Design of an efficient hyper-heuristic algorithm CMA-VNS for combinatorial black-box optimization problems

Fan Xue*
Faculty of Architecture
The University of Hong Kong
Pokfulam
Hong Kong, Hong Kong SAR
xuef@hku.hk

Geoffrey Q Shen
Faculty of Construction and Environment
The Hong Kong Polytechnic University
Hunghom
Kowloon, Hong Kong SAR
bsqpshen@polyu.edu.hk

ABSTRACT

We present a hyper-heuristic algorithm for solving combinatorial black-box optimization problems. The algorithm named CMA-VNS stands for a hybrid of variants of Covariance Matrix Adaptation Evolution Strategy (CMA-ES) and Variable Neighborhood Search (VNS). The framework design and the design profiles of variants of CMA-VNS are introduced to enhance the intensification of searching for conventional CMA-ES solvers. We explain the parameter configuration details, the heuristic profile selection, and the rationale of incorporating machine learning methods during the study. Experimental tests and the results of the first and the second Combinatorial Black-Box Optimization Competitions (CBBOC 2015, 2016) confirmed that CMA-VNS is a competitive hyper-heuristic algorithm.

KEYWORDS

Combinatorial black-box optimization, CMA-VNS, hyper-heuristics, NK-model

*Part of this work was done when the author was with Faculty of Construction and Environment, The Hong Kong Polytechnic University.

GECCO '17 Companion, Berlin, Germany

ACM Reference format:
Fan Xue and Geoffrey Q Shen. 2017. Design of an efficient hyper-heuristic algorithm CMA-VNS for combinatorial black-box optimization problems. In *Proceedings of GECCO '17 Companion, Berlin, Germany, July 15-19, 2017,* 9 pages.
DOI: http://dx.doi.org/10.1145/3067695.3082054

1 INTRODUCTION

The characteristics of optimization problems, such as derivatives of the objective function or linear constraints over variables, contain critical information for finding the optimal solutions [1]. However, such kinds of characteristics are often unavailable, unreliable, or impractical to obtain in many areas of engineering and applied science such as civil engineering, molecular biology and material sciences [17]. Similar difficulties are also encountered in the fields of operations research, such as the configuration of algorithm parameters and the adaptation of heuristics into various applications. Under such circumstances, methods for solving these black-box [8] (or cross-domain [15]) optimization problems are always required.

Researchers from engineering and applied sciences have developed many successful methodologies, such as derivative-free optimization [4] methods, surrogate models, and hyper-heuristics [2], to map the search space of solutions into some model space [4, 22]. Examples include general pattern search algorithms resulting in a significantly better result for protein structure prediction [14], surrogate management frameworks for optimization of cardiovascular geometries in surgical planning and treatment design [12], and life-long learning hyper-heuristics for bin packing [19]. The increasing research interests in the black-box optimization also led to a list of online competition tracks, as shown in Table 1. According to the results of the competitions, such as [6], some algorithms including variants of *DIviding RECTangles* (DIRECT) [4] and *covariance matrix adaptation evolution strategy* (CMA-ES) [7] are among the best solvers for challenging and expensive black-box optimization problems. Poli and Graff [16] pinpointed that taking advantage of instance-specific, dataset-specific, and domain-specific features promisingly is a key to avoiding the pitfall of the *no free lunch* theorems [5, 20] in the circumstance of such competitions.

Table 1: A list of competitions related to black-box optimization

Year	Competition title (URL)	Organizer / Conference
2009–2017	Black-Box Optimization Benchmarking (BBOB) (http://coco.gforge.inria.fr/doku.php)	INRIA, GECCO, CEC
2011	Cross-domain Heuristic Search Challenge (CHeSC) (http://www.asap.cs.nott.ac.uk/external/chesc2011/)	University of Nottingham, OR
2014–2017	Real Parameter Single Objective Optimization (the expensive track) (http://sites.ieee.org/cec2015)	CEC
2015–2016	Combinatorial Black-Box OptimizationCompetition (CBBOC) (http://web.mst.edu/~tauritzd/CBBOC/)	MST, GECCO
2015–2017	Black Box Optimization Competition (BBComp) (http://bbcomp.ini.rub.de/)	GECCO, CEC, EMO

The algorithm presented in this paper aims to promote intensification of searching by a variant of *variable neighborhood search* (VNS) [13] with respect to the semi-adapted covariance matrix model of CMA-ES in the combinatorial black-box optimization, where a part of or the whole set of variables are discrete values. We implemented CMA-VNS for the CBBOC competitions, where the problems are black-box NK-models [9] with n ($50 \leq n \leq 300$) binary variables, generated from four classes including random, Ising Spin Glasses, MAX-kSAT, and Concatenated Traps. We introduce the algorithmic components and different design profiles of CMA-VNS and developed a design and configuration rationale in the context of the CBBOC competitions.

2 THE CMA-VNS HYPER-HEURISTIC

In this section, we describe the algorithmic components, parameter configuration, solution space exploration profiles, implementation tips, and their final integration in the CMA-VNS algorithm. The `C++` source code is available at https://github.com/ffxue/cmavns.

2.1 Low-level heuristic components of CMA-VNS

Figure 1 shows the pseudo code of the framework of CMA-VNS. There are two main blocks in the pseudo code, i.e. the CMA-ES block (line 5–8) and the VNS block (line 9–19). The first CMA-ES block is employed to estimate the overall landscape of solution space, while the latter VNS block mainly handles the ruggedness in the local landscape (i.e. neighborhood) of NK-models. There are five components involved in the two blocks:

- A *bi-population CMA-ES* [11] implemented in the LIBCMAES[1] library for recommending promising candidate solutions with its semi-adapted covariance matrix model,
- An elite set for collecting some (or all) of the best-so-far solutions during searching,
- A *backbone* [24] (result of logical and of binary variables in this case) of the elite set for intensification of searching,
- A *variable candidate set* of local search consisting of binary flips of a given solution and a number of candidates recommended by CMA-ES, with a tabu list of recent flips, and
- An *adaptive acceptance* [10] heuristic for the tolerance of a non-best-so-far solution with a probability.

1: **procedure** CMA-VNS ▷ Note: Main procedure
2: *attributes* ← perceive instance attribute() ▷ Note: Awareness of instance-specific features
3: *params* ← heuristic selection by pre-trained rules(*attributes*) ▷ Note: Core of hyper-heuristic
4: *elites* ←
5: **repeat** ▷ Note: CMA-ES runs first
6: *solutions* ← BIPOPCMAES(*params.lambda*)
7: *elites* += *solutions*.new_best_knowns
8: **until** *params.cmaes_eval* is met
9: **repeat** ▷ Note: Followed by VNS
10: *backbone* ← $\wedge$(*elites*) ▷ Note: Logical and for vectors of binary variables
11: s_0 ← random_start(*backbone*) ▷ Note: Guided by *backbone* of *elites*
12: *candidates* ← FLIP(s_0) $\cup$ bipopCMAES.predict(s_0, *backbone*) $\setminus$ TABU ▷ Note: Candidate set
13: s_0 ← VNS(*candidates*)
14: **if** s_0 is new best known **then**
15: *elites* += s_0
16: **elseif** *params.AA* && ADAPTIVE_ACCEPTANCE(s_0) **then** ▷ Note: Adaptive acceptance
17: **goto** 12
18: **end if**
19: **until** *params.vns_eval* is met
20: **end procedure**

Figure 1: Pseudo code of the framework of CMA–VNS

[1]See https://github.com/beniz/libcmaes.

2.2 Heuristic selection by a dataset-specific parameter configuration: CMA-VNS'15

CMA-VNS has three main parameters, i.e. the proportion of evaluations in the CMA-ES block and the VNS block($p = \frac{cmaes\_eval}{vns\_eval}$), the population size of CMA-ES (*lambda*), and the switch of adaptive acceptance heuristic (AA). In general, a greater proportion p stands for a higher necessity of estimating the overall landscape by CMA-ES, while a smaller p means that the algorithm needs to focus more on intensified local search of VNS.

In the CBBOC 2015 entry of CMA-VNS, we employed a dataset-specific parameter configuration mechanism to adapt the CMA-VNS to the instances in the dataset of the CBBOC competition. For example, in CBBOC 2015, we generated over 10,000 random training instances from the official API[2]. Each black-box instance has n variables and allows mn^2 times of evaluations, where m indicates the abundant level of evaluations. Thus, a small m indicates that the instance is expensive, while a great m stands for relatively affordable evaluations. Three decision trees, as shown in Figure 2, were trained offline for the three variables from computational results on the training instances with m and n as the two decision attributes. The decision trees were generated by the best-first decision tree learning method [18] implemented in Weka[3].

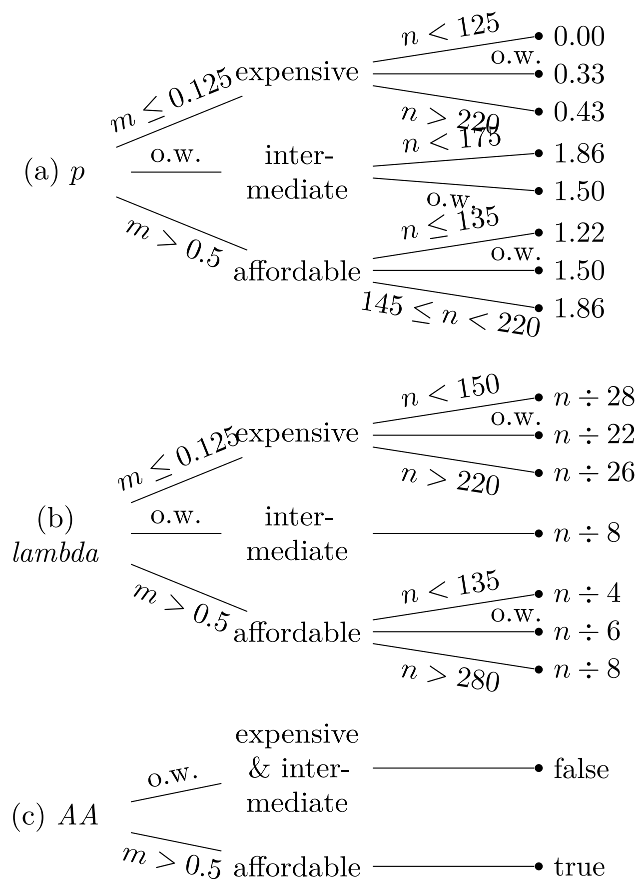


Figure 2: The decision trees of parameter configuration of CMA-VNS trained for CBBOC 2015

---

[2] See: https://github.com/cbboc.

[3] See: http://www.cs.waikato.ac.nz/ml/weka/.

The decision trees in Figure 2 show that all the parameters are more sensitive to m instead of n. The first tree denotes that the proportion p should be considerably smaller when the instances are expensive, whereas in some small-scale ($n < 125$) and expensive evaluation ($m \leq 0.125$) cases p should be set to 0. In other words, it is not worthy to allocate limited evaluations to adaptation of the covariance matrix by CMA-ES for expensive and small-scale instances. In the relatively affordable ($m > 0.5$) and intermediate ($0.125 \leq m \leq 0.5$) instances, p can vary from 1.22 to 1.86. A greater portion of CMA-ES ($p > 1.2$) denotes that the CMA-ES block should consume more evaluations than the VNS block. The second tree shows that the population size *lambda* of CMA-ES also should also be considerably smaller when the instances are expensive. In comparison, the suggested values of *lambda* are three to six times greater in the relatively affordable and intermediate instances. The third tree shows that the adaptive acceptance should be enabled only in the relatively affordable evaluation ($m > 0.5$) instances. One can also find that the intermediate category is actually much closer to the affordable category than the expensive category regarding the trained configuration of parameters of CMA-VNS'15.

According to the decision trees, CMA-VNS can result in a finite number of combinations of the low-level heuristics in model space. Even the three parameters are set to free, the size of the model space is no more than countable infinity ($\aleph_0$). According to Xue's definition [22] of subcategories of hyper-heuristics, i.e., heuristic selection and heuristic generation, CMA-VNS is a heuristic (combinations of parametric components) selection algorithm, a subcategory of hyper-heuristics.

2.3 Heuristic profile selection: CMA-VNS'16

Profiles and portfolios are known as effective mechanisms for solving challenging problems, in particular for the competition entries such as SATzilla [21]. In the entry of CBBOC 2016, five different profiles of CMA-VNS were designed as follows.

Table 2: A table of deciding the best heuristic profiles of CMA-VNS trained for the CBBOC 2016

Resource of evaluations (m)	Instance category	Dimension (n) $2^{5.5}$	2^{6}	$2^{6.5}$	2^{7}	$2^{7.5}$	2^{8}	$2^{8.5}$
$m \leq 0.125$	Expensive	P2	P2	P2	P2	P2	P1	P1
$0.125 < m \leq 0.5$	Intermediate	P2	P2	P2	P2† P3‡	P3	P1	P1
$m > 0.5$	Affordable	P2† P3‡	P3	P3	P2	P2	P1	P1

†: When v was lower than a threshold ($v \leq 1.50$); ‡: Otherwise.

P1 The profile P1 is exactly the same as the 2015 entry of CMA-VNS as described in Subsections 2.1 and 2.2.

P2 The profile P2 is an intensification (emphasizing the local search in the VNS block) version of P1. In particular, the adaptive acceptance heuristic is set to higher tolerance to suppress frequent restarts of local search. A random k-point crossover operation is implemented to replace the random start from the backbone of the elite set under certain circumstances. The size of the elite set is also controlled to support the effectiveness of the backbone. The space of the initial solutions to the VNS block is hence considerably increased.

P3 The profile P3 is a diversification (emphasizing the restarts in the VNS block) of P2. The profile P3 introduces an elite set pruning after very new best-so-far solution. Hence the elite set is usually much smaller, and the backbone is longer. The profile P3 also limits the size of the candidate set by reducing the number of solution recommended by CMA-ES. The overall effect aims to jump out of the attraction of local optima.

P4 The profile P4 is a variant of P3. The profile P4 abandons the elite set pruning and resets the composition of the candidate set back to that in P1.

P5 The profile P5 is an iterated version of CMA-VNS. The end of the VNS block, in profile P5, will trigger the next iteration of CMA-VNS, i.e. both the CMA-ES block and the VNS block instead of restarts of local search. Also, the variables in the backbone of the elite set will be set as fixed values in the next iteration. In other words, the dimension n is reduced after an iteration. The maximum number of maximum available evaluations is also updated accordingly.

In the 2016 entry of CMA-VNS, three decision attributes were considered for the parameter configuration of each profile and the selection of the heuristic profiles. The attributes were the problem dimension (n), the resource of evaluations (m), and the best objective value per dimension ($v = \frac{\textit{best-so-far}}{n}$). The new attribute v was introduced to indicate the latent nature of the problem instance. Table 2 shows the best heuristic profiles of CMA-VNS regarding aggregated average results of solutions. The results were calculated from a large number of experimental tests based on generated training instances of CBBOC 2016.

Table 2 of heuristic profile selection shows that two attributes n and m can discriminate almost all cases. The profile P1 returns the best results for large-scale instances (approximately $n > 220$), while the profile P2 works very well for other instances roughly $n \leq 220$). The profile P3 is slightly better than P2 in some cases, including small-scale (approximately $n \leq 110$) and relatively affordable evaluation ($m > 0.5$) instances and intermediate-scale (approximately $110 < n \leq 220$) and intermediate evaluation ($0.125 < m \leq 0.5$) instances. The attribute v is needed in rare cases to choose between P2 and P3, such as the smallest ($n \approx 50$) and relatively affordable evaluation instances and a few intermediate-scale (approximately $n \approx 130$) and intermediate evaluation instances. The profile P3 is preferred when $v > 1.50$, which stands for the latent nature of an instance is quite different from random instances of NK-models. The profiles P4 and P5 are completely dominated in the range of the training instances of CBBOC 2016.

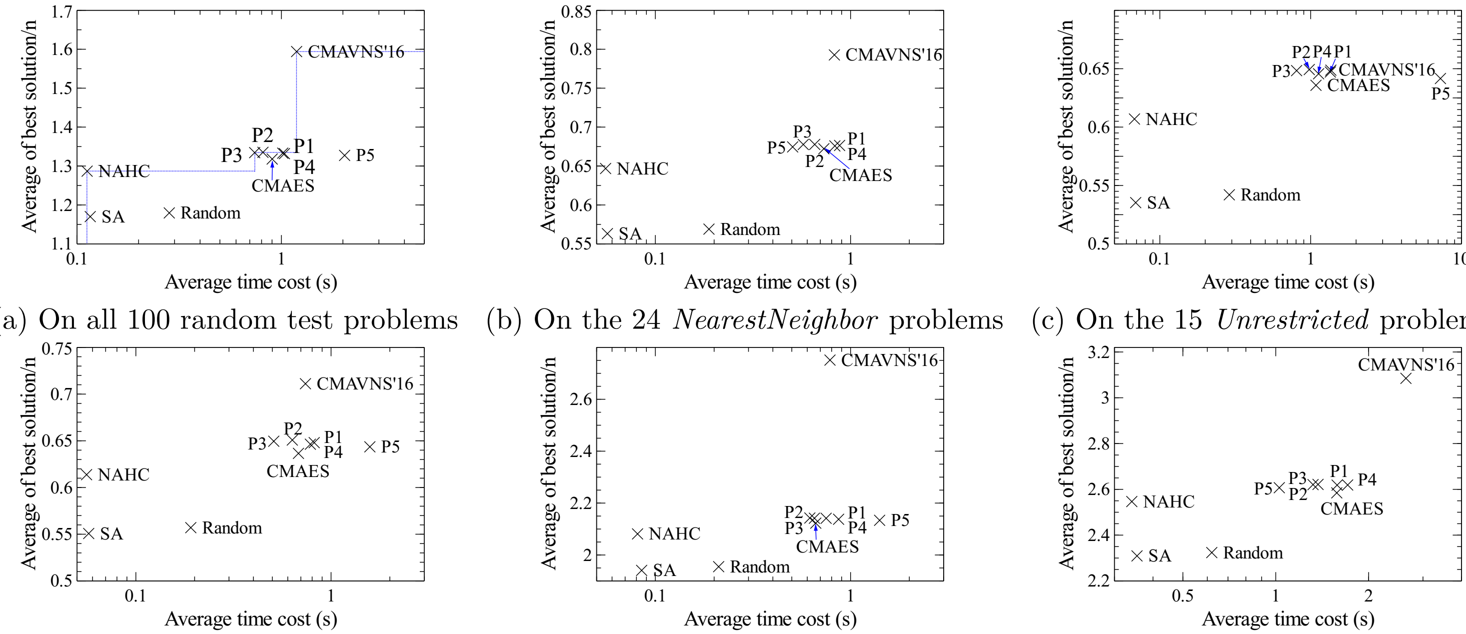
2.4 Design and configuration rationale

The main framework of CMA-VNS, which follows the Pearl Hunter hyper-heuristic [3], is a heuristic selection trained by offline or online learning. The Pearl Hunter algorithm always performs diversification of searching (or "snorkeling") before the expensive intensification of model (low-level heuristics) space exploration (or "dive"). Meanwhile, the Pearl Hunter tries to "recognize" the model space for the instance into several categories and take corresponding actions of search.

Balancing the diversification and the intensification is one of the most important issues for the design of a searching algorithm. For example, experimental tests showed that intensification of the CMA-ES was limited in the rugged landscape of NK-models in the later stage of problem-solving. Hence a typical intensification scheme such as VNS has the chance to improve the CMA-ES. Empirical comparison and analysis of different profiles of diversification and intensification may also improve the overall performance of an algorithm.

Meta-models such as derivative-free optimization methods, surrogate models, and hyper-heuristics are proven successful for many black-box or cross-domain optimization applications. Such a model map the solution space of instances onto their model space such as combinations of low-level heuristics. The mappings usually depend on the objective function value and a few other attributes such as the dimension of variables. If the mapping can be well-established before testing, such as meta-model can be very competitive against some *ad hoc* algorithms, e.g. the Pearl Hunter also found a number of new best-known solutions for the staff shift scheduling dataset [3].

Incorporating machine learning techniques, which focus on the correlation between features (characteristics of problems) and classification (indicating the performance of components and profiles of a meta-model), into heuristic selection leads to an efficient estimation of the mapping between the solution space and the model space. Hence, machine learning techniques can be found in many successful meta-models including



(a) On all 100 random test problems (b) On the 24 *NearestNeighbor* problems (c) On the 15 *Unrestricted* problems (d) On the 21 *Separable* problems (e) On the 23 *Mesh* problems (f) On the 17 *SAT_like* problems

Figure 3: The results of the average of best solution (higher is better), time cost (lower is better), and the Pareto frontier (dotted line in (a)) of tests of CMA-VNS and other algorithms on 100 randomly generated test problems, 50 instances for each problem, where Random, SA, and NAHC were three test algorithms implemented in the CBBOC framework

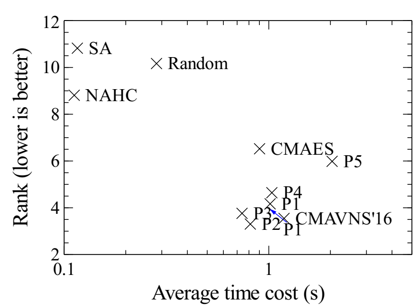


Figure 4: The results of the overall rank (lower is better) and time cost (lower is better) of the results in Figure 3 based on CBBOC's Schulze Voting method

CMA-ES and SATzilla. Characteristics of problems, datasets, or domains are, in both theory [16] and practice, keys for black-box optimization algorithm design. Furthermore, matured machine learning methods can provide not only reliable heuristic selection rule sets for the algorithm but also inspirational insights for researchers. For example, the decision trees about p and *lambda* in Figure 2 shows that the intermediate and the affordable categories which require similar configurations of the two parameters may have comparable difficulty in problem-solving.

A large number of training instances and experiments are thus necessary for training the machine learning methods for a dataset or domain, just like one needs a large number of sample points to reconstruct a solution space of an instance. Experiments were conducted on the instances to compare the average performances, regarding the mean best objective values, of the three profiles.

2.5 Implementation tips

A solution cache was employed in CMA-VNS to record solutions (bit vectors in C++) and their objective function values. Before spending any resource of valuable evaluations, a solution is tested in the cache first. A successful hit can save both time and adequate evaluations. Thanks to the efficient implementation of *std::unordered_map* in C++11 standard, such a solution cache can be created in a single line of code.

The offline trained heuristic selection and heuristic profile selection were prepared for the (online) no-training tracks. For the short-training and long-training tracks, a good practice was leaving one or two key parameters, such as the proportion p in CMA-VNS'15 and the profile in CMA-VNS'16, to determine during the online training process.

2.6 Experimental tests and the results the CBBOC competitions

We tested the CMA-VNS in the CBBOC development framework, in comparison with the five profiles, CMA-ES, and three test algorithms in CBBOC framework — Random, *simulated annealing* (SA) and *next ascend hill-climbing* (NAHC). 100 problems were randomly generated, 50 instances for each problem. The results are shown in Figure 3 and 4.

Figure 3 shows the average value of best solutions (higher is better) over the dimension n against time cost. The dotted line in Figure 3 (a) denotes a Pareto frontier of the tested algorithms. It can be observed that SA, Random, CMA-ES, P1 , P4 , and P5 were generally dominated in both solution quality and time. On the top of the Pareto frontier, CMA-VNS with trained profile selection achieved significant increments in solution quality for many problems. By comparing the sub-figures, one can find the spatial relationships of the markers of algorithms are stable against different problem classes, but a weakness of CMA-VNS's heuristic profile selection can be found for the *Unrestricted* class. Although the five profiles are different hybrids of CMA-ES and VNS local search, the five profiles cannot be clearly distinguished from the entry of CMA-ES in any sub-figures in Figure 3. Figure 4 shows the average rank of each algorithm which was calculated with Schulze Voting method. The order of ranking, in which CMA-VNS was ranked the third, was considerably different from the order of average solution quality.

The first entry of CMA-VNS with the configuration exactly shown in Figure 2 (or P1) won three tracks in 2015. The 2016 entry with the profile selection shown in Table 2 won two more tracks in 2016.

3 CONCLUSION

This paper introduces the design of CMA-VNS, a hyper-heuristic algorithm for combinatorial black-box optimization. We first selected five low-level heuristic components and designed a hybrid of CMA-ES and VNS. The parameters of the components were thoroughly trained and configured for the CBBOC competitions. Moreover, four different design profiles were derived by adjusting the diversification and intensification of the first version of CMA-VNS. The design and the rationale have been confirmed by experiments and the results of competitions.

One interesting direction for future research is a learning-based automated hyper-heuristic development library. Since the main frameworks of CMA-VNS and Pearl Hunter are more or less the same, they might be automatically produced from one template of hyper-heuristics. Another direction can be some industrial applications of hyper-heuristics such as automated 3D modeling of civil infrastructures [23].

REFERENCES

[1] Dimitri P Bertsekas. 1999. *Nonlinear programming.* Athena scientific Belmont, Belmont.

[2] Edmund Burke, Graham Kendall, Jim Newall, Emma Hart, Peter Ross, and Sonia Schulenburg. 2003. Hyper-heuristics: An emerging direction in modern search technology. In *Handbook of metaheuristics.* Springer, 457–474. DOI:http://dx.doi.org/10.1007/0-306-48056-5_16

[3] Ching-Yuen Chan, Fan Xue, WH Ip, and CF Cheung. 2012. A hyper-heuristic inspired by pearl hunting. In *Learning and intelligent optimization.* Springer, 349–353. DOI:http://dx.doi.org/10.1007/978-3-642-34413-8_26

[4] Andrew R Conn, Katya Scheinberg, and Luis N Vicente. 2009. *Introduction to derivative-free optimization.* SIAM, Philadelphia, PA, USA.

[5] Joseph C Culberson. 1998. On the futility of blind search: An algorithmic view of "no free lunch". *Evolutionary Computation* 6, 2 (1998), 109–127. DOI:http://dx.doi.org/10.1162/evco.1998.6.2.109

[6] Nikolaus Hansen, Anne Auger, Raymond Ros, Steffen Finck, and Petr Pošík. 2010. Comparing results of 31 algorithms from the black-box optimization benchmarking BBOB-2009. In *Proceedings of the 12th annual conference companion on Genetic and evolutionary computation.* ACM, 1689–1696. DOI:http://dx.doi.org/10.1145/1830761.1830790

[7] Nikolaus Hansen, Sibylle D. Müller, and Petros Koumoutsakos. 2003. Reducing the Time Complexity of the Derandomized Evolution Strategy with Covariance Matrix Adaptation (CMA–ES). *Evolutionary Computation* 11, 1 (March 2003), 1–18. DOI:http://dx.doi.org/10.1162/106365603321828970

[8] Deng Huang, Theodore T Allen, William I Notz, and Ning Zeng. 2006. Global optimization of stochastic black-box systems via sequential kriging meta-models. *Journal of global optimization* 34, 3 (2006), 441–466. DOI:http://dx.doi.org/10.1007/s10898-005-2454-3

[9] Stuart A Kauffman. 1993. *The origins of order: Self-organization and selection in evolution.* Oxford University Press, New York, USA.

[10] Ahmed Kheiri, Ender Özcan, and Andrew J Parkes. 2014. A stochastic local search algorithm with adaptive acceptance for high-school timetabling. *Annals of Operations Research* (2014), 1–17. DOI:http://dx.doi.org/10.1007/s10479-014-1660-0

[11] Ilya Loshchilov, Marc Schoenauer, and Michèle Sebag. 2013. Bi-population CMA-ES agorithms with surrogate models and line searches. In *Proceedings of the 15th annual conference companion on Genetic and evolutionary computation.* ACM, 1177–1184. DOI:http://dx.doi.org/10.1145/2464576.2482696

[12] Alison L Marsden, Jeffrey A Feinstein, and Charles A Taylor. 2008. A computational framework for derivative-free optimization of cardiovascular geometries. *Computer Methods in Applied Mechanics and Engineering* 197, 21 (2008), 1890–1905. DOI:http://dx.doi.org/10.1016/j.cma.2007.12.009

[13] Nenad Mladenović and Pierre Hansen. 1997. Variable neighborhood search. *Computers & Operations Research* 24, 11 (1997), 1097–1100. DOI:http://dx.doi.org/10.1016/S0305-0548(97)00031-2

[14] Giuseppe Nicosia and Giovanni Stracquadanio. 2008. Generalized pattern search algorithm for peptide structure prediction. *Biophysical journal* 95, 10 (2008), 4988–4999. DOI:http://dx.doi.org/10.1529/biophysj.107.124016

[15] Gabriela Ochoa, Matthew Hyde, Tim Curtois, Jose A Vazquez-Rodriguez, James Walker, Michel Gendreau, Graham Kendall, Barry McCollum, Andrew J Parkes, Sanja Petrovic, and others. 2012. Hyflex: A benchmark framework for cross-domain heuristic search. In *European Conference on Evolutionary Computation in Combinatorial Optimization.* Springer, 136–147. DOI:http://dx.doi.org/10.1007/978-3-642-29124-1_12

[16] Riccardo Poli and Mario Graff. 2009. There is a free lunch for hyper-heuristics, genetic programming and computer scientists. In *European Conference on Genetic Programming.* Springer, 195–207. DOI:http://dx.doi.org/10.1007/978-3-642-01181-8_17

[17] Luis Miguel Rios and Nikolaos V Sahinidis. 2013. Derivative-free optimization: A review of algorithms and comparison of software implementations. *Journal of Global Optimization* 56, 3 (2013), 1247–1293. DOI:http://dx.doi.org/10.1007/s10898-012-9951-y

[18] Haijian Shi. 2007. *Best-first decision tree learning.* Master's thesis. The University of Waikato. http://hdl.handle.net/10289/2317

[19] Kevin Sim, Emma Hart, and Ben Paechter. 2015. A lifelong learning hyper-heuristic method for bin packing. *Evolutionary computation* 23, 1 (2015), 37–67. DOI:http://dx.doi.org/10.1162/EVCO_a_00121

[20] David H Wolpert and William G Macready. 1997. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation* 1, 1 (1997), 67–82. DOI:http://dx.doi.org/10.1109/4235.585893

[21] Lin Xu, Frank Hutter, Holger H Hoos, and Kevin Leyton-Brown. 2008. SATzilla: portfolio-based algorithm selection for SAT. *Journal of artificial intelligence research* 32 (2008), 565–606. DOI:http://dx.doi.org/10.1613/jair.2490

[22] Fan Xue. 2013. *A suboptimum- and proportion-based heuristic generation method for combinatorial optimization problems.* Ph.D. Dissertation. The Hong Kong Polytechnic University, Hong Kong. http://hdl.handle.net/10397/6371

[23] Fan Xue, Ke Chen, Diandian Liu, Yuhan Niu, and Weisheng Lu. 2016. An optimization-based semantic building model generation method with a pilot case of a demolished construction. In *Proceedings of the 21st international conference on advancement of construction management and real estate (CRIOCM 2016).* CRIOCM, Springer (to appear), Hong Kong.

[24] Weixiong Zhang and Moshe Looks. 2005. A Novel Local Search Algorithm for the Traveling Salesman Problem That Exploits Backbones. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI'05).* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 343–348. http://dl.acm.org/citation.cfm?id=1642293.1642348