Investigating the Use of Local Search for Improving Meta-Hyper-Heuristic Performance

Jacomine Grobler
Department of Industrial and Systems Engineering
University of Pretoria and Denel Dynamics Pty (Ltd).
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 use of local search strategies to improve the performance of a meta-hyper-heuristic algorithm, a hyper-heuristic which employs one or more meta-heuristics as low-level heuristics. Alternative mechanisms for selecting the solutions to be refined further by means of local search, as well as the intensity of subsequent refinement in terms of number of allowable function evaluations, are investigated. Furthermore, defining a local search as one of the low-level heuristics versus applying the algorithm directly to the solution space is also investigated. Performance is evaluated on a diverse set of floating-point benchmark problems. The addition of local search was found to improve algorithm results significantly. Random selection of solutions for further refinement was identified as the best selection strategy and a higher intensity of refinement was identified as most desirable. Better results were obtained by applying the local search algorithm directly to the search space instead of defining it as a low-level heuristic.**

I. Introduction

During the last two decades the hybridization of more than one solution strategy has received increasing research attention. One of the earliest examples of this hybridization can be found in the field of memetic computation. Memetic algorithms were first defined as being the algorithmic pairing of a population-based search method with one or more refinement methods [1].

Meta-heuristics are well known for their robustness and ability to avoid local optima. However, room for improvement exists with regard to a meta-heuristic's ability to successfully exploit good solutions further [2]. The hybridization of a meta-heuristic algorithm with one or more refinement methods can be useful to balance the trade-off between exploration and exploitation.

Another emerging trend in optimization has also been developing at the same time: hyper-heuristics 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 control 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.

Similar to the use of local search (LS) strategies to improve meta-heuristic performance [3], this paper investigates the use of local search strategies for improving hyper-heuristic performance. The Heterogeneous Meta-Hyper-Heuristic (HMHH) algorithm of Grobler *et al.* [4] is used as basis for further investigation. The HMHH algorithm makes use of five common meta-heuristic algorithms as low-level heuristics. The idea is that these low-level algorithms can be intelligently used at different stages of the optimization process by the high-level hyper-heuristic strategy to optimize different solutions in the search space.

Two strategies for using local search to improve performance of the HMHH algorithm are introduced. The first is based on defining a local search algorithm as an additional low-level heuristic of the HMHH algorithm. The second strategy applies the local search directly to the solution space at each HMHH algorithm iteration. In this second framework, the selection of entities for local search is done independently of the hyper-heuristic selection mechanism.

Within the context of the two frameworks, two additional memetic algorithm design issues are also investigated. The selection mechanism used to select entities for further exploitation by local search and the effect of the intensity of refinement, are also explored.

Performance is evaluated on a set of varied floating-point benchmark problems, and the second framework with direct application of the local search algorithm to the solution space is shown to be superior. Higher intensities of refinement are also found to have a more significant impact on algorithm performance. Finally, a comparison of the multi-method hyper-heuristic algorithm versus its low-level meta-heuristic sub-algorithms, is also shown to be robust and gives promising results.

Local search has already been considered to improve hyper-heuristic performance [5]. However, the cited investigation focused on using different local search heuristics as high level hyper-heuristics and no refinement intensity was considered. To the best of the authors' knowledge, this paper describes the first investigation of local search in conjunction with meta-heuristic based low-level heuristics in a hyper-heuristic framework.

The rest of the paper is organized as follows: Section II provides an overview of existing literature. Section III provides a brief overview of the HMHH algorithm while Section IV describes the local search based improvement strategies which were evaluated. The results are documented in Section V before the paper is concluded in Section VI.

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. Examples include memetic computation [1], algorithm portfolios [6], algorithm ensembles [7], and hyper-heuristics [8]. 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.

This section provides a brief review of the most promising multi-method algorithms which utilize more than one meta-heuristic algorithm during the optimization process. Since the paper is aimed at investigating the use of local search to improve multi-method algorithm performance, a number of memetic algorithm design issues will also be discussed. Finally, a number of existing local search hyper-heuristics are also presented.

A. Multi-method algorithms

In addition to hyper-heuristic research, a large amount of research has investigated multi-method techniques [9] [10].

The heterogeneous cooperative algorithm of Olorunda and Engelbrecht [11] is one of the first multi-method algorithms which dynamically assigns entities to algorithms during the course of an optimization run. The algorithm makes use of different evolutionary algorithms to update a number of sub-populations, in a cooperative algorithm framework, thereby combining the strengths and weaknesses of various optimization strategies within the same algorithm.

Peng *et al.* [12] developed the population-based algorithm portfolio. This algorithm is based on the principle of multiple sub-populations each assigned to one algorithm from a portfolio of available algorithms. At pre-specified time intervals, entities are migrated between sub-populations to ensure effective information sharing between the different optimization algorithms.

Vrugt *et al*'s [13] highly successful population-based genetic adaptive method for single-objective optimization (AMALGAM-SO) 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 three evolutionary algorithms. A restart strategy is used to update the percentages based on algorithm performance.

Another successful adaptive algorithm selection mechanism was investigated by Fialho *et al.* [14]. Comparisons of alternative credit assignment methods [15] and strategy selection mechanisms within a differential evolution framework [14], highlighted the superior performance of the fitness-based area-under-curve bandit technique with a rank-based reward scheme.

B. Memetic algorithm design issues

A number of key issues that need to be considered during memetic algorithm design are defined by [2] and [1]. Firstly, the number of individuals which should undergo refinement should be determined. The first memetic algorithms recommended that all individuals in the population should be refined at each iteration of the memetic algorithm [3]. Due to various constraints such as computational budget and the need to maintain a suitable level of diversity in the population, this strategy is not always desirable [9]. Secondly, the intensity of refinement should be considered. This issue relates to the computational budget that is allocated to the refining algorithm [16].

Various other memetic algorithm design issues such as the type of local search algorithm employed, integration of local search with existing evolutionary operators [17], and Lamarckian versus Baldwinian learning [18], could also be considered. These strategies will, however, not be explicitly investigated in this paper.

C. Local search and hyper-heuristics

A number of different strategies have already been used in the hyper-heuristic literature to exploit the benefits of local search algorithms to improve hyper-heuristic algorithm performance:

Firstly, perturbative hyper-heuristics aim to improve a candidate solution through a process of automatically selecting and applying one of a set of available heuristics to an existing candidate solution [19]. A number of local search strategies have already been used as high-level hyper-heuristic strategies. In other words, these hyper-heuristics consists of a local search algorithm which manipulates a number of low level algorithms. A detailed review of a large number of perturbative hyper-heuristics is provided in [19].

Secondly, local search algorithms can also be incorporated into the set of available low-level heuristics [21]. This option can be considered an intervention in heuristic space diversity, especially when metaheuristics are utilized as low-level heuristics, since a more diverse set of algorithms are made available to the high-level strategy.

Finally, local search can be applied directly to the solution space. A good example of this is Qu and Burke's graph based hyper-heuristic framework [5] where a local search algorithm operates directly on the solution space in conjunction with a hyper-heuristic strategy which operates in heuristic space.

III. THE HETEROGENEOUS META-HYPER-HEURISTIC ALGORITHM

The Tabu-search based HMHH algorithm (Figure 1) [4] consists of a common population of entities with each representing a candidate solution which is evolved over time, a set of low-level meta-heuristic sub-algorithms, an acceptance strategy, and a selection strategy.

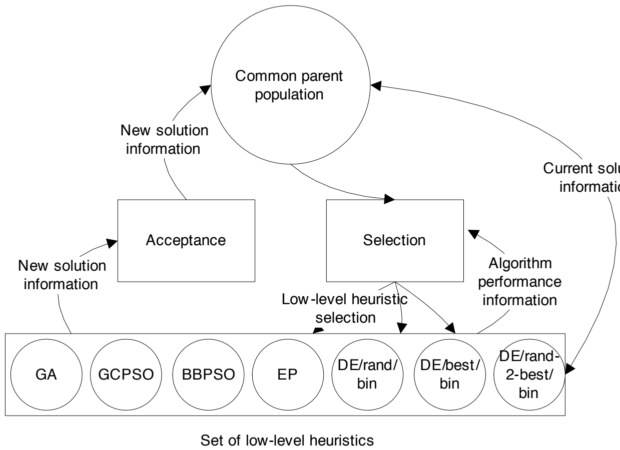

Fig. 1. The heterogeneous meta-hyper-heuristic.

The idea is that an intelligent algorithm can be evolved which selects the appropriate meta-heuristic 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 algorithm-assignment is maintained for k iterations, while the common parent population is continuously updated with new information and better solutions. Re-allocation of entities to algorithms is then performed, and the process continues.

This strategy was considered promising since each entity can use unique 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.

This paper uses five common meta-heuristic sub-algorithms as the set of low-level heuristics:

- A genetic algorithm with a floating-point representation, tournament selection, blend crossover [20] [11], and self-adaptive gaussian mutation [4].
- The guaranteed convergence particle swarm optimization algorithm (GCPSO) [22].
- Two differential evolution algorithms (*DE/best/bin* and *DE/rand-to-best/bin*) [23].
- The covariance matrix adapting evolutionary strategy algorithm (CMAES) [24].

For all results described in this paper, the Broyden-Fletcher-Goldfarb-Shanno (BFGS) Quasi-Newton method with a cubic line search procedure as implemented in Matlab's optimization toolbox, was utilized.

IV. Investigating alternative local search improvement strategies

Two design issues relating to the use of local search as a means of improving the ability of an algorithm to exploit good solutions in the search space, are investigated in this paper. Firstly, the selection of algorithm entities for the application of local search and secondly, the intensity of the local search.

A. Entity selection for local search

Four selection mechanisms for selecting entities to which a local search algorithm is to be applied, were investigated. The number of entities to which local search should be applied was defined *a priori* for the first three algorithms. At each iteration a selection mechanism independent of the hyper-heuristic is then applied to the solution space to select the entities to be exploited.

- LS1HH - local search is applied to only the best entity of each iteration.
- LS2HH - local search is applied to a single randomly selected entity at each iteration.
- LS3HH - roulette wheel selection is applied to the entire population to select an entity at each iteration.

Applying the local search algorithm to the best performing entity is thought to be productive since the assumption is that this entity has a greater probability of being positioned in the basin of attraction of a global optimum. As defined in this paper, LS1HH has the highest selective pressure and because LS3HH is based on roulette-wheel selection it has a lower selective pressure than LS1HH, but a higher selective pressure than LS2HH. LS2HH, with the lowest selective pressure, maintains the highest solution space diversity for the longest period of time.

The final algorithm is an implementation of LSHH where no local search is applied directly to the search space. LS4HH makes the local search algorithm available for selection and application to the algorithm entities by defining the local search as one of the low-level heuristics. The high level hyper-heuristic strategy is thus responsible for selecting the number of entities per iteration, as well as the specific entities of the population to which the local search algorithm should be applied.

B. Investigating intensity of refinement

The effect of local search intensity was investigated by setting the maximum number of local search iterations allowed to the local search algorithm per hyper-heuristic algorithm iteration. Two extreme scenarios were investigated: an upper bound of 100 000 iterations where the local search is basically run to convergence (LSNHH (long)), and a lower bound of 10 iterations (LSNHH (short)), which were usually too few for the LS to converge. The best performing memetic-like strategy, namely LS2HH, as will be shown later from the results obtained, as well as the hyper-heuristic incorporating LS as low level heuristic, namely LS4HH, were used to investigate the impact of LS intensity on algorithm performance.

V. Empirical results

The various strategies were evaluated on the first 17 problems of the 2005 IEEE Congress of Evolutionary Computation benchmark problem set, which enables algorithm performance evaluation on both unimodal and multimodal functions and includes various expanded and hybridized problems, some with noisy fitness functions.

The algorithm control parameter values listed in Table I were found to work well for the algorithms under study during previous research by the authors. $m \longrightarrow n$ indicates that the associated parameter is decreased linearly from m to n over 95% of the maximum number of iterations, I_{max}.

TABLE I
HMHH ALGORITHM PARAMETERS.

Parameter	**Value used**
Number of entities in common population (n_s)	100
Number of iterations between re-allocation (k)	5
Size of tabu list	3
PSO parameters	
Acceleration constant (c_1)	$2.0 \longrightarrow 0.7$
Acceleration constant (c_2)	$0.7 \longrightarrow 2.0$
Inertia weight (w)	$0.9 \longrightarrow 0.4$
DE parameters	
Probability of reproduction (p_r)	$0.75 \longrightarrow 0.25$
Scaling factor (F)	$0.75 \longrightarrow 0.125$
GA parameters	
Probability of crossover (p_c)	$0.6 \longrightarrow 0.4$
Probability of mutation (p_m)	0.1
Blend crossover parameter (α)	0.5
GA tournament size (N_t)	13
CMAES parameters	As specified in [24].

The results of the first local search based improvement strategy comparison is presented in Table III, where the results for each strategy were recorded over 30 independent simulation runs. μ and σ denote the mean and standard deviation associated with the corresponding performance measure and #FEs denotes the number of function evaluations which were needed to reach the global optimum. 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 highlighted in Table III.

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 outperforms the second strategy, performs similarly, or is outperformed by the second strategy, is recorded. The results in Table II are subsequently provided in the form: “Number of wins, number of draws, number of losses”. To illustrate, (15-15-21) in row 1 column 2, indicates that LS1HH outperformed LS2HH 15 times over the benchmark problem set.

From the results it is clear that applying the local search algorithm directly to the search space independently of the hyper-heuristic is a better strategy than defining the algorithm as a low level heuristic. Due to the line search included in a single iteration of the local search algorithm, the algorithm is computationally very expensive and consumes a large number of function evaluations per iteration. Whereas LS1HH to LS3HH limits the number of entities which can

TABLE II
HYPOTHESES ANALYSIS OF ALTERNATIVE LOCAL SEARCH SELECTION STRATEGIES.

	LS1HH	**LS2HH**	**LS3HH**
LS1HH	NA	$15-15-21$	$15-17-19$
LS2HH	$21-15-15$	NA	$14-37-0$
LS3HH	$19-17-15$	$0-37-14$	NA
LS4HH	$3-7-41$	$2-2-47$	$3-2-46$
	LS4HH	**TOTAL**	
LS1HH	$41-7-3$	$71-39-43$	
LS2HH	$47-2-2$	$82-54-17$	
LS3HH	$46-2-3$	$65-56-32$	
LS4HH	NA	$8-11-134$	

be exploited by means of local search to one, no such restrictions are placed on LS4HH. It is thus suspected that a larger computational budget is used earlier during the optimization run in the LS4HH algorithm when compared to the other three algorithms. This early exploitation could have disastrous consequences on population diversity and algorithm performance.

LS2HH is the best performing algorithm. This algorithm is also the algorithm with the least selective pressure and slowest convergence rate.

The results of the investigation into refinement intensity is presented in Table IV and a statistical analysis is provided in Table V. Mann Whitney U tests were again used to complete a pairwise comparison of each strategy versus all of the other strategies. The same “number of wins, draws and losses” format of Table II was also used.

TABLE V
HYPOTHESES ANALYSIS OF LOCAL SEARCH STRATEGIES OF DIFFERENT INTENSITY.

	LS2HH (short)	**LS2HH (long)**
TSHH (No local search)	$21-28-2$	$15-16-20$
LS2HH (short)	NA	$13-15-23$
LS2HH (long)	$23-15-13$	NA
LS4HH (short)	$1-18-32$	$7-10-34$
LS4HH (long)	$10-4-37$	$2-2-47$
	LS4HH (short)	**LS4HH (long)**
TSHH (No local search)	$33-17-10$	$38-3-10$
LS2HH (short)	$32-18-1$	$37-4-10$
LS2HH (long)	$34-10-7$	$47-2-2$
LS4HH (short)	NA	$34-5-12$
LS4HH (long)	$12-5-34$	NA
	TSHH (No local search)	**TOTAL**
TSHH (No local search)	NA	$107-64-33$
LS2HH (short)	$2-28-21$	$84-65-55$
LS2HH (long)	$20-16-15$	$124-43-37$
LS4HH (short)	$1-17-33$	$43-50-111$
LS4HH (long)	$10-3-38$	$34-14-156$

A number of conclusions can be drawn from the results in Table V. Firstly, refinement intensity had little impact on the performance of LS2HH versus LS4HH, with LS2HH

TABLE III

RESULTS OF ALTERNATIVE LOCAL SEARCH SELECTION STRATEGY EVALUATION ON THE 2005 IEEE CEC BENCHMARK PROBLEM SET.

Prob	Dims	LS1HH				LS2HH				LS3HH				LS4HH			
		FFV		# FEs		FFV		# FEs		FFV		# FEs		FFV		# FEs	
		μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
1	10	1E − 06	0	**149.5**	5.594	1E − 06	0	154.63	4.5523	1E − 06	0	155.37	2.0083	1E−06	0	1193.8	188
1	30	1E − 06	0	**256.03**	5.6598	1E − 06	0	275.67	14.863	1E − 06	0	269.47	15.73	1E−06	0	3581.7	658.22
1	50	1E − 06	0	**385.6**	25.412	1E − 06	0	400.9	15.562	1E − 06	0	397.5	19.332	1E−06	0	6257.4	1122.3
2	10	1E − 06	0	**764.03**	55.301	1E − 06	0	1471.5	771.4	1E − 06	0	2345.8	1528.5	1E−06	0	14727	3136.4
2	30	1E − 06	0	**8728.3**	2389.3	1E − 06	0	1.32e+05	45352	1E − 06	0	2.42e+05	42439	1E−06	0	1.65e+05	53451
2	50	**1E − 06**	0	56907	52167	0.113	0.2061	5E + 05	0	109.83	62.77	5E+05	0	3.2327	8.3786	5E + 05	8717.6
3	10	1E − 06	0	**50115**	9559.5	1E − 06	0	83478	12136	0.0050	0.0256	96270	5291.7	1E−06	0	1E + 05	1832.4
3	30	**96.776**	407.81	3E + 05	325.7	83289	56128	3E + 05	0	2.03e+05	1.04e+05	3E+05	0	4.12e+05	3.44e+05	3E + 05	8406.3
3	50	**8798.4**	13315	5E + 05	0	3.74e+05	1.23e + 05	5E + 05	0	1.15e+06	4.09e+05	5E+05	0	6.92e+05	4.34e+05	5E + 05	9862
4	10	1E − 06	0	42397	1943.7	1E − 06	0	**41866**	1890.9	1E − 06	0	42198	3182.6	3.3443	12.069	1E + 05	395.54
4	30	**44.071**	74.218	3E + 05	180.03	85.901	125.9	3E + 05	167.53	46.168	80.377	3E+05	172.62	29763	9271.8	3E + 05	1902.6
4	50	10412	6390.2	5E + 05	202.17	9004.1	5953.5	5E + 05	202.99	**6993.8**	4870.8	5E+05	198.92	1.04e+05	21613	5E + 05	1768.3
5	10	1E − 06	0	**56301**	6147.3	1E − 06	0	68475	5301.5	1E − 06	0	68458	7310.5	46.676	59.338	1E + 05	4148.2
5	30	4203.3	1167.6	3E + 05	352.51	**1710.5**	582.76	3E + 05	338.16	1867.5	730.96	3E+05	358.86	4434.7	932.4	3E + 05	7927
5	50	10011	1746.7	5E + 05	284.87	5617.8	1398.5	5E + 05	76.436	**5303.6**	1182.5	5E+05	74.855	14672	2684	5E + 05	8761.9
6	10	**0.0093**	0.0025371	6584	14674	0.0096667	0.0018	15119	7659.5	0.0097	0.0019	33342	5796.3	0.01	0	46255	10722
6	30	0.6716	1.5048	**78217**	1.08e+05	19.345	55.169	2.32e+05	61954	46.962	80.321	3E+05	4509.6	9.1657	19.445	2.49e+05	56597
6	50	**32.651**	91.223	1.74e+05	1.74e+05	39.68	62.143	5E + 05	0	146.34	157.63	5E+05	0	143.66	230.85	5E + 05	8850.7
7	10	0.182	0.26759	96691	18060	0.004	0.005	**14208**	9505.2	0.005	0.0051	17262	19114	0.003	0.0047	49388	27254
7	30	0.0067	0.0047946	71742	1.28e+05	0.007	0.005	**16209**	8010	0.0067	0.0048	34377	6885.5	0.0087	0.0035	48292	19762
7	50	0.003	0.0046609	89693	1.87e+05	0.007	0.005	**34683**	14293	0.007	0.0047	70224	23563	0.006	0.0050	2.19e+05	1.74e+05
8	10	**19.99**	0	1E + 05	220.02	**19.99**	0	1E + 05	220.82	**19.99**	0	1E+05	224.11	**19.99**	0	1E + 05	2969.8
8	30	20.016	0.053529	3E + 05	438.54	20.049	0.0177	3E + 05	0	20.043	0.0196	3E+05	0	**19.991**	0.0035	3E + 05	9307.7
8	50	20.013	0.049753	5E + 05	265.99	20.183	0.0330	5E + 05	0	20.174	0.0337	5E+05	0	**20.004**	0.0206	5E + 05	13829
9	10	0.36867	0.65622	67276	30298	0.13833	0.3358	**62464**	23540	0.203	0.3952	65362	27423	9.1073	6.1478	1E + 05	2569.9
9	30	22.773	9.9239	3E + 05	82.158	10.832	3.707	3E + 05	160.83	**10.436**	3.9601	3E+05	87.035	151.3	54.629	3E + 05	3717.8
9	50	81.445	26.174	5E + 05	136.7	51.064	9.0906	5E + 05	232.71	**49.007**	10.903	5E+05	218.02	404.29	121.48	5E + 05	7231.4
10	10	27.384	14.667	99998	50.873	4.565	2.404	**99982**	54.319	6.09	2.8596	99993	52.935	19.298	8.2907	1E + 05	1073.9
10	30	162.03	66.908	3E + 05	159.9	**24.232**	6.2578	3E + 05	263.61	39.224	14.287	3E+05	164.32	252.74	95.446	3E + 05	4401.7
10	50	227.1	102.01	5E + 05	264.73	**65.226**	15.256	5E + 05	256.58	82.608	19.954	5E+05	305.64	572.45	194.21	5E + 05	6441.6
11	10	**4.4361**	1.6362	1E + 05	85.615	5.8842	2.2351	1E + 05	105.01	5.1001	2.1869	99097	5495.3	10.041	1.5166	1E + 05	1536.1
11	30	**25.092**	6.7628	3E + 05	192.01	26.826	4.7552	3E + 05	186.49	27.853	4.4759	3E+05	167.02	44.364	1.5443	3E + 05	5771.2
11	50	51.336	6.2922	5E + 05	306.86	**50.423**	6.1459	5E + 05	269.63	55.657	10.93	5E+05	245.45	79.815	1.8393	5E + 05	8337
12	10	238.41	555.24	33930	47577	0.01	0	**1435**	1022.4	0.01	0	1466.8	1047.5	155.62	469.09	55921	43888
12	30	1970.2	2643.7	3E + 05	0	**33.987**	56.522	3E + 05	2050.3	64.052	70.784	3E+05	0	1318.9	1671.1	3E + 05	6905.2
12	50	20170	22844	5E + 05	0	**2004.9**	1672.9	5E + 05	0	2489.2	1623.2	5E+05	0	9476.5	5133.5	5E + 05	10011
13	10	0.629	0.23178	1E + 05	50.847	**0.32633**	0.1145	1E + 05	200.95	0.4083	0.1298	1E+05	224.07	4.7887	2.677	1E + 05	1854.2
13	30	6.8327	2.8308	3E + 05	276.34	**2.851**	0.7777	3E + 05	0	3.0553	0.8812	3E+05	0	66.441	44.8	3E + 05	8159
13	50	24.447	12.001	5E + 05	289.94	**11.667**	3.092	5E + 05	0	13.831	6.027	5E+05	0	112.99	59.385	5E + 05	9885.5
14	10	3.537	0.2788	1E + 05	112.14	**3.3867**	0.3873	1E + 05	259.34	3.5487	0.3348	1E+05	261.33	4.0563	0.2301	1E + 05	2741.1
14	30	13.245	0.22198	3E + 05	441.9	12.646	0.5535	3E + 05	0	**12.619**	0.4674	3E+05	0	14.1	0.2288	3E + 05	9158.9
14	50	23.034	0.26587	5E + 05	286.63	22.246	0.6296	5E + 05	0	**22.234**	0.3865	5E+05	0	23.899	0.1925	5E + 05	10318
15	10	310.66	181.89	1E + 05	101.45	**172.49**	162.07	1E + 05	135.15	264.13	190.21	98994	6011.3	515.96	125.08	1E + 05	1891.2
15	30	**336.28**	107.24	3E + 05	181.3	358.48	123.71	3E + 05	254.94	383.64	114.22	3E+05	223.68	557.4	99.355	3E + 05	4803.3
15	50	**290**	87.171	5E + 05	315.59	308	91.376	5E + 05	289.18	714.12	259.27	5E+05	2.33e+05	778.49	162.55	5E + 05	2.51e+05
16	10	151.95	35.447	1E + 05	101.3	**122.69**	14.34	1E + 05	100.8	134.54	20.206	1E+05	121.72	218.49	49.955	1E + 05	1778.1
16	30	286.15	130.87	3E + 05	241.47	232.86	161.57	3E + 05	262.84	**188.07**	140.04	3E+05	310.61	338.92	96.303	3E + 05	5104.7
16	50	269.98	79.703	5E + 05	287.65	**117.07**	61.452	5E + 05	296.84	436.68	355.17	5E+05	2.45e+05	666.71	269.17	5E + 05	2.53e+05
17	10	130.43	27.247	1E + 05	78.12	135.86	29.211	1E + 05	78.65	**124.2**	21.231	1E+05	81.651	195.82	38.976	1E + 05	608.06
17	30	**211.77**	165.87	3E + 05	188.83	236.14	184.39	3E + 05	169.31	254.42	174.23	3E+05	146.36	394.65	109.29	3E + 05	2030.3
17	50	**355.03**	101.96	5E + 05	270.56	356.2	89.104	5E + 05	200.17	549.49	260.59	5E+05	2.45e+05	696.19	216.87	5E + 05	2.53e+05

TABLE IV

RESULTS OF THE INVESTIGATION INTO LOCAL SEARCH STRATEGIES OF DIFFERENT INTENSITIES (2005 IEEE CEC BENCHMARK PROBLEM SET).

Prob	Dims	TSHH (No local search)				LS2HH (short)				LS2HH (long)				LS4HH (short)				LS4HH (long)			
		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	12320	498.55	1E − 06	0	13490	800.82	1E − 06	0	**154.63**	4.5523	1E−06	0	16794	1603	1E − 06	0	1193.8	188
1	30	1E−06	0	43837	3322.2	1E − 06	0	55793	3569.8	1E − 06	0	**275.67**	14.863	1E−06	0	1.08e+05	14935	1E − 06	0	3581.7	658.22
1	50	1E−06	0	68217	3912.6	1E − 06	0	1.02e+05	6747	1E − 06	0	**400.9**	15.562	1E−06	0	2.21e+05	46387	1E − 06	0	6257.4	1122.3
2	10	1E−06	0	13747	851.67	1E − 06	0	15440	1090.8	1E − 06	0	**1471.5**	771.4	1E−06	0	23307	3198.3	1E − 06	0	14727	3136.4
2	30	1E−06	0	**65860**	9423.5	1E − 06	0	88678	13422	1E − 06	0	1.32e+ 05	45352	1.8713	5.1876	2.85e+05	22543	1E − 06	0	1.65e+05	53451
2	50	1E−06	0	**1.22e + 05**	13223	1E − 06	0	1.92e+05	21388	0.113	0.2061	5E+05	0	2700.4	1841.6	5E + 05	358.57	3.2327	8.3786	5E + 05	8717.6
3	10	1E−06	0	**20297**	2077.4	1E − 06	0	22807	1945.1	1E − 06	0	83478	12136	87.953	481.74	52895	21498	1E − 06	0	1E + 05	1832.4
3	30	1E−06	0	**1.59e + 05**	17036	1E − 06	0	2.07e+05	21091	83289	56128	3E+05	0	1.48e+ 06	7.88e+ 05	3E + 05	230.98	4.12e+05	3.44e+05	3E + 05	8406.3
3	50	1E−06	0	**3.50e + 05**	26481	227.78	1227.1	4.95e+05	9681.8	3.74e+05	1.23e+05	5E+05	0	6.57e+ 06	2.82e+ 06	5E + 05	295.29	6.92e+05	4.34e+05	5E + 05	9862
4	10	1E−06	0	**15530**	885.96	1E − 06	0	17312	952.4	1E − 06	0	41866	1890.9	1E−06	0	29189	4890.6	3.3443	12.069	1E + 05	395.54
4	30	1E−06	0	**95663**	8584.6	1E − 06	0	1.21e+05	17135	85.901	125.9	3E+05	167.53	199.99	274.15	3E + 05	219.68	29763	9271.8	3E + 05	1902.6
4	50	6.249	26.283	**2.16e + 05**	79336	25.427	134.43	3.14e+05	58658	9004.1	5953.5	5E+05	202.99	27284	12279	5E + 05	378.96	1.04e+05	21613	5E + 05	1768.3
5	10	1E−06	0	**16573**	1025.2	1, 0E−06	0	18296	1032.7	1E − 06	0	68475	5301.5	1E−06	0	22198	1767	46.676	59.338	1E + 05	4148.2
5	30	**402.7**	426.57	3E + 05	0	503.9	555.52	3E + 05	0	1710.5	582.76	3E+05	338.16	2446.8	1156.3	3E + 05	132.63	4434.7	932.4	3E + 05	7927
5	50	5317	1239.5	5E + 05	0	**5222.7**	1275.4	5E + 05	0	5617.8	1398.5	5E+05	76.436	8039	2122	5E + 05	347.52	14672	2684	5E + 05	8761.9
6	10	0.0013	0.0035	31027	15074	0.001	0.0031	30914	12132	0.0097	0.0018	**15119**	7659.5	0.1327	0.7267	46557	13835	0.01	0	46255	10722
6	30	0.796	1.6192	**2, 00E + 05**	58393	1.1833	1.997	2.38e+05	53324	19.345	55.169	2.32e+ 05	61954	10.064	19.04	2.92e+05	24396	9.1657	19.445	2.49e+05	56597
6	50	1.071	1.8069	**4.01e + 05**	86423	8.731	16.868	4.83e+05	44413	39.68	62.143	5E+05	0	71.042	50.153	5E + 05	207.81	143.66	230.85	5E + 05	8850.7
7	10	0.1113	0.0715	99900	0	0.119	0.0894	97061	16152	0.004	0.005	**14208**	9505.2	0.126	0.1062	1E + 05	44.624	0.003	0.0047	49388	27254
7	30	0.0073	0.0101	1.74e+05	1.28e+05	0.008	0.0110	2.04e+05	1.1921e+ 05	0.007	0.005	**16209**	8010	0.0113	0.0155	2.21e+05	1.07e+05	0.0087	0.0035	48292	19762
7	50	0.0047	0.0078	2.41e+05	2E + 05	0.0033	0.0076	2.55e+05	1.7661e+ 05	0.007	0.005	**34683**	14293	0.002	0.0055	2.32e+05	1.37e+05	0.006	0.0050	2.19e+05	1.74e+05
8	10	20.069	0.1237	**99900**	0	20.073	0.137	1.0001e+ 05	0	19.99	0	1E+05	220.82	20.094	0.1417	1E + 05	71.097	19.99	0	1E + 05	2969.8
8	30	20.144	0.1400	3E + 05	0	20.139	0.0960	3E + 05	0	20.049	0.0177	3E+05	0	20.847	0.2659	3E + 05	281.58	**19.991**	0.0035	3E + 05	9307.7
8	50	20.649	0.3782	5E + 05	0	20.889	0.3932	5E + 05	0	20.183	0.0330	5E+05	0	21.191	0.0558	5E + 05	535.84	**20.004**	0.0206	5E + 05	13829
9	10	0.4000	0.6626	72400	27113	0.562	0.761	76086	28946	0.1383	0.3358	**62464**	23540	0.8583	1.0293	81788	27626	9.1073	6.1478	1E + 05	2569.9
9	30	**8.6423**	4.6861	3E + 05	0	10.693	3.235	3E + 05	0	10.832	3.707	3E+05	160.83	17.501	5.4115	3E + 05	135.68	151.3	54.629	3E + 05	3717.8
9	50	**38.811**	7.2198	5E + 05	0	48.852	10.012	5E + 05	0	51.064	9.0906	5E+05	232.71	62.871	12.752	5E + 05	153.51	404.29	121.48	5E + 05	7231.4
10	10	14.535	4.7969	99900	0	15.422	8.3693	1E + 05	0	**4.565**	2.404	99982	54.319	16.084	7.6694	1E + 05	66.162	19.298	8.2907	1E + 05	1073.9
10	30	70.644	26.411	3E + 05	0	75.039	23.797	3E + 05	0	**24.232**	6.2578	3E+05	263.61	77.916	33.869	3E + 05	120.58	252.74	95.446	3E + 05	4401.7
10	50	117.32	35.389	5E + 05	0	128.82	47.701	5E + 05	0	**65.226**	15.256	5E+05	256.58	247	125.17	5E + 05	299.53	572.45	194.21	5E + 05	6441.6
11	10	4.6779	2.4914	**91957**	24238	5.1947	2.2514	97442	14064	5.8842	2.2351	1E+05	105.01	5.2403	2.1023	1E + 05	50.878	10.041	1.5166	1E + 05	1536.1
11	30	**26.79**	5.4967	3E + 05	0	26.969	4.6599	3E + 05	0	26.826	4.7552	3E+05	186.49	27.933	4.5815	3E + 05	203.63	44.364	1.5443	3E + 05	5771.2
11	50	50.268	6.3932	5E + 05	0	**48.379**	6.1125	5E + 05	0	50.423	6.1459	5E+05	269.63	67.901	8.5467	5E + 05	453.22	79.815	1.8393	5E + 05	8337
12	10	292.45	562.84	52067	40045	296.43	596.43	67362	38862	0.01	0	**1435**	1022.4	130.13	408.81	56814	29863	155.62	469.09	55921	43888
12	30	6118.9	6945.3	3E + 05	0	6902.7	5513.4	3E + 05	0	**33.987**	56.522	3E+05	2050.3	9188.7	9492.5	3E + 05	139.28	1318.9	1671.1	3E + 05	6905.2
12	50	68473	50901	5E + 05	0	49169	40381	5E + 05	0	**2004.9**	1672.9	5E+05	0	61523	41353	5E + 05	112.9	9476.5	5133.5	5E + 05	10011
13	10	0.5217	0.1545	99900	0	0.515	0.1524	1E + 05	0	**0.32633**	0.1145	1E+05	200.95	0.5763	0.20373	1E + 05	70.902	4.7887	2.677	1E + 05	1854.2
13	30	**2.7503**	0.9714	3E + 05	0	2.762	0.7973	3E + 05	0	2.851	0.7777	3E+05	0	6.4013	5.2332	3E + 05	165.53	66.441	44.8	3E + 05	8159
13	50	**7.3933**	3.9263	5E + 05	0	7.5367	2.7399	5E + 05	0	11.667	3.092	5E+05	0	34.123	8.948	5E + 05	447.69	112.99	59.385	5E + 05	9885.5
14	10	3.606	0.3076	99900	0	3.6627	0.3194	1E + 05	0	**3.3867**	0.3873	1E+05	259.34	3.6693	0.2558	1E + 05	88.596	4.0563	0.2301	1E + 05	2741.1
14	30	13.252	0.3556	3E + 05	0	13.208	0.3387	3E + 05	0	**12.646**	0.5535	3E+05	0	13.12	0.334	3E + 05	251.59	14.1	0.2288	3E + 05	9158.9
14	50	22.72	0.4353	5E + 05	0	22.711	0.4547	5E + 05	0	**22.246**	0.6296	5E+05	0	23.069	0.392	5E + 05	320.54	23.899	0.1925	5E + 05	10318
15	10	277.04	181.97	98490	7462.8	312.98	162.01	97191	15442	**172.49**	162.07	1E+05	135.15	338.8	149.02	1E + 05	68.869	515.96	125.08	1E + 05	1891.2
15	30	335.14	109.56	3E + 05	0	**316.13**	103.69	3E + 05	0	358.48	123.71	3E+05	254.94	349.37	109.93	3E + 05	134.88	557.4	99.355	3E + 05	4803.3
15	50	275.97	96.874	5E + 05	0	290.31	77.027	5E + 05	0	308	91.376	5E+05	289.18	**264.56**	89.023	5E + 05	196.26	778.49	162.55	5E + 05	2.51e+05
16	10	123.85	16.844	99900	0	133.01	22.426	1E + 05	0	**122.69**	14.34	1E+05	100.8	133.77	24.509	1E + 05	49.782	218.49	49.955	1E + 05	1778.1
16	30	229.43	159.87	3E + 05	0	210.18	145.61	3E + 05	0	232.86	161.57	3E+05	262.84	**179.18**	129.13	3E + 05	143.09	338.92	96.303	3E + 05	5104.7
16	50	184.93	128.3	5E + 05	0	180.77	135.92	5E + 05	0	**117.07**	61.452	5E+05	296.84	252.96	137.07	5E + 05	348.33	666.71	269.17	5E + 05	2.53e+05
17	10	133.08	18.54	99900	0	**125.05**	23.043	1E + 05	0	135.86	29.211	1E+05	78.65	134.17	22.954	1E + 05	46.973	195.82	38.976	1E + 05	608.06
17	30	194.16	138.25	3E + 05	0	**157.25**	132.32	3E + 05	0	236.14	184.39	3E+05	169.31	262.86	180.85	3E + 05	118.01	394.65	109.29	3E + 05	2030.3
17	50	**131.88**	120.33	5E + 05	0	146.66	132.68	5E + 05	0	356.2	89.104	5E+05	200.17	705.55	251.94	5E + 05	2.39e+05	696.19	216.87	5E + 05	2.53e+05

remaining the better performing algorithm. Secondly, it appeared that a higher refinement intensity was preferred for the LS2HH algorithm whereas the LS4HH algorithm performed better at a lower refinement intensity. Another important point is that a high refinement intensity HH2LS algorithm outperformed a hyper-heuristic with no local search algorithm incorporated. This is significant and illustrates the value of adding local search capabilities to the HMHH algorithm.

In an attempt to verify the actual performance of the selection strategies, the best performing local search hyper-heuristic hybrid algorithm from the previous analyses (LS2HH (long)) was also compared under similar conditions to its composing algorithms. The actual results are recorded in Table VII. In Table VI Mann Whitney U tests were used to compare the performance of each composing algorithm to the LS2HH algorithm.

TABLE VI
HYPOTHESES ANALYSIS OF HMHH ALGORITHM VERSUS ITS COMPOSING ALGORITHMS.

Algorithm	**LS2HH (long)**
CMAES	$15 - 8 - 28$
DE/best/bin	$22 - 20 - 9$
DE/rand-to-best/bin	$24 - 19 - 8$
GA	$28 - 15 - 8$
GCPSO	$27 - 9 - 15$
TOTAL	$116 - 71 - 68$

From the results it can be seen that the HH algorithms perform statistically significantly better a large number of times when compared to six of its seven composing algorithms. CMAES performs better than the HH when solving unimodal problems, but the HH performance starts to improve in comparison with CMAES as problem size and complexity increases. An inspection of the algorithm ranks does, however, indicate that the LS2HH algorithm is able to identify CMAES as the best performing algorithm and bias the search towards CMAES. The inefficiency of the LS2HH algorithm is then understandable since computational resources are required to first "learn" which algorithm is the best algorithm for the problem at hand.

VI. CONCLUSION

This paper investigated the use of local search strategies to improve the performance of a meta-hyper-heuristic algorithm. Various issues such as the application of local search directly to the search space versus the heuristic space, the mechanisms used to select entities for further exploitation, as well as the effect of the intensity of refinement were investigated. Experimental results indicated that applying the local search directly to the solution space to a single randomly selected individual per iteration at a relatively high intensity is the best strategy. It was also shown that the LS2HH algorithm compared favourably to its composing algorithms.

Future work could focus on an adaptive local search mechanism to allow the algorithm to better adapt to the requirements of the problem at hand.

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] N. Krasnogor and J. Smith, "A tutorial for competent memetic algorithms: Model, taxonomy and design issues," *IEEE Transactions on Evolutionary Computation*, vol. 9, no. 5, pp. 474–488, 2005.

[3] P. Moscato, "On evolution, search, optimization, genetical algorithms and martial arts: Toward memetic algorithms," tech. rep., California Institute of Technology, 1989.

[4] J. Grobler, A. P. Engelbrecht, G. Kendall, and V. S. S. Yadavalli, "Alternative hyper-heuristic strategies for multi-method global optimization," *Proceedings of the 2010 IEEE World Congress on Computational Intelligence*, pp. 826–833, 2010.

[5] R. Qu and E. K. Burke, "Hybridisations within a graph based hyper-heuristic framework for university timetabling problems," *Journal of the Operational Research Society*, vol. 60, pp. 1273–1285, 2009.

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

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

[8] 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.

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

[10] A. P. Engelbrecht, "Heterogeneous particle swarm optimization," *Swarm Intelligence*, 2011.

[11] 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.

[12] 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.

[13] 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.

[14] 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.

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

[16] D. Sudholt, "Local search in evolutionary algorithms: The impact of the local search frequency," *Algorithms and Computation (Lecture Notes in Computer Science)*, no. 359–368, 2006.

[17] M. Land, *Evolutionary algorithms with local search for combinatorial optimization*. PhD thesis, University of California, San Diego, 1998.

[18] D. Whitley, V. S. Gordon, and K. Mathias, "Lamarckian evolution, the baldwin effect and function optimization," *PPSN III Proceedings of the International Conference on Evolutionary Computation*, pp. 6–15, 1994.

[19] 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.

[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] E. Ozcan, B. Bilgin, and E. E. Korkmaz, "A comprehensive survey of hyperheuristics," *Intelligent Data Analysis*, vol. 12, no. 1, pp. 1–21, 2008.

[22] 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.

[23] R. Storn and K. Price, "Differential evolution — a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, pp. 341–359, 1997.

[24] 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.

TABLE VII

COMPARISON RESULTS OF THE LSHH2 ALGORITHM VERSUS ITS COMPOSING ALGORITHMS ON THE 2005 IEEE CEC BENCHMARK PROBLEM SET.

Prob (Dims)	LS2HH FFV μ	LS2HH FFV σ	LS2HH # FEs μ	LS2HH # FEs σ	CMAES FFV μ	CMAES FFV σ	CMAES # FEs μ	CMAES # FEs σ	Best performing DE FFV μ	Best performing DE FFV σ	Best performing DE # FEs μ	Best performing DE # FEs σ	GA FFV μ	GA FFV σ	GA # FEs μ	GA # FEs σ	GCPSO FFV μ	GCPSO FFV σ	GCPSO # FEs μ	GCPSO # FEs σ
1(10)	1E − 06	0	**154.63**	4.5523	1E−06	0	8526.7	302.78	1E − 06	0	13850	354.04	1E − 06	0	18557	1582.4	138.4	6.8073	1E+05	0
1(30)	1E − 06	0	**275.67**	14.863	1E−06	0	19110	447.48	1E − 06	0	46510	1323.6	1E − 06	0	99297	4860.4	231.81	29.491	3E+05	0
1(50)	1E − 06	0	**400.9**	15.562	1E−06	0	26930	726.42	86.415	235.56	1.53e+05	1.58e+05	1E − 06	0	4.11e+05	20554	356.19	50.373	5E+05	0
2(10)	1E − 06	0	**1471.5**	771.4	1E−06	0	9156.7	286.1	8.5863	29.553	37093	21349	1.0873	1.54	1E + 05	0	544.1	1.5915	1E+05	0
2(30)	1E − 06	0	1.32e+05	45352	1E−06	0	**26783**	739.1	12.075	11.044	3E + 05	0	446.67	228.69	3E + 05	0	567	3.0169	3E+05	0
2(50)	0.113	0.2061	5E + 05	0	1E−06	0	**52903**	869.2	316.96	978.05	5E + 05	0	6142	2003.1	5E + 05	0	595.87	4.663	5E+05	0
3(10)	1E − 06	0	83478	12136	1E−06	0	**13320**	379.11	71395	77215	1E + 05	0	9.76e+05	1.05e+06	1E + 05	0	970.96	1411.8	99177	4509.6
3(30)	83289	56128	3E + 05	0	1E−06	0	**61173**	1387.4	6.3815e + 06	3.53e+06	3E + 05	0	5.93e+06	2.65e+06	3E + 05	0	10543	8705.2	3E+05	0
3(50)	3.74e+05	1.23e+05	5E + 05	0	1E−06	0	**1.57e + 05**	2244.2	1.7441e + 07	6.47e+06	5E + 05	0	1.47e+07	4.65e+06	5E + 05	0	92056	80840	5E+05	0
4(10)	1E − 06	0	41866	1890.9	1E−06	0	**9590**	283.27	1E − 06	0	32810	945.17	380.19	510.25	1E + 05	0	320.69	0.2081	1E+05	0
4(30)	85.901	125.9	3E + 05	167.53	1E−06	0	**29357**	570.35	55.601	42.726	3E + 05	0	15946	7051.1	3E + 05	0	325.45	1.5416	3E+05	0
4(50)	9004.1	5953.5	5E + 05	202.99	1E−06	0	**59607**	998.25	3508.7	1174.8	5E + 05	0	49819	11900	5E + 05	0	333.64	3.3105	5E+05	0
5(10)	1E − 06	0	68475	5301.5	1E−06	0	17433	546.04	1E − 06	0	**13130**	534.43	869.56	1241.9	1E + 05	0	12.858	0.2830	1E+05	0
5(30)	1710.5	582.76	3E + 05	338.16	1E−06	0	**1.15e + 05**	3960.1	1293.8	1499.7	3E + 05	0	12887	3070	3E + 05	0	22.356	0.5212	3E+05	0
5(50)	5617.8	1398.5	5E + 05	76.436	1E−06	0	**3.41e + 05**	29588	1889	473.88	5E + 05	0	23042	3062.5	5E + 05	0	31.806	0.5094	5E+05	0
6(10)	0.0097	0.0018	**15119**	7659.5	0.0007	0.0025	18950	744.52	1.9677	0.8871	1E + 05	0	346.91	1274	1E + 05	0	175.39	54.32	1E+05	0
6(30)	19.345	55.169	2.32e+05	61954	0.1327	0.7267	**1.20e + 05**	37870	22.126	1.286	3E + 05	0	1281.6	2540.4	3E + 05	0	126.4	67.495	3E+05	0
6(50)	39.68	62.143	5E + 05	0	0.1327	0.7267	**2.8902e + 05**	1830	44.053	1.987	5E + 05	0	2356.8	4657.4	5E + 05	0	135.12	47.409	5E+05	0
7(10)	0.004	0.005	**14208**	9505.2	1267	4.63e − 13	1E + 05	0	1267	4.63e−13	1E + 05	0	1267	4.63e−13	1E + 05	0	433.94	23.69	1E+05	0
7(30)	0.007	0.005	**16209**	8010	4696.3	2.781e − 12	3E + 05	0	4696.3	2.78e−12	3E + 05	0	4696.3	2.78e−12	3E + 05	0	551.63	137.44	3E+05	0
7(50)	0.007	0.005	**34683**	14293	6195.3	0	5E + 05	0	6195.3	0	5E + 05	0	6195.3	0	5E + 05	0	570.07	105.33	5E+05	0
8(10)	**19.99**	0	1E + 05	220.82	20.312	0.1127	1E + 05	0	20.329	0.0878	1E + 05	0	20.197	0.1229	1E + 05	0	393.28	23.459	1E+05	0
8(30)	**20.049**	0.0177	3E + 05	0	20.892	0.17460	3E + 05	0	20.931	0.0639	3E + 05	0	20.368	0.0943	3E + 05	0	577.37	212.99	3E+05	0
8(50)	**20.183**	0.0330	5E + 05	0	21.127	0.0302	5E + 05	0	21.123	0.0333	5E + 05	0	20.458	0.0851	5E + 05	0	490.93	130.51	5E+05	0
9(10)	0.1383	0.3358	62464	23540	1.9457	1.5105	88203	30601	0.001	0.0031	61257	2891.9	0.0033	0.0048	**18983**	6020.3	120.01	7.23e−14	26480	799.31
9(30)	10.832	3.707	3E + 05	160.83	39.564	6.4543	3E + 05	0	9.5447	2.4587	3E + 05	0	0.0043	0.0050	1.57e+05	39913	120.01	7.23e−14	**69423**	1540
9(50)	51.064	9.0906	5E + 05	232.71	62.344	7.3313	5E + 05	0	25.701	4.792	5E + 05	0	2.79	1.4605	4.98e+05	8873.1	120.01	7.23e−14	**1.13e + 05**	772
10(10)	4.565	2.404	99982	54.319	1.647	1.1767	**90947**	27625	6.994	2.453	1E + 05	0	31.174	12.988	1E + 05	0	120.01	7.23e−14	38517	1131.4
10(30)	24.232	6.2578	3E + 05	263.61	**9.391**	3.2817	3E + 05	0	57.809	15.385	3E + 05	0	122.94	32.782	3E + 05	0	120.01	7.23e−14	2E+05	11139
10(50)	65.226	15.256	5E + 05	256.58	**24.551**	7.5998	5E + 05	0	113.05	30.051	5E + 05	0	210.01	48.102	5E + 05	0	120.01	7.23e−14	4.93e+05	9271
11(10)	5.8842	2.2351	1E + 05	105.01	1.2989	1.3647	**30310**	10496	3.0363	1.821	1E + 05	0	7.5876	1.1929	1E + 05	0	64449	60668	1E+05	0
11(30)	26.826	4.7552	3E + 05	186.49	**9.0255**	3.0546	3E + 05	0	18.799	4.7233	3E + 05	0	30.659	3.4532	3E + 05	0	6.68e+05	3.13e+05	3E+05	0
11(50)	50.423	6.1459	5E + 05	269.63	**19.875**	5.2923	5E + 05	0	44.413	8.0193	5E + 05	0	54.808	6.0167	5E + 05	0	1.16e+06	5.83e+05	5E+05	0
12(10)	0.01	0	**1435**	1022.4	1546.1	2735.5	70053	43090	31.331	57.576	98553	5590.6	873.7	1566.8	1E + 05	0	9.99	0	43280	3701.2
12(30)	**33.987**	56.522	3E + 05	2050.3	20324	19261	3E + 05	0	19275	10637	3E + 05	0	15214	11587	3E + 05	0	2358.2	1403.9	3E+05	0
12(50)	**2004.9**	1672.9	5E + 05	0	65826	69500	5E + 05	0	39679	20890	5E + 05	0	96421	45347	5E + 05	0	28915	8503	5E+05	0
13(10)	0.3263	0.1145	1E + 05	200.95	0.897	0.2532	1E + 05	0	0.5053	0.1715	1E + 05	0	0.4397	0.1695	1E + 05	0	180.01	1.45e−13	**41123**	2053
13(30)	2.851	0.7778	3E + 05	0	3.179	0.5606	3E + 05	0	2.2653	0.6094	3E + 05	0	**1.657**	0.4726	3E + 05	0	4499.3	974.88	3E+05	0
13(50)	11.667	3.092	5E + 05	0	5.3587	0.8337	5E + 05	0	**4.4253**	0.6463	5E + 05	0	4.7423	0.9329	5E + 05	0	10650	2614.5	5E+05	0
14(10)	3.3867	0.3873	1E + 05	259.34	**2.5847**	0.5302	1E + 05	0	2.8747	0.4770	1E + 05	0	3.6913	0.3142	1E + 05	0	696.49	21.172	99730	1478.9
14(30)	12.646	0.5535	3E + 05	0	**10.394**	0.8103	3E + 05	0	12.75	0.3902	3E + 05	0	13.092	0.3185	3E + 05	0	715.66	46.216	3E+05	0
14(50)	22.246	0.6296	5E + 05	0	**19.45**	1.0963	5E + 05	0	22.656	0.3133	5E + 05	0	22.696	0.3419	5E + 05	0	730.94	25.802	5E+05	0
15(10)	172.49	162.07	1E + 05	135.15	343.32	97.143	1E + 05	0	259.07	183.34	1E + 05	0	230.47	215.32	**66487**	39189	967.01	0.0610	1E+05	0
15(30)	358.48	123.71	3E + 05	254.94	266.27	64.911	3E + 05	0	233.32	75.81	3E + 05	0	330.52	176.73	**2.82e + 05**	54544	4398.1	10.078	3E+05	0
15(50)	308	91.376	5E + 05	289.18	245	63.66	5E + 05	0	**199.99**	1.45e−13	5E + 05	0	263.56	76.502	5E + 05	0	5895.3	9.25e−13	5E+05	0
16(10)	122.69	14.34	1E + 05	100.8	**96.626**	9.3047	1E + 05	0	103.94	6.0136	1E + 05	0	162.3	23.794	1E + 05	0	239.76	0.0975	1E+05	0
16(30)	232.86	161.57	3E + 05	262.84	130.49	150.98	3E + 05	0	**108.08**	70.925	3E + 05	0	210.75	100.3	3E + 05	0	239.2	0.1151	3E+05	0
16(50)	117.07	61.452	5E + 05	296.84	**98.632**	126.2	5E + 05	0	100.86	49.207	5E + 05	0	191.18	84.78	5E + 05	0	239.16	0.1752	5E+05	0
17(10)	135.86	29.211	1E + 05	78.65	99.584	10.511	1E + 05	0	116.31	10.965	1E + 05	0	160.47	28.047	1E + 05	0	447.56	1.4475	**96700**	12585
17(30)	236.14	184.39	3E + 05	169.31	175.66	179.47	3E + 05	0	**144.86**	49.841	3E + 05	0	256.62	162.06	3E + 05	0	419.07	11.331	3E+05	0
17(50)	356.2	89.104	5E + 05	200.17	182.28	166.26	5E + 05	0	**171.19**	43.478	5E + 05	0	237.14	111.19	5E + 05	0	366.43	18.114	5E+05	0