VOL. 14, NO. 17, SEPTEMBER 2019 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

# EMPIRICAL STUDY OF HYPER-HEURISTIC ON SEVERELY LIMITED NOISY OPTIMIZATION FUNCTION

Jia Hui Ong and Jason Teo
Evolutionary Computing Laboratory, Faculty of Computing and Informatics, Universiti Malaysia Sabah, Malaysia
E-Mail: ongkjh@gmail.com

**ABSTRACT**

Hyper-heuristics solve optimization problems by utilizing a collection of Low-level Heuristic (LLH) to search for optimal solution. The main aim of Hyper-heuristics is to be a generalize algorithms that can be used in any problem domain. Numerous studies which implement Hyper-heuristics as the search algorithm mostly focus in discrete optimization. Not much work was done using Hyper-heuristics in continuous problem domain, especially in expensive real-world problems. Expensive real-world optimization problems refer to optimizations that only permit a short number of evaluations due to the high expenses in resources. Research in expensive optimization problems often focus on tailoring an algorithm to perform well in a specific problem instances. Once the problem instants changes, the algorithm need to be tweak and tailored again to perform at an optimum level, hence more time, budget and expertise needed in order to do so. The main aim of using Hyper-heuristics is to be able to apply a general yet efficient optimization algorithm to all expensive problem instances with very minor or minimal tweaks. This paper focuses on using Hyper-heuristics in severely limited evaluation noisy optimizations problems that mimic the real-world expensive optimization problems. Hyper-heuristics are introduced and compared against few algorithms that are often used in optimization problems, it posted a good average ranking when it is compared against Differential Algorithm (DE), Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Covariance Matrix Adaptation (CMA) and Mean Variance Mapping Optimization (MVMO) in eight different noisy optimization problems.

**Keywords:** hyper-heuristics, expensive optimization, hyper-heuristic component selection, MVMO, noisy optimization.

## 1. INTRODUCTION

Hyper-heuristics is defined as a high-level approach that can be applied to any problem instantly at any decision point by using its collection of Low-level Heuristics. Cowling (Cowling *et al*, 2000) first introduced Hyper-heuristics in 2000 and he further extends the study of hyper-heuristics in scheduling problems (Cowling and Chakhlevith, 2003). Burke reviewed the methods and structures of hyper-heuristics usage (Burke *et al*, 2013) and in his paper he mentioned one of the main challenges in this method are, how to develop the framework to be more generally applicable to all search methodologies. There are only a few hyper-heuristics usages in continuous problem domain (Kiraz *et al*, 2013; Topcuoglu *et al,* 2014; Maashi *et al,* 2014). McClymont and Keedwell (2011) attempted to use hyper-heuristics in a reduce number of evaluations multi-objectives problems. Ong and Teo (2016) introduced the use of Hyper-heuristic to solve expensive continuous optimization problems. Expensive optimization problems mimic the real world optimization that requires a large amount of resources to run. Hence it is important to be able to search for an optimum solution in a short span of evaluation. A competition on expensive optimization problem in Congress on Evolutionary Computation (CEC) 2015 was held to gathered interest in this research area. Mean Variance Mapping Optimization (MVMO) emerge as the best performing algorithm in the competition and again in CEC 2016. Ong and Teo (2016) compared the performance of Hyper-heuristic against MVMO in the CEC 2015 test suits and the performance of Hyper-heuristics is very encouraging. Hence in this paper, Hyper-heuristics will be tested in even more vigorous environment which includes severely limited evaluation and noisy optimization problems. The performance of Hyper-heuristics is compare against DE, GA, PSO, CMA, and MVMO. These algorithms are often tuned and tailored to work well in a specific problem instances. Tuning and tailoring an algorithm is time consuming and requires expertise on the algorithm. Since the tailored algorithm was created to perform well towards a specific algorithm, when the problem instances change, more time and expertise are needed to tune the algorithm hence the reusability of tailored algorithm is relatively low.

The next section of this paper will go through the basic framework of a hyper-heuristics and explanation of how the component of the hyper-heuristics works. Third section of this paper will touch on the benchmark problems used for this study. Before moving into the experiment setup and experiment results, the framework of Hyper-heuristics used in this study will be discuss. The future work of hyper-heuristics in this research area will be discussed in the last section of this paper.

## 2. HYPER-HEURISTICS

### A. Structure of Hyper-heuristics

Hyper-heuristics can be categories into hyper-heuristics selection and hyper-heuristics generative (Burke *et al,* 2010) as shown in Figure-1. Selection hyper-heuristics will choose and applies low level heuristics to continue the search (Kheiri, 2014) while generative hyper-heuristics refer to the hyper-heuristics methodologies for generating new heuristics from the component of existing ones (Burke *et al*, 2010).

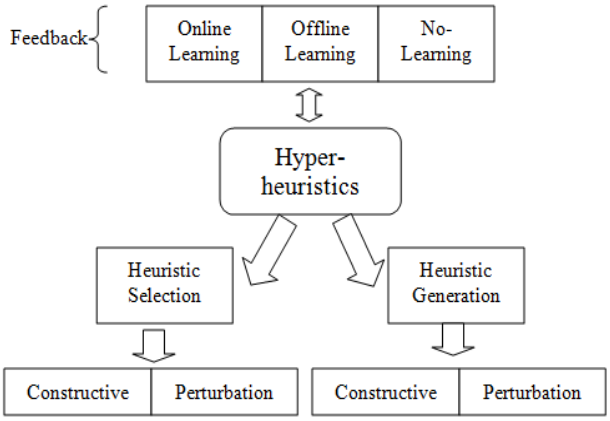


**Figure-1.** Classification of Hyper-heuristicss.

In the second level of the hyper-heuristics framework is the perturbation and constructive heuristics (Hoos and Stützle, 2004). In perturbation heuristics, the searching processes involve complete solutions while constructive heuristic will only processes partial solution. Feedback mechanism behind hyper-heuristics framework is to support learning algorithmand there are three categories in this feedback mechanism; online learning, offline learning and no learning. Online learning is a feedback that learns during the process while offline learning is a feedback that learns before the start of the process.

**B. Selection Hyper-heuristics**

Selection hyper-heuristics are considered as a high level problem solving framework. It has two major mechanisms namely heuristic selection and move acceptance as shown in Figure-2. There are various heuristic selection mechanisms and Cowling presented a few simple heuristic selections in his work (Cowling *et al*, 2000). They studied the performance of Simple Random (SR), Random Descent (Gradient) (RD (G)), Random Permutation (RP), Random Permutation Descent (Gradient) (RPD (G)), and Choice Function (CF).

- Simple random selection: - randomly choose LLH based on a uniform probability.
- Random descent (gradient): - randomly selects LLH and applies it until solution stop improving next selection will be randomly selected again.
- Random permutation randomly: - generates LLH into a list of all low level heuristics and applies them one after another at each step sequentially.
- Random permutation descent (gradient): - use the same concept as random permutation but random descent (gradient) approach without changing the order of heuristics.
- Greedy uses all low-level heuristics to generate potential solution and selects a heuristic that has the biggest improvement.
- Choice Function uses a mechanism that grades the low-level heuristic based on the improvement or worsen solution and also the duration of the last use low-level heuristic.

In Cowling's work, Tabu-searched was introduced as a selection mechanism (Cowling and Chakhlevith, 2003). Tabu-search will ranked the low-level heuristic to determine the next selection while keeping a list of disallow low-level heuristic to avoid using bad performing low-level heuristic. Tabu-search was first introduced by Glover (1986) in an integer programming.

$S_{curr}$ = candidate solution
$S_{best}$ = best solution
$LLH = \{LLH_1, LLH_2, LLH_3 \ldots LLH_n\}$
$S_{curr} \leftarrow S_{initial}$
$S_{best} \leftarrow S_{initial}$
***WHILE*** Termination_Criteria equals **False** ***Do***
    $LLH_i \leftarrow$ Select_llh(LLH)
    $S_{new} \leftarrow$ Apply_heuristic($LLH_i$, $S_{curr}$ )
    **IF** Acceptance($S_{curr}$ , $S_{new}$) **Then**
        $S_{curr} \leftarrow S_{new}$
    **End**
$S_{best} \leftarrow$ Update_Best_Solution($S_{curr}$)
**End WHILE**
**return** $S_{best}$

**Figure-2.** Selection Hyper-heuristics Pseudocode.

Second mechanism of Selection Hyper-heuristics is move acceptance. This mechanism is the decision maker to accept or not for the newly generated solution. The basic move acceptances that were suggested by Cowling (Cowling *et al*, 2000) are:

- All Moves (AM): - accept all the generated solution
- Only Improving (OI): - if current solution improved from previous best and only accept all the improved solution
- Improving or Equal (IE): - accepts non worsening solutions from the previous best solution

It can be further distinguished into deterministic and non-deterministic. Deterministic refers to move acceptance that return the same decision at every iteration, while non-deterministic refer to move acceptance criteria that depends on the current iteration. It can then be categorized into stochastic or non-stochastic category. These categories exist when probabilistic framework is considered while making acceptance decision.

Multi-point search refers to the search that maintains a pool of population solution while single-point search refers to search that improves and maintains a single solution. Most of the researches done are based on single point perturbation approach and only a few studies use the multi-point perturbation approach (Burke *et al,* 2013).

## 3. TEST SUITS AND HYPER-HEURISTICS SETUP

Eight noisy functions are used in this study and all functions are minimization problems (Das, 2005) (Bhattacharya, 2014).

Sphere (5D):

$$f_{sphere}(x) = \sum_{i=1}^{n} x_i^2 \quad \text{with } -100 \leq x_i \leq 100$$

Rosenbrock (50D):

$$f_{rosenbrock}(x) = \sum_{i=1}^{n} [\,100\ (x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$$

with $-50 \leq x_i \leq 50$

Rastrigin (50D):

$$f_{rastrigin}(x) = \sum_{i=1}^{n} [\,x_i^2 - 10\cos(2\pi x_i) + 10$$

with $-5.12 \leq x_i \leq 5.12$

Griewank (50D):

$$f_{griewank}(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$$

with $-600 \leq x_i \leq 600$

Beale (2D):

$$f_{beale}(x) = [1.5 - x_1(1 - x_2)]^2 + [2.25 - x_1(1 - x_2^2)]^2 + [2.625 - x_1(1 - x_2^3)]^2$$

with $-10 \leq x_i \leq 10$

Ackley (50D)

$$f_{ackley} = -20\exp\left(-0.2\sqrt{\left(\frac{1}{50}\sum_{i=1}^{n} x_i^2\right)}\right) - \exp\left(\frac{1}{50}\sum_{i=1}^{n}\cos 2\pi x_i\right) + 20 + e$$

with $-32 \leq x_i \leq 32$

Schaffer (2D):

$$f_{schaffer}(x) = 0.5 - \frac{(\sin\sqrt{x_1^2 + x_2^2})^2 - 0.5}{(1.0 + 0.001(x_1^2 + x_2^2))^2}$$

with $-100 \leq x \leq 100$

Schwefel (50D):

$$f_{schwefel} = 418.9829(50) - \sum_{i=1}^{n} x_i \sin(\sqrt{|x_i|})$$

with $-100 \leq x_i \leq 100$

In this paper, Hyper-heuristic selection method used are Tabu-Search Biased Ranking (TSBR) and move acceptance used are only improving (OI)

There are five low-level heuristics that will be used by BROI:

- Covariance Matrix Adaptation (CMA), $\lambda$ = 4+ 3 log ($N$), $\mu = \lambda/2$.
- Particle Swarm Optimization (PSO), initial velocity 1 and maximum velocity 3.
- Differential evolution (DE), $Cr$ = .9, F = .2,
- Genetic Algorithms (GA), $Cr$ = .5, *Mutation Rate* = .1. Tournament selection, tournament size = 2
- Mean Variance Mapping Optimization, archive size = 3

These LLH were chosen as a popular algorithm used in many research fields. To utilize the basic MVMO (Longatt *et al*, 2012) an archive was used to store three best solutions found. This archive is used to calculate the mean and shape variable to generate new solution as describe by Longatt. A basic version of MVMO was used instead of the extended version. Island based uses a single heuristic chosen by heuristic selection and controls a population in a single iteration while single based will allow different heuristics to generate single solution that forms a population in a single iteration. It was found that island based setup yields better results in a multi-point search hence island base setup will be used in this study for all algorithms.

In this study, island based will be separate into two sections. The first section will be based on Tabu-search selection and score will be assign to the LLH based on whether the solution generate improved or not. The second section will be fully controlled by the highest rank LLH to continue the search. The section division is based on the number of evaluation allowed and divides it into two. The overall view of the Hyper-heuristic framework used in this study are illustrated in Figure-3.

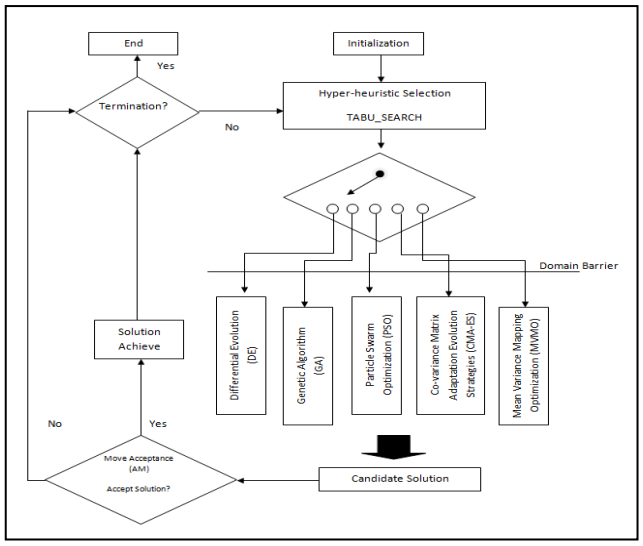


**Figure-3.** Selection Hyper-heuristics Overview.

## 4. EXPERIMENTAL SETUP

The proposed algorithm will be tested using the eight function mention in previous section. Population size of 5 was used for the algorithm as it was found to be the ideal population size to be used in order to get a good result from island base setup. Each function will be tested for 51 times. Evaluations of all experiments were carried out on the same PC. The algorithm will be tested using three different number of evaluations 100, 250 and 500.

## 5. RESULTS AND DISCUSSIONS

**Table-1.** Noisy sphere

| No. Evaluations | 100 | 250 | 500 |
|---|---|---|---|
| DE | 6.44E+02 | 7.23E+02 | 5.34E+02 |
| GA | 1.34E+03 | 3.05E+02 | 2.00E+02 |
| PSO | 1.07E+00 | 7.11E-01 | 5.72E-01 |
| CMAES | 9.02E+03 | 5.66E+02 | 3.90E+00 |
| MVMO | 1.17E+03 | 5.33E+02 | 3.54E+02 |
| HH-TSBR | 8.55E-01 | 6.89E-01 | 6.08E-01 |

**Table-2.** Noisy rosenbrock

| No. Evaluations | 100 | 250 | 500 |
|---|---|---|---|
| DE | 5.92E+09 | 5.72E+09 | 5.98E+09 |
| GA | 4.87E+09 | 4.59E+09 | 4.25E+09 |
| PSO | 1.78E+03 | 1.01E+03 | 7.55E+02 |
| CMAES | 6.19E+09 | 6.18E+09 | 6.18E+09 |
| MVMO | 5.42E+08 | 1.92E+08 | 8.41E+07 |
| HH-TSBR | 1.82E+03 | 1.69E+03 | 1.41E+03 |

**Table-3.** Noisy Rastrigin

| No. Evaluations | 100 | 250 | 500 |
|---|---|---|---|
| DE | 8.90E+02 | 8.65E+02 | 8.39E+02 |
| GA | 8.52E+02 | 8.28E+02 | 8.10E+02 |
| PSO | 4.94E+02 | 4.66E+02 | 4.51E+02 |
| CMAES | 8.45E+02 | 8.41E+02 | 8.34E+02 |
| MVMO | 4.82E+02 | 3.79E+02 | 3.13E+02 |
| HH-TSBR | 4.22E+02 | 4.10E+02 | 3.90E+02 |

**Table-4.** Noisy griewank

| No. Evaluations | 100 | 250 | 500 |
|---|---|---|---|
| DE | 7.10E+00 | 6.22E+00 | 6.00E+00 |
| GA | 1.29E+03 | 1.17E+03 | 1.11E+03 |
| PSO | 1.20E+00 | 1.20E+00 | 1.20E+00 |
| CMAES | 1.46E+03 | 1.46E+03 | 1.46E+03 |
| MVMO | 3.84E+02 | 2.15E+02 | 1.47E+02 |
| HH-TSBR | 1.41E+00 | 1.41E+00 | 1.41E+00 |

**Table-5.** Noisy Beale

| No. Evaluations | 100 | 250 | 500 |
|---|---|---|---|
| DE | 5.07E+02 | 5.07E+02 | 5.06E+02 |
| GA | 4.25E+04 | 1.13E+04 | 8.51E+02 |
| PSO | 5.08E+02 | 5.07E+02 | 5.06E+02 |
| CMAES | 8.49E+05 | 7.42E+05 | 6.48E+05 |
| MVMO | 5.04E+02 | 5.02E+02 | 5.01E+02 |
| HH-TSBR | 5.05E+02 | 5.05E+02 | 5.04E+02 |

**Table-6.** Noisy Ackley

| No. Evaluations | 100 | 250 | 500 |
|---|---|---|---|
| DE | 3.40E+00 | 3.32E+00 | 3.27E+00 |
| GA | 2.21E+01 | 2.21E+01 | 2.20E+01 |
| PSO | 4.04E+00 | 3.84E+00 | 3.65E+00 |
| CMAES | 2.28E+01 | 2.28E+01 | 2.28E+01 |
| MVMO | 1.99E+01 | 1.81E+01 | 1.66E+01 |
| HH-TSBR | 3.61E+00 | 3.58E+00 | 3.56E+00 |

**Table-7.** Noisy Schaffer.

| No. Evaluations | 100 | 250 | 500 |
|---|---|---|---|
| DE | 3.88E-01 | 3.28E-01 | 3.14E-01 |
| GA | 4.99E-01 | 4.70E-01 | 4.50E-01 |
| PSO | 5.34E-01 | 4.96E-01 | 4.39E-01 |
| CMAES | 1.18E+00 | 1.18E+00 | 1.17E+00 |
| MVMO | 9.19E-01 | 8.56E-01 | 7.90E-01 |
| HH-TSBR | 4.72E-01 | 4.42E-01 | 4.16E-01 |

**Table-8.** Noisy Schwefel.

| No. Evaluations | 100 | 250 | 500 |
|---|---|---|---|
| DE | 2.02E+04 | 2.03E+04 | 2.02E+04 |
| GA | 1.94E+04 | 1.92E+04 | 1.85E+04 |
| PSO | 2.04E+04 | 2.04E+04 | 2.04E+04 |
| CMAES | 2.09E+04 | 2.09E+04 | 2.09E+04 |
| MVMO | 1.70E+04 | 1.63E+04 | 1.57E+04 |
| HH-TSBR | 2.02E+04 | 2.02E+04 | 2.02E+04 |

**Table-9.** Average Ranking for algorithms comparison.

| No. Evaluations | 100 | 250 | 500 |
|---|---|---|---|
| DE | 3.250 | 3.625 | 3.625 |
| GA | 4.250 | 3.875 | 4.125 |
| PSO | 2.875 | 2.875 | 2.625 |
| CMAES | 5.750 | 5.750 | 5.500 |
| MVMO | 3.000 | 2.875 | 3.000 |
| HH-TSBR | 1.875 | 2.000 | 2.125 |

In Table-1, the results obtain for Noisy Sphere function are shown. HH-TSBR manage to obtain the best results in 100 evaluations and 250 evaluations with $8.55E^{-01}$ and $6.89E^{-01}$ respectively. While in evaluations 500, HH-TSBR $6.08E^{-01}$ came in second behind PSO's $5.72E^{-01}$. In Table-2 results for Noisy Rosenbrock, PSO is the overall best in all three evaluations $1.78E^{+03}$, $1.01E^{+03}$, $7.55E^{+02}$ but HH-TSBR is not too far behind with $1.82E^{+03}$, $1.69E^{+03}$ and $1.41E^{+03}$. Noisy Rastrigin results are shown in Table-3, PSO able to post the best results in all three evaluations $4.22E^{+02}$, $4.10E^{+02}$ and $3.90E^{+02}$. In Noisy Griewank results shown in Table-4, HH-TSBR obtained $1.41E^{+00}$ for 100 evaluations, $1.41E^{+00}$ for 200 evaluations and $1.41E^{+00}$ 500 evaluations. HH-TSBR came in second behind PSO in this function.

Table-5 shows the results obtained for Noisy Beale functions, in this functions HH-TSBR obtained second best results for all three evaluations. While MVMO posted the best results in all three evaluations. The results for Noisy Ackley functions are shown in Table-6. HH-TBSR again obtained the second best results for all three evaluations while DE managed to obtain the best results in all three evaluations. Table-7 shows the results for Noisy Schaffer, HH-TSBR obtained $4.72E^{-01}$ for evaluations 100, $4.42E^{-01}$ for evaluations 250 and $4.16E^{-01}$ for evaluations 500. HH-TSBR posted the second best overall results in all three evaluations while DE again becomes the best overall in these functions. DE obtained $3.88E^{-01}$ for evaluations 100, $3.28E^{-01}$ for evaluations 250 and $3.14E^{-01}$ for evaluations 500. In the last functions results Noisy Schwefel shown in Table-8. HH-TSBR managed to obtained second rank in two evaluations which is 100 and 250 evaluations.

Although in some functions HH-TSBR did not managed to post the best results, it still managed to gives good results that are quite close to overall best result. As shown in Table-9, HH-TSBR posted an average ranking of 1.875 in the evaluations 100 which is the best average ranking and 2.000 in evolutions 250 and 2.125 in evaluations 500. In the average ranking we can see that HH-TSBR can still perform well in any given functions and the single algorithms that were compare against HH-TSBR can only perform well in certain functions.

## 6. CONCLUSIONS

From the results obtained, it shows that hyper-heuristics is capable to post good results as a generalized algorithm against tailored algorithms even in extreme condition where only a short number of evaluations are allow with added noisy into functions. If a generalized algorithm is able to post good solutions it will reduce the time and expertise needed to tailored an algorithm for different problem instances. It is shown from the results where none one the tested algorithms can dominate across all functions as stated in No Free Lunch Theorem. Hyper-heuristics leverage on the usage of different LLH to solve different type of problems and this might help in addressing No Free Lunch Theorem.

More attention and work on Hyper-heuristics in continuous problem domain should be done in order to gain from the convenient of using Hyper-Heuristics especially in real-world optimization problems.

## REFERENCES

E. K. Burke, M. Hyde, G. Kendall, G. Ochoa, E. Özcan, and J. R. Woodward. 2010. A classification of hyper-heuristicss approaches. In Handbook of Metaheuristic. 146: 449-468.

E.K. Burke, M. Gendreau, M. Hyde, G. Kendall, G. Ochoa, E. Özcan, and R. Qu. 2013. Hyper-heuristicss: a survey of the state of art. Journal of the Operational Research Society. 64(12): 1695-1724.

P. Cowling, G. Kendall and E. Soubeiga. 2001. A hyperheuristic approach to scheduling a sales summit. in Practice and Theory of Automaed Timetabling III. 2079: 176-190.

P. Cowling and K. Chakhlevith. 2003. Hyperheuristic for managing a large collection of low level heuristic to schedule personnel, in IEEE Congres of Evolutionary Computation (CEC' 03). pp. 1214-1221.

F. Glover. 1986. Future paths for integer programming and links to artificial intelligence. Computers and Operations Research. 13(5): 533-549.

H. H. Hoos and T. Stützle. 2004. Stochastic local search: foundation and applications. Elsevier/Morga Koufmann: San Francisco, CA.

S. H. Jun. 2006. A Hybrid Genetic Algorithm and New Criterion for Determining the Number of Clusters. International Journal of Soft Computing. 1: 313-318.

A. Kheiri. 2014. Multi-stage hyper-heuristicss for optimization problems. Phd Thesis.

B. Kiraz, A. S. Uyar and E. Özcan. 2013. Selection hyper-heuristicss in dynamic environments.Journal of the Operational Research Society. 64(12): 1753‑1769.

F. G. Longatt, J. Rueda., I, Erlich, W, Villa. and D, Bogdano. 2012. Mean Variance Mapping Optimization for the identification of Gaussian Mixture Model: Test case. In Proceedings IS'2012 - 2012 6th IEEE International Conference Intelligent Systems.

M. Maashi, E. Ozcan, and G. Kendall. 2014. A multi-objective hyperheuristic based on choice function. Expert Systems with Applications. 41(9): 4475‑4493.

K. McClymont and E. C. Keedwell. 2011. Markov chain hyper-heuristicss (MCHH): an online selective hyper-heuristics for multi-objective continuous problems. In Proceedings of the 13th Annual Conference on Genetic and EvolutionaryComputation (GECCO '11), pp. 2003‑2010, New York, NY, USA, ACM.

M. Nasereddin. 2006. Using Genetic Algorithms to Find Weights for Multiple Heuristic for the Stochastic Resource Constrained Project Scheduling Problem. International Journal of Soft Computing. 1: 255-260.

I. Thamarai and S. Murugavalli. 2016. Analogy Based Software Effort Estimation Based on Differential Evolution and Hybrid Fuzzy Logic and Firefly Algorithm. Asian Journal of Information Technology. 15: 1484-1493.

H. R. Topcuoglu, A. Ucar, and L. Altin. 2014. A hyper-heuristicss based framework for dynamic optimization problems. Applied Soft Computing. 19: 236‑251.