating of a computationally complex entity-to-algorithm allocation procedure. The AUC-Bandit approach also performed significantly worse than the other two algorithms. Further investigation into the rate of learning and adaptation to a changing environment could shed light on the poor performance.

In an attempt to verify the actual performance of the multi-method algorithms, the best performing multi-method algorithm from the previous analysis, HMHH, was also compared under similar conditions to its constituent algorithms. The results are recorded in Table V. In Table III Mann-Whitney U tests were used to compare the performance of each constituent algorithm to the HMHH algorithm. The same "number of wins-draws-losses" format of Table II was used.

Similar to the results obtained in [7], it can be seen that the HMHH algorithm performed statistically significantly better a large number of times when compared to three of its four constituent algorithms. CMAES performed better than the HMHH when solving unimodal problems, but the HMHH performance improved in comparison with CMAES as problem size and complexity increased. An inspection of the algorithm ranks does, however, indicate that the HMHH algorithm was able to identify CMAES as the best performing algorithm

TABLE IV: Results of the evaluation of alternative multi-method frameworks on the 2005 IEEE CEC benchmark problem set.

Prob (Dims)	PAP				AUC-Bandit				HMHH			
	FFV		# FEs		FFV		# FEs		FFV		# FEs	
	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
1(10)	$1,00E-06$	0	13857	630,64	$1,00E-06$	0	34853	16340	$1,00E-06$	0	**11870**	538,93
1(30)	$1,00E-06$	0	**39190**	5945,1	$1,00E-06$	0	$1,97E+05$	45674	$1,00E-06$	0	52983	2723,3
2(10)	$1,00E-06$	0	18760	1030,4	0,040	0,095	$1,00E+05$	0	$1,00E-06$	0	**14013**	1246,7
2(30)	$1,00E-06$	0	**90063**	2729,3	51,812	42,305	$3,00E+05$	0	$1,00E-06$	0	90727	15876
3(10)	$1,00E-06$	0	46067	2040,3	$3,03E+05$	$2,00E+05$	$1,00E+05$	0	$1,00E-06$	0	**22750**	3003
3(30)	**1,00E-06**	0	$2,88E+05$	5481,5	$1,91E+06$	$1,07E+06$	$3,00E+05$	0	295,35	998,74	$2,79E+05$	27782
4(10)	$1,00E-06$	0	20483	1424,7	0,11733	0,250	$1,00E+05$	0	$1,00E-06$	0	**15853**	1341,8
4(30)	4448,8	5562,2	$2,87E+05$	39164	2094,3	1418,6	$3,00E+05$	0	**1,00E-06**	0	$1,50E+05$	27962
5(10)	$1,00E-06$	0	25010	2790,5	$1,00E-06$	0	68693	25238	$1,00E-06$	0	**17110**	883,31
5(30)	1115,5	1446,5	$3,00E+05$	0	4781,4	1025,9	$3,00E+05$	0	**174,5**	278,17	$3,00E+05$	0
6(10)	0,133	0,727	50613	10870	27,518	52,871	$1,00E+05$	0	**0**	0	33417	7388,8
6(30)	0,519	1,346	$2,75E+05$	27166	209,62	291,54	$3,00E+05$	0	**0,133**	0,727	$2,43E+05$	33935
7(10)	**0,003**	0,006	62867	35481	0,162	0,136	$1,00E+05$	0	0,162	0,136	$1,00E+05$	0
7(30)	**0**	0	82047	41811	0,004	0,008	$1,30E+05$	$1,13E+05$	0,004	0,008	$1,30E+05$	$1,13E+05$
8(10)	**20,058**	0,086	$1,00E+05$	0	20,328	0,085	$1,00E+05$	0	20,084	0,127	$1,00E+05$	0
8(30)	20,264	0,163	$3,00E+05$	0	20,936	0,041	$3,00E+05$	0	**20,169**	0,120	$3,00E+05$	0
9(10)	0,220	0,401	68493	26929	0,189	0,463	74063	19618	**0,005**	0,005	43320	19202
9(30)	2,6757	1,92	**2,81E+05**	55032	6,917	2,264	$3,00E+05$	0	**2,376**	1,396	$2,98E+05$	10079
10(10)	12,857	6,265	$1,00E+05$	0	**10,443**	5,797	$1,00E+05$	0	15,556	9,175	$1,00E+05$	0
10(30)	70,555	19,764	$3,00E+05$	0	61,007	27,958	$3,00E+05$	0	**55,768**	19,838	$3,00E+05$	0
11(10)	**4,085**	1,2619	$1,00E+05$	0	5,833	1,781	$1,00E+05$	0	5,046	2,253	97283	14880
11(30)	**20,034**	2,6451	$3,00E+05$	0	36,834	3,369	$3,00E+05$	0	24,378	5,122	$3,00E+05$	0
12(10)	**231,53**	539,16	71737	28064	431,33	637,73	$1,00E+05$	0	302,86	546,06	69543	39317
12(30)	3573,5	3310,2	$3,00E+05$	0	6273,5	7858,3	$3,00E+05$	0	2611,5	3622,3	**2,95E+05**	20162
13(10)	**0,498**	0,159	$1,00E+05$	0	1,035	0,506	$1,00E+05$	0	0,526	0,126	$1,00E+05$	0
13(30)	**1,74**	0,421	$3,00E+05$	0	4,642	1,903	$3,00E+05$	0	2,018	0,476	$3,00E+05$	0
14(10)	3,407	0,317	$1,00E+05$	0	**3,097**	0,315	$1,00E+05$	0	3,625	0,309	$1,00E+05$	0
14(30)	**13,026**	0,319	$3,00E+05$	0	13,197	0,353	$3,00E+05$	0	13,14	0,440	$3,00E+05$	0

TABLE V: Comparison results of the HMHH algorithm versus its constituent algorithms on the 2005 IEEE CEC benchmark problem set.

Prob (Dims)	HMHH				CMAES				SaNSDE				GA				GCPSO			
	FFV		# FEs		FFV		# FEs		FFV		# FEs		FFV		# FEs		FFV		# FEs	
	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
1(10)	$1E-06$	0	11870	538, 93	$1E-06$	0	**8526,7**	302, 78	$1,00E-06$	0	20180	424, 59	$1,00E-06$	0	18557	1582, 4	138, 4	6, 807	$1,00E+05$	0
1(30)	$1E-06$	0	52983	2723, 3	$1E-06$	0	**19110**	447, 48	$1,00E-06$	0	38973	839, 51	$1,00E-06$	0	99297	4860, 4	231, 81	29, 491	$3,00E+05$	0
2(10)	$1E-06$	0	14013	1246, 7	$1E-06$	0	**9156,7**	286, 1	$1,00E-06$	0	38377	2088, 3	1, 0873	1, 54	$1,00E+05$	0	544, 1	1, 592	$1,00E+05$	0
2(30)	$1E-06$	0	90727	15876	$1E-06$	0	**26783**	739, 1	$1,00E-06$	0	$2,89E+05$	19917	446, 67	228, 69	$3,00E+05$	0	567	3, 017	$3,00E+05$	0
3(10)	$1E-06$	0	22750	3003	$1E-06$	0	**13320**	379, 11	$1,00E-06$	0	46337	2059, 9	$9,76E+05$	$1,05E+06$	$1,00E+05$	0	970, 96	1411, 8	99177	4509, 6
3(30)	295, 35	998, 74	$2,79E+05$	27782	$1E-06$	0	**61173**	1387, 4	$1,79E+05$	$1,15E+05$	$3,00E+05$	0	$5,93E+06$	$2,65E+06$	$3,00E+05$	0	10543	8705, 2	$3,00E+05$	0
4(10)	$1,00E-06$	0	15853	1341, 8	$1E-06$	0	**9590**	283, 27	$1,00E-06$	0	42767	2605, 2	380, 19	510, 25	$1,00E+05$	0	320, 69	0, 208	$1,00E+05$	0
4(30)	$1,00E-06$	0	$1,50E+05$	27962	$1E-06$	0	**29357**	570, 35	195, 19	186, 51	$3,00E+05$	0	15946	7051, 1	$3,00E+05$	0	325, 45	1, 5416	$3,00E+05$	0
5(10)	$1,00E-06$	0	**17110**	883, 31	$1E-06$	0	17433	546, 04	$1,00E-06$	0	36280	1098, 7	869, 56	1241, 9	$1,00E+05$	0	12, 858	0, 283	$1,00E+05$	0
5(30)	174, 5	278, 17	$3,00E+05$	0	$1E-06$	0	**1,15E+05**	3960, 1	978, 56	425, 28	$3,00E+05$	0	12887	3070	$3,00E+05$	0	22, 356	0, 521	$3,00E+05$	0
6(10)	0	0	33417	7388, 8	0, 001	0, 003	**18950**	744, 52	0, 265	1, 010	52987	13213	346, 91	1274	$1,00E+05$	0	840, 01	$4,63E-13$	26480	799, 31
6(30)	0, 133	0, 727	$2,43E+05$	33935	0, 133	0, 727	$1,20E+05$	37870	0, 859	1, 719	$2,76E+05$	33794	1281, 6	2540, 4	$3,00E+05$	0	840, 01	$4,63E-13$	**69423**	1540
7(10)	0, 162	0, 136	$1,00E+05$	0	1267	$4,63E-13$	$1,00E+05$	0	0, 042	0, 026	99920	438, 18	1267	$4,63E-13$	$1,00E+05$	0	270, 01	$1,73E-13$	**38517**	1131, 4
7(30)	0, 004	0, 008	**1,30E+05**	$1,13E+05$	4696, 3	$2,78E-12$	$3,00E+05$	0	0, 012	0, 018	$1,83E+05$	$1,19E+05$	4696, 3	$2,78E-12$	$3,00E+05$	0	270, 01	$1,73E-13$	$2,00E+05$	11139
8(10)	**20,084**	0, 127	$1,00E+05$	0	20, 312	0, 113	$1,00E+05$	0	20, 345	0, 078	$1,00E+05$	0	20, 197	0, 123	$1,00E+05$	0	64679	60668	$1,00E+05$	0
8(30)	**20,169**	0, 120	$3,00E+05$	0	20, 892	0, 175	$3,00E+05$	0	20, 886	0, 042	$3,00E+05$	0	20, 368	0, 094	$3,00E+05$	0	$6,68E+05$	$3,13E+05$	$3,00E+05$	0
9(10)	0, 005	0, 005	43320	19202	1, 946	1, 511	88203	30601	0	0	43690	2247, 8	0, 003	0, 005	**18983**	6020, 3	120, 01	$7,23E-14$	43280	3701, 2
9(30)	2, 376	1, 396	$2,98E+05$	10079	39, 564	6, 454	$3,00E+05$	0	0	0	**1,06E+05**	4250, 5	0, 004	0, 005	$1,57E+05$	39913	2228, 2	1403, 9	$3,00E+05$	0
10(10)	15, 556	9, 175	$1,00E+05$	0	**1,647**	1, 177	90947	27625	5, 646	1, 367	$1,00E+05$	0	31, 174	12, 988	$1,00E+05$	0	19, 99	0	41123	2053
10(30)	55, 768	19, 838	$3,00E+05$	0	**9,391**	3, 282	$3,00E+05$	0	31, 503	5, 471	$3,00E+05$	0	122, 94	32, 782	$3,00E+05$	0	4699, 3	974, 88	$3,00E+05$	0
11(10)	5, 046	2, 253	97283	14880	1, 299	1, 365	**30310**	10496	5, 325	1, 049	$1,00E+05$	0	7, 588	1, 193	$1,00E+05$	0	306, 49	21, 172	99730	1478, 9
11(30)	24, 378	5, 122	$3,00E+05$	0	**9,026**	3, 055	$3,00E+05$	0	27, 5	1, 648	$3,00E+05$	0	30, 659	3, 453	$3,00E+05$	0	325, 66	46, 216	$3,00E+05$	0
12(10)	302, 86	546, 06	**69543**	39317	1546, 1	2735, 5	70053	43090	24, 701	30, 914	81150	23456	873, 7	1566, 8	$1,00E+05$	0	1547	0, 061	$1,00E+05$	0
12(30)	2611, 5	3622, 3	**2,95E+05**	20162	20324	19261	$3,00E+05$	0	10594	2868, 8	$3,00E+05$	0	15214	11587	$3,00E+05$	0	4978, 1	10, 078	$3,00E+05$	0
13(10)	0, 526	0, 126	$1,00E+05$	0	0, 897	0, 253	$1,00E+05$	0	**0,342**	0, 056	$1,00E+05$	0	0, 440	0, 170	$1,00E+05$	0	10, 237	0, 098	$1,00E+05$	0
13(30)	2, 018	0, 476	$3,00E+05$	0	3, 179	0, 561	$3,00E+05$	0	**1,298**	0, 118	$3,00E+05$	0	1, 657	0, 473	$3,00E+05$	0	10, 803	0, 115	$3,00E+05$	0
14(10)	3, 625	0, 309	$1,00E+05$	0	**2,585**	0, 530	$1,00E+05$	0	3, 274	0, 253	$1,00E+05$	0	3, 691	0, 314	$1,00E+05$	0	27, 558	1, 448	96700	12585
14(30)	13, 14	0, 440	$3,00E+05$	0	10, 394	0, 810	$3,00E+05$	0	12, 707	0, 237	$3,00E+05$	0	13, 092	0, 318	$3,00E+05$	0	**8,6763**	7, 171	$3,00E+05$	0

TABLE III: Hypotheses analysis of the HMHH algorithm versus its constituent algorithms.

Algorithm	HMHH
CMAES	$25-3-0$
SaNSDE	$9-2-17$
GA	$3-3-22$
GCPSO	$7-1-20$
TOTAL	$39-18-55$

and bias the search towards CMAES. The inefficiency of the HMHH algorithm is then understandable since computational resources are required to first "learn" which algorithm is the best algorithm for the problem at hand.

V. Conclusion

This paper has investigated three alternative multi-method algorithms using the same set of constituent algorithms. The heterogenous meta-hyper-heuristic algorithm was shown to outperform the population-based portfolio algorithm and AUC-Bandit approach and also compared favourably to its constituent algorithms.

Future work could focus on extending this study to include other algorithms such as AMALGAM [11] and one or two of the ensemble algorithms developed in the differential evolution domain [14]. A more in-depth analysis of the impact of various entity-to-algorithm allocation features on multi-method algorithm performance could also be useful.

References

[1] X. Chen, Y. Ong, M. Lim, and K. Tan, "A multi-facet survey on memetic computation," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 5, pp. 591–607, 2011.

[2] C. P. Gomes and B. Selman, "Algorithm portfolios," *Artificial Intelligence*, vol. 126, no. 1–2, pp. 43–62, 2001.

[3] T. G. Dietterich, "Ensemble methods in machine learning," *Lecture notes in computer science*, pp. 1–15, 2000.

[4] 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.

[5] F. Peng, K. Tang, G. Chen, and X. Yao,"Population-Based Algorithm Portfolios for Numerical Optimization," *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 5, pp. 782–800, 2010.

[6] A. Fialho, M. Schoenauer, and M. Sebag, "Fitness-AUC bandit adaptive strategy selection vs. the probability matching one within differential evolution: an empirical comparison on the BBOB-2010 noiseless testbed," *Proceedings of the GECCO 2010 Workshop on Black-Box Optimization Benchmarking*, 2010.

[7] J. Grobler, A. P. Engelbrecht, G. Kendall, and V. S. S. Yadavalli, "Investigating the impact of alternative evolutionary selection strategies on multi-method global optimization," *Proceedings of the 2011 IEEE Congress on Evolutionary Computation*, pp. 2337-2344, 2011.

[8] W. E. Hart, N. Krasnogor, and J. E. Smith, "Recent Advances in Memetic Algorithms," *Springer-Verlag*, 2005.

[9] A. K. Qin and P. N. Suganthan,"Self-adaptive differential evolution algorithm for numerical optimization," *Proceedings of the 2005 IEEE Congress on Evolutionary Computation*, pp. 1785–1791, 2005.

[10] O. Olorunda and A. P. Engelbrecht, "An Analysis of Heterogeneous Cooperative Algorithms," *Proceedings of the 2009 IEEE Congress on Evolutionary Computation*, pp. 1562–1569, 2009.

[11] J. A. Vrugt, B. A. Robinson, and J. M. Hyman, "Self-Adaptive Multimethod Search for Global Optimization in Real-Parameter Spaces," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 243–259, 2009.

[12] W. Gong, A. Fialho, and Z. Cai, "Adaptive strategy selection in differential evolution," *Proceedings of the 2010 Genetic and Evolutionary Computation Conference*, 2010.

[13] A. Fialho, R. Ros, M. Schoenauer, and M. Sebag, "Comparison-based Adaptive Strategy Selection in Differential Evolution," *Proceedings of the 2011 Genetic and Evolutionary Computation Conference*, 2011.

[14] R. Mallipeddi, P. N. Suganthan, Q. K. Pan, and M. F. Tasgetiren "Differential evolution algorithm with ensemble of parameters and mutation strategies," *Applied Soft Computing*, vol. 11, pp. 1679–1696, 2011.

[15] J. Zhong, M. Shen, J. Zhang, H. Chung, Y. Shi, and Y. Li "A Differential Evolution Algorithm with Dual Populations for Solving Periodic Railway Timetable Scheduling Problem," *IEEE Transactions on Evolutionary Computation*, In press.

[16] J. Tvrdik "Modifications of Differential Evolution with Composite Trial Vector Generation Strategies," *Soft Computing Models in Industrial and Environmental Applications Advances in Intelligent Systems and Computing*, vol. 188, pp. 113-122, 2013.

[17] E. K. Burke, M. Hyde, G. Kendall, G. Ochoa, E. Ozcan, and R. Qu, "Hyper-heuristics: A survey of the state of the art," tech. rep., University of Nottingham, 2010.

[18] E. K. Burke, M. Hyde, G. Kendall, G. Ochoa, E. Ozcan, and J. R. Woodward, "A Classification of Hyper-heuristic Approaches," *International Series in Operations Research and Management Science*, In M. Gendreau and J-Y Potvin (Eds.), Springer (in press).

[19] K. A. Dowsland, E. Soubeiga, and E. K. Burke, "A simulated annealing based hyperheuristic for determining shipper sizes for storage and transportation," *European Journal of Operational Research*, vol. 179, pp. 759–774, 2007.

[20] L. J. Eshelman and J. D. Schaffer, "Real-coded genetic algorithms and interval schemata," In D. Whitley, editor, *Foundations of Genetic Algorithms*, vol. 2, pp. 187–202, 1993.

[21] F. Van den Bergh and A. P. Engelbrecht, "A new locally convergent particle swarm optimiser," *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, vol. 3, pp. 6–12, 2002.

[22] Z. Yang, K. Tang, and X. Yao, "Self-adaptive Differential Evolution with Neighbourhood Search," *Proceedings of the 2008 IEEE Congress on Evolutionary Computation*, pp. 1110-1116, 2008.

[23] A. Auger, and N. Hansen, "A Restart CMA evolution strategy With increasing population size," *Proceedings of the 2005 IEEE Congress on Evolutionary Computation*, pp. 1769-1776, 2005.

[24] D. Hadka and P. Reed "Borg: An auto-adaptive Many-Objective Evolutionary Computing Framework," *Evolutionary Computation*, 2012.

[25] P. N. Suganthan, N. Hansen, J. J. Liang, K. Deb, Y. P. Chen, A. Auger, S. Tiwari, "Problem Definitions and Evaluation Criteria for the CEC 2005 Special Session on Real-Parameter Optimization," *Technical Report, Nanyang Technological University, Singapore and KanGAL Report Number 2005005 (Kanpur Genetic Algorithms Laboratory, IIT Kanpur)*, 2005.