Enhancing generality of meta-heuristic algorithms through adaptive selection and hybridization

Kamal Z. Zamli
Faculty of Computer Systems & Software Engineering
Universiti Malaysia Pahang
Kuantan, Pahang, Malaysia
kamalz@ump.edu.my

***Abstract*— Solving complex optimization problems can be painstakingly difficult endeavor considering multiple and conflicting design goals. A growing trend in utilizing meta-heuristic algorithms to solve these problems has been observed as they have shown considerable success in dealing with tradeoffs between conflicting design goals. Many meta-heuristic algorithms have been developed to date (e.g. Simulated Annealing (SA), Particle Swarm Optimization (PSO), Teaching Learning based Optimization (TLBO), Grey Wolf Optimizer(GWO) to name a few). Much of these algorithms have adopted elegant metaphors (e.g. heating and cooling of metals in the case of SA and swarming of flocking birds in the case of PSO) from nature in order to derive the mathematical models for generating the solution as well as provides control over their exploration (i.e. sufficient roaming of the search space) and exploitation (i.e. using known knowledge of the surroundings). In line with the no free lunch theorem (), this paper argues that rather than focusing on designing new algorithm, new research should focus on adaptive hybridization of meta-heuristics algorithms in order to compensate the limitation of one with the strengths of another. In this paper, we review the meta-heuristic and hyper-heuristic algorithms in order to highlight the current-state-of-the-arts and suggest areas for future research.**

Keywords—meta-heuristics; hyper-heuristics

I. Introduction

Solving complex optimization problems can be painstakingly difficult endeavor considering multiple and conflicting design goals. A growing trend in utilizing meta-heuristic algorithms to solve these problems has been observed as they have shown considerable success in dealing with tradeoffs between conflicting design goals. Many meta-heuristic algorithms have been developed in the past 25 years. In order to control the exploration (sufficient roaming of the search space) and exploitation (using known knowledge of the surroundings), some meta-heuristic algorithms require parameter controls that are required to balance between exploitation and exploration. For example, Genetic Algorithm [2] adopts three parameter controls consisting of population size, mutation and cross over rate. In the same manner, Particle Swarm Optimization [3] adopts the population size, inertia weight, social and cognitive parameters as parameters.

To avoid difficulties in tuning the control parameters, many researchers have advocated the adoption of *parameter free* meta-heuristic algorithms (e.g. Teaching Learning based Optimization [4], Symbiotic Organism Search [5], Jaya Algorithm [6] and Sine Cosine Algorithm [7]). While avoiding any parameter controls can alleviate the tuning issues, *parameter free* meta-heuristic algorithms often suffer from preset pattern for exploration and exploitation. For example, the Teaching Learning based Optimization [4] will always perform global search (i.e. termed teacher phase) and local search (i.e. termed student phase) in sequence. Such preset pattern could be counter-productive given potentially different search spaces for different problems.

Bounded by the no free lunch theorem (i.e. no single algorithm can be best for all optimization problems [8-10]), this paper reviews the meta-heuristic and hyper-heuristic algorithms (in order to highlight the current-state-of-the-arts and suggest areas for future research). In particular, i.e. no single algorithm can be best for all optimization problems this paper argues for adaptive hybridization of meta-heuristic algorithms and highlights their potential opportunities.

II. Meta-heuristic Versus Hyper-heuristic Algorithims

Meta-heuristic algorithms, first introduced in TS algorithm by Glover [11], are part of stochastic algorithms that efficiently explore the search space by some systematic trial and error process. Meta-heuristic algorithms can be single solution or population based. Single solution meta-heuristic algorithms manipulate a single vector via tweaking one or more of its neighborhood values. For this reason, single solution meta-heuristic algorithms can potentially be sensitive to its initial starting position. Population-based algorithms relies on multiple vectors, thus, a solution can start from many positions of solution space. Therefore, each member of the population is a candidate to be the best solution. Often, any meta-heuristic algorithms provide search guidance for movement in the search space in order to explore overall search space efficiently. The search guidance is in the form of (problem specific) fitness function (i.e. each solution obtains the defined scores based on its quality). Here, the fitness function can be maximisation or minimization of certain parameters based on problem at hand.

Apart from adopting single solution or population-based approach, all meta-heuristic algorithms have two core components: exploitation (local search) and exploration (global search). Exploitation explores the promising regions within the close neighbor in the hope to find better solutions. On the other hand, exploration ensures that all regions of the search space have been visited, which enables the algorithm to get out of any local optima.

Since each meta-heuristic algorithm has its own advantages, meta-heuristic hybridization can be helpful to compensate for the limitation of one algorithm with the strengths of another.[12]. Apart from hybridization, developing generic general purpose meta-heuristic algorithms can also be a futile effort [13-15]. As the name suggests, general purpose meta-heuristic algorithms can avoid problem dependent solution and enhance the algorithms' applicability across other optimization problems.

Given the two aforementioned considerations, variants of meta-heuristic algorithms (called hyper-heuristic algorithms) have been suggested in the literature. Considering the different possible options for hybridization of meta-heuristic algorithms (see Fig. 1), there are three possible options. The first option is the standard meta-heuristic algorithm. Apart from the first option, the hybrid meta-heuristic algorithms can be in two

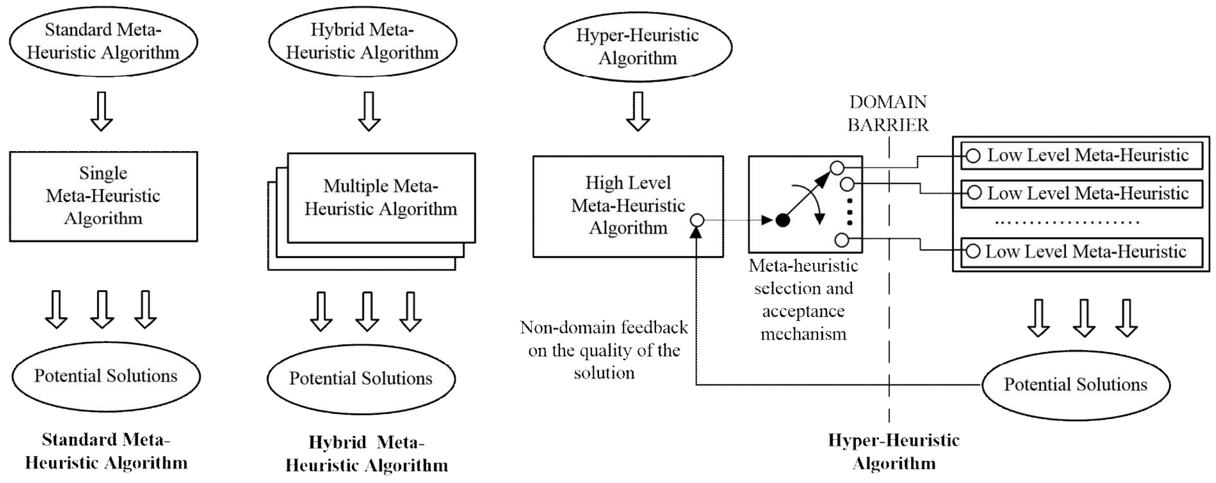
Standard Meta-Heuristic Algorithm
Hybrid Meta-Heuristic Algorithm
Hyper-Heuristic Algorithm
Single Meta-Heuristic Algorithm
Multiple Meta-Heuristic Algorithm
High Level Meta-Heuristic Algorithm
DOMAIN BARRIER
Low Level Meta-Heuristic
Low Level Meta-Heuristic
Low Level Meta-Heuristic
Meta-heuristic selection and acceptance mechanism
Non-domain feedback on the quality of the solution
Potential Solutions
Potential Solutions
Potential Solutions
Standard Meta-Heuristic Algorithm
Hybrid Meta-Heuristic Algorithm
Hyper-Heuristic Algorithm

Fig.1 Standard Meta-Heuristic, Hybrid Meta-Heuristic and Hyper-Heuristic adopted from [1]

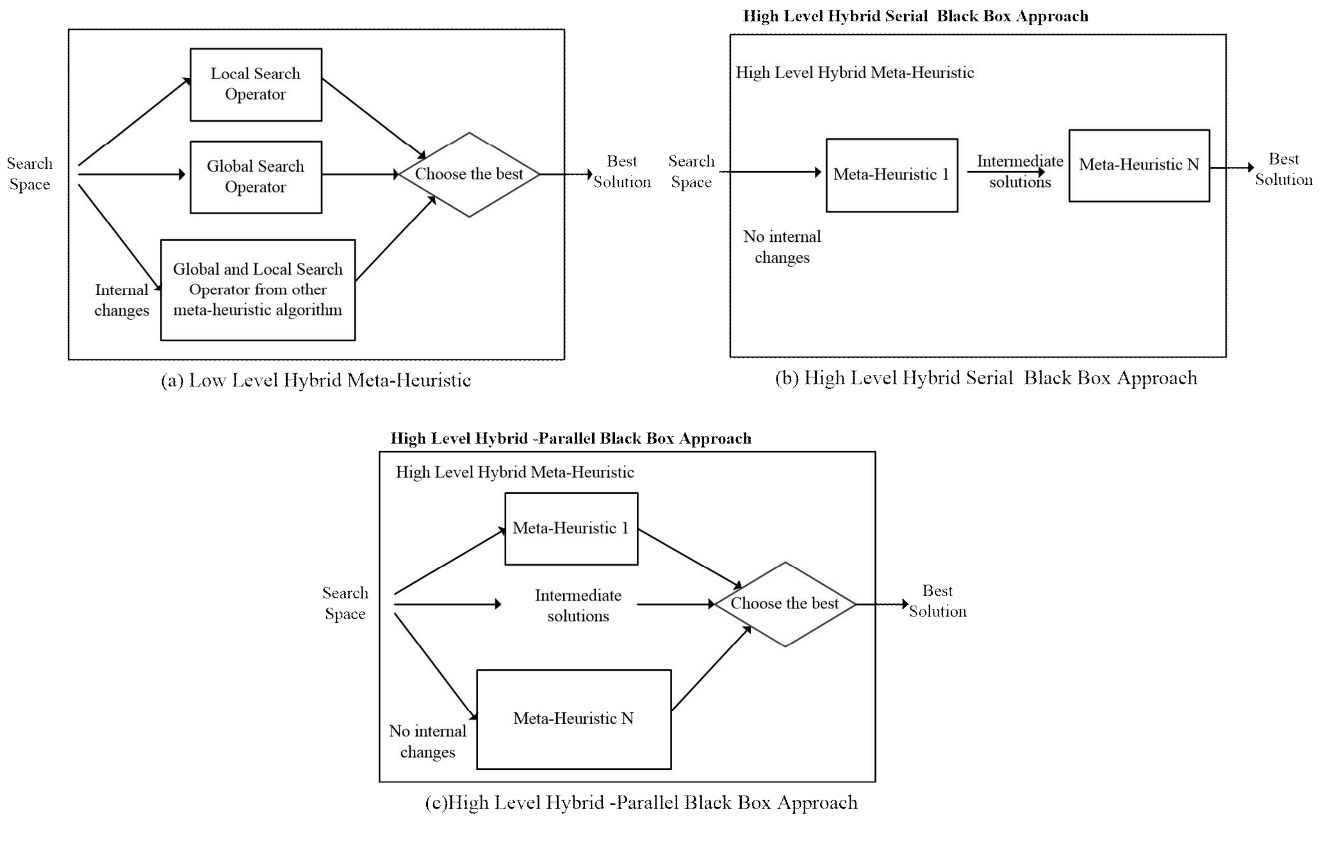
Search Space
Local Search Operator
Global Search Operator
Global and Local Search Operator from other meta-heuristic algorithm
Internal changes
Choose the best
Best Solution
(a) Low Level Hybrid Meta-Heuristic
High Level Hybrid Serial Black Box Approach
High Level Hybrid Meta-Heuristic
Search Space
Meta-Heuristic 1
Intermediate solutions
Meta-Heuristic N
Best Solution
No internal changes
(b) High Level Hybrid Serial Black Box Approach
High Level Hybrid -Parallel Black Box Approach
High Level Hybrid Meta-Heuristic
Search Space
Meta-Heuristic 1
Intermediate solutions
Choose the best
Best Solution
Meta-Heuristic N
No internal changes
(c)High Level Hybrid -Parallel Black Box Approach

Fig. 2 Low Level versus High Level Hybrid Meta-Heuristic

forms: low level and high level. Low level hybrid meta-heuristic algorithms involve combining one or more search operators from different meta-heuristic algorithms to work as a single algorithm. Referring to Fig. 2(a), low level hybrid meta-heuristic algorithms require changes in the source code (i.e. to allow code integration from one meta-heuristic algorithm to its host meta-heuristic algorithm). High level hybrid meta-heuristic algorithms combine two or more standard meta-heuristic algorithms as black box components running in sequence or in parallel. As shown in Fig. 2(b) and Fig. 2(c), no internal changes of the source code are required. In the case of Fig. 2(a) only the optimized population from one meta-heuristic algorithm is passed through the second one. Meanwhile, in the case of Fig. 2(c), only the best solution is selected from the parallel running of more than one meta-heuristic algorithm.

Owing to the integration of more than integration of more than one meta-heuristic algorithm (refer to Figure 1), hyper-

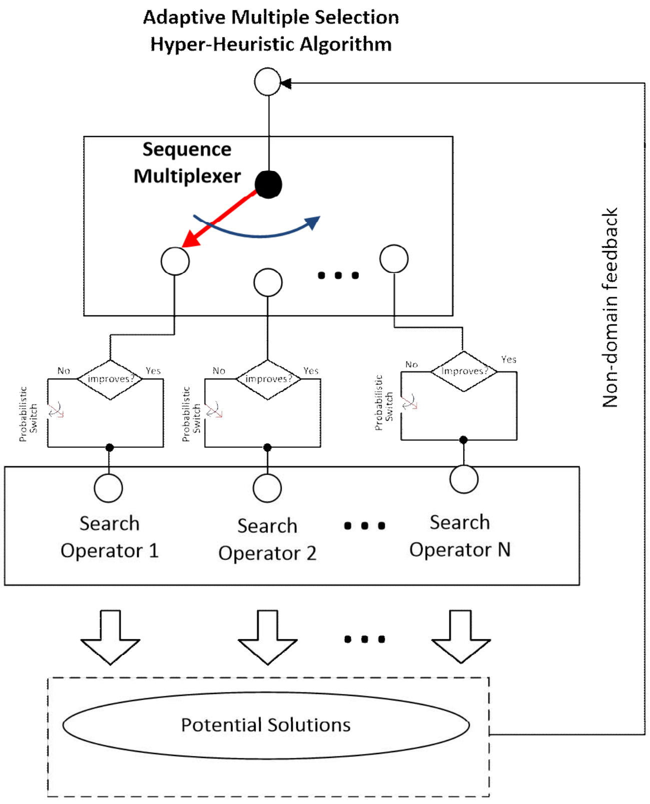

Fig.3 Adaptive Probabilistic Multiple Selection Hyper-Heuristic Algorithm

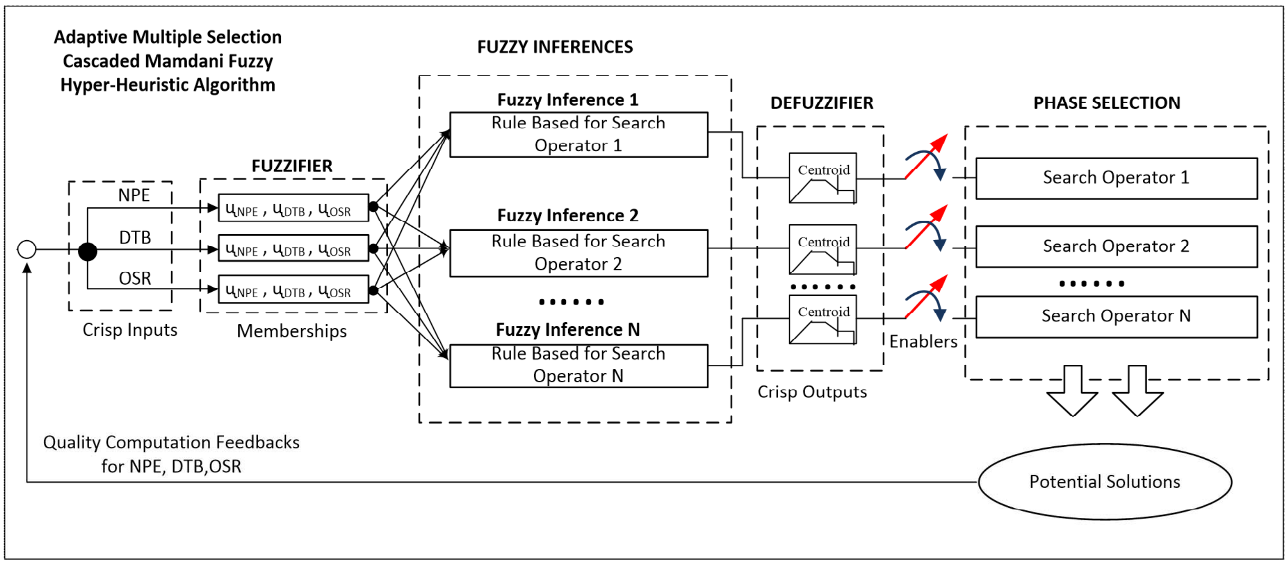

Fig. 4 Adaptive Multiple Selection Cascaded Mamdani Fuzzy Hyper-Heuristic Algorithm

heuristic algorithm could be seen as a hybrid meta-heuristic Nonetheless, unlike a typical hybrid meta-heuristic, hyper-heuristics (or *meta-meta heuristics* [13-15]) adopts a high level meta-heuristic (i.e. as a coordinator) to adaptively choose from a set of low level meta-heuristics (LLHs) through a domain barrier. It is the selection and acceptance mechanism that decides to choose the right low level meta-heuristics accordingly based on the problem at hand. Apart from the aforementioned arrangement, it is possible to construct generative hyper-heuristic algorithms which effectively learn about the best combination of low-level heuristics as a way to generate new combinations of high level heuristics.

III. Review of Selected Hyper-heuristic Algorithms and its Applications

This section reviews some of the relevant hyper-heuristic algorithms. Choice Function (CF) and Exponential Monte Carlo with Counter (EMCQ) [16] are among the earliest hyper-heuristics reported in the scientific literature. CF exploits the reinforcement learning framework to penalize and reward (meta)-heuristics through a set of choice functions (f_1, f_2, f_3). The first parameter f_1 relates to the effectiveness of the currently employed heuristic. The second parameter f_2 evaluates the effectiveness of two heuristics when used consecutively. The third parameter f_3 increases the probability of a heuristic being selected, over time, to encourage exploration. EMCQ adopts a simulated annealing like probability density function that is a function of the number of iterations. A worsening fitness causes EMCQ to decrease its acceptance probability. Recently, Jia et al. [17] adopts EMCQ-like mechanism to select from variants of six low-level search operators (i.e. single/multiple/smart mutation, simple/smart add and delete row).

Asmuni et al. [18] and Zamli et al. [19] exploit the Mamdani type fuzzy system as a way to adaptively select the search operators. Gudino-Penaloza et al. [20] adopts Takagi-Sugeno based fuzzy inference system to adaptively adjust the control parameters of GA as the high level hyper-heuristic algorithms. In other work, Zamli et al. [1] implemented improvement selection rules (ISR) utilizing selective hyper-heuristic based on tabu search and three measures (quality, diversify and intensify) to assist the heuristic selection process. Although showing promising results, the ISR selection rules are too strict and support only Boolean outcomes. Furthermore, ISR is also computationally heavy as its implementation is based on full meta-heuristic algorithms (i.e. comprising of and Cuckoo Search Algorithm (CS) [22]), Teaching Learning based Optimization (TLBO) [4], Particle Swarm Optimization (PSO)[3], and Global Neighborhood Algorithm (GNA) [21], as its search operators. To date, there are already many hyper-heuristic a*lgorithms* (e.g.) introduced in the literature and being adopted in many different areas of research including time tabling[15] vehicle routing problems [23], search based software engineering [1, 17] and network optimization[24].

IV. Where to go from here?

Revisiting Figure 1, meta-heuristic selection mechanism for hyper-heuristic algorithm always selects one LLH/search operators for every iteration. As the search process is dynamic, there is indeed a need to adaptively execute one or more combinations of search operators per iteration. In the early stage of the searching process, executing one or more combinations of search operators per iteration ensures diversity of solution and allows sufficient roaming of the search area. Towards the end of the searching process, execution of one or more combinations of search operators per iteration can facilitate convergence and avoid getting trapped in local optima. In fact, recent evidences in the literature [4, 5] suggest that executing more than one operator per iteration can facilitate convergence and achieve good performance. Thus, a potentially new direction of research is to develop adaptive multiple selection hyper-heuristic algorithms (see Fig. 3).Here, the adaptive switch can be varied according to the need of the search via some probabilistic function. In similar manner, adaptive multiple selection hyper-heuristic algorithms can be built from cascaded Mamdani fuzzy system (see Fig. 4). Here, one must define the right inputs (e.g. NPE, DTB, and OSR) and the fuzzy rules so as to be able to adaptively select one or more than one combinations of operators in the selection phase.

V. Closing Remarks

Summing up, this paper highlights the need for more research on hyper-heuristic algorithms. Specifically, rather than concentrating on new metaphors for developing meta-heuristic algorithms, this paper argue against reinventing the wheesl. For instance, mimicking another animal behavior or another natural phenomenon may give different mathematical equations. Yet, in the end, those mathematical equations are just another set of equations to perform local (i.e. small random displacement of the current value) and global search (i.e. large random displacement of the current value). We argue for enhancing existing meta-heuristic algorithms to form adaptive and hybrid hyper-heuristic algorithms. The benefit of adaptive and hybridization is twofold. Adaptive behavior allows the hyper-heuristic to make decision on which particular heuristic is suitable at any particular instance of the searching process. Hybridization, on the other hand, allows the hyper-heuristic to compensate the limitation of one meta-heuristic (or its search operator) algorithm with the strength of others.

As hyper-heuristic involves hybridization of more than one meta-heuristic algorithm, there could be potentially risk in terms of computational costs. Indeed, taking hybridization of meta-heuristic algorithm in total could result in bulky implementation and high computational costs. However, one can also develop a lightweight hyper-heuristic algorithm that adopts the individual search operator rather than the whole algorithm. In this manner, the implementation would not incur any additional overhead costs.

As the scope of future work, this paper also suggests potential new areas for research particularly on multiple selection hyper-heuristic algorithms capable of selecting one or more combination of low level meta-heuristic/search operators for execution.

Acknowledgment

The work undertaken in this research is funded by Fundamental Research Grant from Ministry of Higher Education Malaysia [Grant Number: RDU170103] titled: A Reinforcement Learning Sine Cosine based Strategy for Combinatorial Test Suite Generation.

References

[1] K. Z. Zamli, B. Y. Alkazemi, and G. Kendall, "A Tabu Search Hyper-Heuristic Strategy for t-way Test Suite Generation," *Applied Soft Computing,* vol. 44, pp. 57-74, 2016.

[2] J. H. Holland, *Adaptation in Natural and Artificial Systems* University of Michigan Press, 1975.

[3] J. Kennedy and R. Eberhart, "Particle Swarm Optimization," in *Proceedings of the IEEE International Conference Neural Networks*, 1995, pp. 1942-1948.

[4] R. V. Rao, V. J. Savsani, and D. P. Vakharia, "Teaching-Learning-based Optimization: A Novel Method for Constrained Mechanical Design Optimization Problems," *Computer Aided Design,* vol. 43, pp. 303-313, 2011.

[5] M.-Y. Cheng and D. Prayogo, "Symbiotic Organism Search: A New Meta-Heuristic Optimization Algorithm," *Computers and Structures,* pp. 98-112, 2014.

[6] R. V. Rao, "Jaya: A Simple and New Optimization Algorithm for Solving Constrained and Unconstrained Optimization Problems," *International Journal of Industrial Engineering Computations,* vol. 7, pp. 19-24, 2016.

[7] S. Mirjalili, "SCA: A Sine Cosine Algorithm for Solving Optimization Problems," *Knowledge Based Systems,* vol. 96, pp. 120-133, 2016.

[8] D. H. Wolpert and W. G. Macready, "No Free Lunch Theorems for Optimization," *IEEE Transactions on Evolutionary Computation,* vol. 1, pp. 67-82, 1997.

[9] A. R. A. Alsewari and K. Z. Zamli, "A Harmony Search Based Pairwise Sampling Strategy for Combinatorial Testing," *International Journal of the Physical Science,* vol. 7, pp. 1062-1072, 2012.

[10] R. R. Othman and K. Z. Zamli, "ITTDG: Integrated t-wayTest Data Generation Strategy for Interaction Testing," *Scientific Research and Essays,* vol. 6, pp. 3638-3648, 2011.

[11] F. Glover, "Tabu search-part I," *ORSA Journal on Computing,* vol. 1, pp. 190-206, 1989.

[12] E.-G. Talbi, "A Unified Taxonomy of Hybrid Metaheuristics with Mathematical Programming, Constraint Programming and Machine Learning," in *Studies in Computational Intelligence.* vol. 434, E.-G. Talbi, Ed., ed: Springer, 2013, pp. 3-76.

[13] E. K. Burke, M. Gendreau, M. Hyde, G. Kendall, G. Ochoa, E. Özcan*, et al.*, "Hyper Heuristics: A Survey of the State of the Art," *Journal of the Operational Research Society,* vol. 64, pp. 1695–1724 2013.

[14] E. K. Burke, M. Hyde, G. Kendall, G. Ochoa, E. Ozcan, and J. R. W. . "A Classification of Hyper-heuristic Approaches," in *In Handbook of Meta-Heuristics*, M. Gendreau and J.-Y. Potvin, Eds., ed: Kluwer, 2010, pp. 449-468.

[15] E. K. Burke, G. Kendall, and E. Soubeiga, "A Tabu-Search Hyperheuristic for Timetabling and Rostering," *Journal of Heuristics,* vol. 9, pp. 451-470, 2003.

[16] M. Ayob and G. Kendall, " A Monte Carlo Hyper-Heuristic to Optimise Component Placement Sequencing For Multi Head Placement Machine," in *Proceedings of the International Conference on Intelligent Technologies*, 2003, pp. 132-141.

[17] Y. Jia, M. B. Cohen, M. Harman, and J. Petke, "Learning Combinatorial Interaction Test Generation Strategies Using Hyperheuristic Search," in *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*, 2015, pp. 540-550.

[18] H. Asmuni, E. K. Burke, and J. M. Garibaldi, "Fuzzy Multiple Heuristic Ordering for Course Timetabling," in *Proceedings of the 5th United Kingdom workshop on computational intelligence (UKCI 2005)*, 2005, pp. 302-309.

[19] K. Z. Zamli, F. Din, G. Kendall, and B. S. Ahmed, "An Experimental Study of Hyper-Heuristic Selection and Acceptance Mechanism for Combinatorial t-way Test Suite Generation," *Information Sciences,* vol. 399, pp. 121-153, 2017.

[20] F. Gudino-Penaloza, M. Gonzalez-Mendoza, J. Mora-Vargas, and N. Hernandez-Gress, "Fuzzy Hyperheuristic Framework for GA Parameters Tuning," in *Artificial Intelligence (MICAI), 2013 12th Mexican International Conference on*, 2013, pp. 53-58.

[21] A. Alazzam and H. W. Lewis III, "A New Optimization Algorithm For Combinatorial Problems," *International Journal of Advanced Research in Artificial Intelligence,* vol. 2, pp. 63-68, 2013.

[22] X.-S. Yang and S. Deb, "Cuckoo Search via Levy Flight," in *Proceedings of World Congress on Nature and Biologically Inspired Computing*, 2009, pp. 210-214.

[23] N. R. Sabar and G. Kendall, "Population based Monte Carlo Tree Search Hyper-Heuristic for Combinatorial Optimization Problems," *Information Sciences,* vol. 314, pp. 225-239, 2015.

[24] C.-W. Tsai, W.-C. Huang, M.-H. Chiang, M.-C. Chiang, and C.-S. Yang, "A Hyper Heuristic Scheduling for Cloud," *IEEE Transactions on Cloud Computing,* vol. 2, pp. 236-250, 2014.