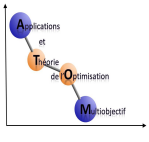

# SLS2019 : International Workshop on Stochastic Local Search Algorithms

*12-13 Sep 2019 Villeneuve d'Ascq (France)*

## Proceedings



## List of accepted papers

Darrell Whitley, Swetha Varadarajan and Gabriela Ochoa. A Micro-Level Frequency Analysis for the Traveling Salesman Problem

Valentin Owczarek, Patrick Franco and Remy Mullot. A genetic algorithm to solve a space-filling curve problem

Raphael Patrick Prager, Pascal Kerschke and Heike Trautmann. Towards Automated Configuration of CMA-ES by Means of Exploratory Landscape Analysis and Machine Learning

Rabin Sahu and Nadarajen Veerapen. Stochastic Joint Replenishment Planning Optimization with an Iterated Local Search

Federico Pagnozzi and Thomas Stützle. Automatic Design of Hybrid Stochastic Local Search Algorithms applied to the permutation flowshop problem

Anna Ouskova Leonteva, Pierre Parrend, Anne Jeannin-Girardon and Pierre Collet. Fast evolutionary algorithm for multi-objective industrial problems

Weerapan Sae-Dan, Marie-Eléonore Kessaci, Nadarajen Veerapen and Laetitia Jourdan. Automatic Configuration of a Dynamic Hill Climbing Algorithm

Sara Tari, Basseur Matthieu and Adrien Goeffon. Sampled Walk: A Partial Neighborhood Search Strategy

Ala-Eddine Yahiaoui, Aziz Moukrim, Mehdi Serairi and Marc Demange. A Hybrid Heuristic for the Synchronized Team Orienteering Problem with Time Windows

Vincent Hénaux, Adrien Goëffon and Frédéric Saubion. Evolving Stochastic Hill Climbers
Hojjat Rakhshani, Lhassane Idoumghar, Julien Lepagnot and Mathieu Brévilliers. A learning-based algorithm for continues optimization

Soheila Ghambari, Lhassane Idoumghar, Laetitia Jourdan and Julien Lepagnot. PRM-based differential evolution for UAV path planning

İsmail Sevim. An engine for configurations of simulated annealing algorithm

Laurent Parmentier, Olivier Nicol, Laetitia Jourdan and Marie-elenore Kessaci
TPOT-SH: a Faster Optimization Algorithm to Solve the AutoML Problem on Large Datasets

# A Micro-Level Frequency Analysis for the Traveling Salesman Problem

D. Whitley[1], S. Varadarajan[1] and G. Ochoa[2]

[1] Colorado State University, Fort Collins, CO, 80523, USA
whitley@colostate.edu, swetha.varadarajan@colostate.edu
[2] Computing Science and Mathematics, University of Stirling, FK9 4LA, Scotland
gabriela.ochoa@cs.stir.ac.uk

## 1 Introduction

A number of papers have explored the idea that the distribution of local optima can help to explain both problem difficulty as well as the effectiveness of inexact heuristics when searching instances of the Traveling Salesman Problem (TSP). Most of these papers use a "macro-level" landscape analysis. We present a "micro-level" analysis that counts the frequency of edges found in the global optimum and how often these edges are sampled during search. This frequency analysis may provide a better understanding of algorithm performance as well as problem difficulty.

Bose et al. [1] introduced the concept of the "Big Valley" distribution of local optima. The definition of a local optimum depends on the choice of a local move operator. Briefly, for the TSP the "Big Valley" hypothesis states that local optima that are closer to the global optimum in evaluation are (subject to some variation) also closer to the global optimum in "bond distance" (the number of shared edges). Hains et al. showed that a classic TSP, instance ATT532, displays not only one large valley or "funnel," but additional funnels, although these funnels were relatively small [4]. Ochoa et al. [7] also documented a multi-funnel structure in TSP instances.

Landscape analysis might be very relevant to Stochastic Local Search methods such as Chained LK and LKH. But it is less clear that it can explain the behavior of population based methods, such as the EAX genetic algorithm [6]. This paper explores the idea that a micro-level analysis can better explain the difficulty (or ease) of solving an instance of the TSP. This micro-level analysis focuses on "edge frequency" across local optima induces by 2-opt. This analysis assumes the global optimum is known, and then asks how often edges found in the global optimum are sampled during search. This analysis shifts the focus from the level of local optima, to the level of individual edges. By combining frequency statistics and structural constraints we not only better explain why certain inexact methods work well, we can potentially discover better methods for exploiting edge frequency information.

Recently, a new genetic algorithm was introduced for the Traveling Salesman Problem. The *Mixing Genetic Algorithm* (MGA) uses Generalized Partition Crossover (GPX) [8]. What is novel about MGA is that it generates both the *best possible offspring* that GPX can produce as well as the *worst possible offspring* that GPX can produce. This is done such that no edges are lost during recombination. Instead migration occurs within the population, so that good solutions recombine with other good solutions, and poorer solutions are recombined with other poorer solutions. This idea is related to Culberson's "Gene Invariant" Genetic Algorithm (GIGA) [2].

To be successful, MGA (in it purest form) requires that all of the edges needed to find the global optimum must be present in the initial population. In Table 1, we empirically ask how large the initial population must be to ensure that all of the edges found in the global optimum are also found in a population improved using 2-opt. For the 20 instances shown in Table 1, 16 of the 20 (80%) needed a population of only 256 individuals or less. The MGA was also able to solve most of these problems to optimality using fewer recombinations than the EAX algorithm.

What this table does not indicate is how *many* of the edges in the initial population were also edges found in the global optimum. We have examined all of instances in Table 1 and found that for almost all problems approximately 70% of edges that occur in local optima are also edges found in the global optimum. An example for TSP instance ATT532 is shown in Figure 1.

The illustrations in Figure 1 provide insight into problem difficulty. Overwhelmingly, the most frequently sampled edges in the population are the edges in the global optimum. This is true for both EAX and MGA. For example, edges that appear in 50% or more of the population make

**Table 1.** List of problem instances and the corresponding smallest population size that has all the edges found in the global optimum P* after performing EAX based 2-opt on the initial population.

| Cluster Problem | Pop Size | PCB Problem | Pop Size | PCB Problem | Pop Size | City Problem | Pop Size | City Problem | Pop Size |
|---|---|---|---|---|---|---|---|---|---|
| C1k.0 | 256 | pcb442 | 16 | u1817 | 128 | att532 | 64 | vm1748 | 128 |
| C3k.0 | 256 | pcb1173 | 256 | d2103 | 16384 | pr1002 | 128 | rl1889 | 128 |
| C3k.1 | 512 | fl1577 | 4096 | u2319 | 64 | vm1084 | 512 | pr2392 | 128 |
| dsj1000 | 128 | d1655 | 256 | pcb3038 | 128 | rl1304 | 64 | fnl4461 | 256 |

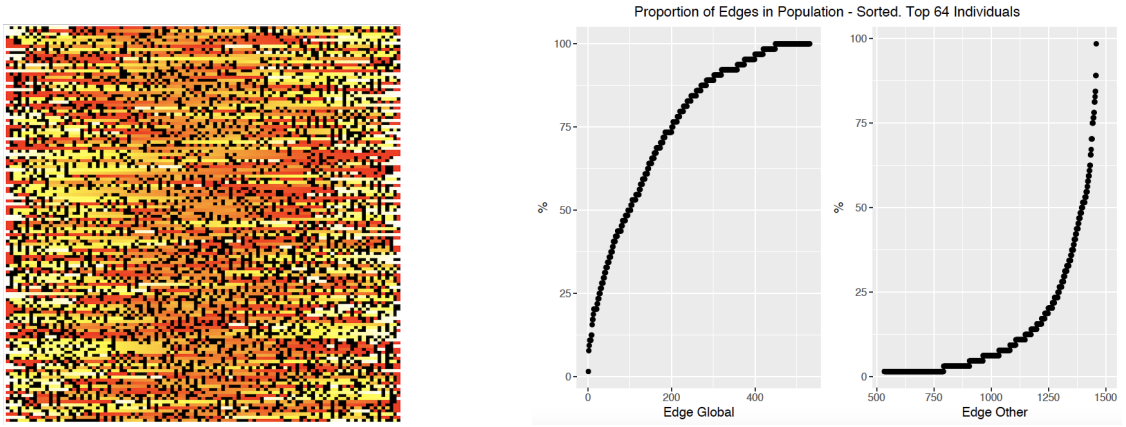



**Fig. 1.** The leftmost figure is a "heatmap." Edges found in the global optimum are colored from yellow to red; edges that are not found in the global optimum are colored black. Each row of the heatmap represents one local optimum. The rightmost figure shows the frequency distribution of edges from the global optimum ("Edge Global") that also appear in the best 64 local optima out of the 512 that were initially sampled; it also shows the frequency distribution of edges that do not appear in the global optimum ("Edge Other"). Edges that appear in the global optimum appear with much higher frequency.

up more than 400 of the 532 edges that appear in the global optimum for instance att532, but include only 60 edges that are not in the global optimum. More than 70% of the **total** edges in the population are also found in the global optimum; however, more than 33% of all the **unique** edges in the population are found in the global optimum (532 out of 1470). Similar results hold for almost all problems in Table 1.

Our analyses suggest that population based methods that intelligently exploit edge frequency information should have a clear advantage over local search methods that explore from a single solution. While EAX and MGA use different recombination operators, both EAX and GPX are exploring a very restricted space that is densely populated with (often all of) the edges that appear in the global optimum. This may also help to explain the performance of Ant Colony Optimization (ACO) [3]; the pheromone trails used by ACO are also frequency based. Finally, this micro-level analysis could help to explain why some TSP instances are easier to solve to optimality than one might expect given that the TSP is an NP-Hard optimization problem [5].

## References

1. K. Boese and A. Kahng and S. Muddu (1994) A new adaptive multi-start technique for combinatorial global optimizations. *Operations Research Letters* 16: 101–113.
2. J. Culberson (1992) Gene invariance: A new paradigm for genetic algorithm design. *Technical Report, University of Alberta.*
3. M. Dorigo and L.M. Gambardella (1997) Ant Colonies for the Travelling Salesman Problem. *Biosystems*, 43(2):73-81.
4. D. Hains and D. Whitley and A. Howe (2011) Revisiting the big valley search space structure in the TSP. *Journal of the Operational Research Society* 62(2), 305–312.
5. A. Mu and J. Dubois-Lacoste and H. Hoos and T. Stutzle (2018) On the empirical scaling of running time for finding optimal solutions to the TSP. *Journal of Heuristics*, 24(6):879-898. 2018
6. Y. Nagata and S. Kobayashi (2013) A Powerful Genetic Algorithm Using Edge Assembly Crossover for the Traveling Salesman Problem INFORMS Journal On Computing. 25(2): 346-363.
7. G. Ochoa and N. Veerapen (2016) Deconstructing the Big Valley Search Space Hypothesis. *EvoCOP: Evolutionary Computation in Combinatorial Optimization.* Springer, LNCS, 9595, pp 58-73.
8. S. Varadarajan and D. Whitley (2019) A Massively Parallel Mixing Genetic Algorithm for the TSP. GECCO 2019.

# A genetic algorithm to solve a space-filling curve problem

V. Owczarek, P. Franco and R. Mullot

La Rochelle université, Laboratoire L3i, 17031 La Rochelle
valentin.owczarek1@univ-lr.fr

## 1 Space-filling curve: A pattern driven algorithm

A space-filling curve (SFC) is an ordering function which creates a bijection between integers $I$ and points $X$ lying in a D-dimensional regular grid. Let consider a point $\mathbf{X}_a \in \mathbf{X} = [0, 2^n - 1]^D$, the associate integer corresponds to $\mathbf{I}_a \in \mathbf{I} = [0, 2^{nD} - 1]$ where $n$ is the order of the curve. This ordering has the property to conserve the distance, i.e. close points in the space have close indices. This property known as the locality preserving. At this time, the Hilbert curve defined by D. Hilbert [1] in 1891 achieve the best locality preserving level [2]. In various domain, many applications are drawn on the locality preservation properties of space-filling curves. For example, one could concern global optimization, data visualization, image compression and encryption.

Recently, G. Nguyen [3] proposed a new algorithm to compute SFCs where the curve at order $n = 1$ is used to recursively define the bijection between the $D$-dimensional grid and the indices. This new formulation is able to define any Hilbert like SFC, by using any pattern that respects the adjacency rule: if $\mathbf{I}_a$ and $\mathbf{I}_b$ are separated by one unit then their associate points $\mathbf{X}_a$ and $\mathbf{X}_b$ are separated by a unit distance. The well know Hilbert curve is defined by using the Reflected binary gray code (RBG).

Using this new definition of SFC, relevant studies have been performed by P. Franco [4] to investigate the influence of the pattern for the locality preserving level of the $n$ order curve. The results proved that it is possible to create comparable or better SFC than the regular Hilbert curve which is considered, in the literature, as the best locality preserving curve.

In fact, finding patterns which have a high level of locality preservation level is equivalent to find all Hamiltonian paths in a hypercube graph, which is NP-complete according to R. Karp [5]. A hypercube graph in dimension $D$ is defined by $2^D$ nodes along with $D\ 2^{D-1}$ edges, and each node has a degree of $D$. Obviously, this problem can not be solved by regular graph traversal techniques when $D > 5$. Moreover, our goal is not to find one unique pattern but a set of non-dominated solutions according to the Faloutsos criteria [2] as

$$F_r(\mathbf{X}) = \frac{1}{2^D} \sum_{k,l \in [2^D],\ k<l,\ d(k,l) \leq r} \max\{d(\mathbf{X}_k, \mathbf{X}_l)\}, \tag{1}$$

where $\mathbf{X}$ is the set of points in the D-dimensional regular grid ordered by a SFC and $d$ is the Manhattan distance.

In the next section, different stochastic local search algorithms are briefly explained and results of our experiments are discussed.

## 2 Stochastic local search algorithms for hamiltonian chain

In our work, genetic algorithms (GAs)[6, 7] are used to find hamiltonian chains. GAs use successively mutation/crossover and selection on a population of solutions to find the optima. In every algorithm, the mutation operator is 2-opt defined by Georges A. Croes [8] and the crossover is single point. The algorithms described in Table 1, are different in terms of selection and fitness function. The distance of the chain is denoted by $|C|$ and the Faloutsos criteria at radius $\alpha$ by $F_\alpha$.

In the elitist selection method "Top 100" the first hundred hamiltonian chain corresponding at each generation to the hundred lowest fitness score are conserved to be the next generation. Whereas "Non-dominated set" stands for the conservation over generations of the set of all non-dominated solutions.

**Table 1.** The different GAs used to find hamiltonian paths, according to $|C|$ the length of a path and $F_\alpha$, the Faloutsos criteria.

| Name | Type | Selection method | Fitness Function |
|---|---|---|---|
| *Classic* | Single-objective | Top 100 | $\|C\|$ |
| *Sum* | Single-objective | Top 100 | $\|C\|^2 + \sum_{i=3}^{6} F_i$ |
| *Multi* | Multi-objective | Non-dominated set | $\{F_3.\|C\|, ..., F_6.\|C\|\}$ |

According to Table 1, Each of them differs in term of selection and fitness function. *Classic* uses only $|C|$ whereas *Sum* and *Multi* use the Faloutsos criteria $F_\alpha$ (Equation 1). *Multi* is a multi-objective GA version with a non-dominated set selection method.

For each algorithm the non-dominated set of the results of 1000 runs, for $D = 5$, is saved. Thus three created fronts are compared, as proposed in [9], using the hypervolume (to be maximized) and the additive epsilon (to be minimized) indicators.

For the hypervolume indicator, *Multi* is better with a score of 0.1254 compared to 0.1197 and 0.0772 for respectively *Sum* and *Classic*. The additive epsilon indicator point out the same performances: *Multi* reached the first place with a score of 0.0625 whereas *Sum* and *Classic* respectively obtained 0.0937 and 0.1875 scores.

The results are promising: *Multi* provided close sets to the exact known front, according to the hypervolume and the additive epsilon indicators. Similarly, *Sum* reached comparable results which would be consistent for further improvements. Indeed extended investigations will be ensure to confirm the results on grids of higher dimension. Moreover, the non-dominated set found during this study, on dimension $D = 5$, shows the non-optimality of the Hilbert curve pattern: the associated Faloutsos scores are dominated by several other patterns.

## References

1. David Hilbert. Ueber die stetige Abbildung einer Line auf ein Flächenstück. *Mathematische Annalen*, 38(3):459–460, 1891.
2. C. Faloutsos and S. Roseman. Fractals for Secondary Key Retrieval. In *Proceedings of the Eighth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, PODS '89, pages 247–252. ACM, 1989.
3. Giap Nguyen. *Courbes remplissant l'espace et leur application en traitement d'images*. PhD thesis, Université de La Rochelle, 2013.
4. Patrick Franco, Giap Nguyen, Remy Mullot, and Jean-Marc Ogier. Alternative patterns of the multi-dimensional Hilbert curve. *Multimedia Tools and Applications*, 77(7):8419–8440, 2018.
5. Richard M. Karp. *Reducibility among Combinatorial Problems*, pages 85–103. Springer US, Boston, MA, 1972.
6. K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, April 2002.
7. Bilel Derbel, Arnaud Liefooghe, Qingfu Zhang, Sébastien Verel, Hernan Aguirre, and Kiyoshi Tanaka. A set-oriented MOEA/D. In *GECCO 2018 - Genetic and Evolutionary Computation Conference*, pages 617–624, Kyoto, Japan, July 2018. ACM Press.
8. G. A. Croes. A method for solving traveling-salesman problems. *Operations Research*, 6(6):791–812, 1958.
9. J. Knowles, L. Thiele, and E. Zitzler. A Tutorial on the Performance Assessment of Stochastic Multiobjective Optimizers. TIK Report 214, Computer Engineering and Networks Laboratory (TIK), ETH Zurich, 2006.

# Towards Automated Configuration of CMA-ES by Means of Exploratory Landscape Analysis and Machine Learning

R.P. Prager, P. Kerschke and H. Trautmann

University of Muenster, 48149 Münster
{raphael.prager, kerschke, trautmann}@wi.uni-muenster.de

## 1 Extended Abstract

The field of optimization is focused by various research areas, such as operations research, evolutionary computation as well as computer science and mathematics in general. A variety of different strategies to tackle such problems have emerged in the last decades. These approaches range from methods specifically designed for a single problem, to local search methods, and so-called general-purpose algorithms such as meta-heuristics, memetic algorithms, and hyper-heuristics. The intrinsic nature of all these algorithms is that they usually expose a variety of different parameters which one can tamper with. Adherent to the NFL-theorem, no single algorithm is superior to all other algorithms without a proper adjustment of parameters, rendering the choice of suitable parameters a meaningful endeavour; this circumstance is known as the *algorithm configuration problem*.

This work proposes an approach, capable of configuring a modularized version of the most prominent meta-heuristic for single-objective continuous optimization, i.e., the *Covariance Matrix Adaption Evolution Strategy (CMA-ES)* [7]. Merits of this modular framework especially lie in its flexibility and timeliness. While it is based on the original CMA-ES (introduced by [1]), it also encompasses advances made in recent years. In other words, instead of using, for example, the conventional sampling strategy, the modular framework offers several other options such as 'mirrored sampling', 'orthogonal sampling', and 'quasi-random sampling'. These different options concerning a single component of the CMA-ES are named as so-called *modules*. In total, this framework comprises eleven modules which can be combined without any constraints. Whereas nine modules offer binary options (either 'active' or 'inactive'), the remaining two allow trinary parameter values. This results in $2^9 \cdot 3^2 = 4\,608$ different configurations. A full account of modules and their purposes is given in [7].

To evaluate the competitiveness of certain modules in general and specific configurations in particular, the modular CMA-ES framework with all candidate configurations is applied to a set of different problem instances. These problem instances are provided by the Black-Box Optimization Benchmark (BBOB) [2], where 480 different, complementary problem instances (of in total 24 functions) are selected from. This results in a dataset (called here *performance features)* comprising the *Expected Runtime (ERT)* for each configuration related to each problem instance. Whereas the performance differences between configurations for some problems are not particularly large, on others they are indeed. Furthermore, when only considering the best configuration per problem instance, not solely a small subset of distinct configurations remains but rather still around 400. Thus, identifying the optimal configuration of the modular CMA-ES framework can be considered crucial and auspicious.

The devised configuration approach of this work strives to find the optimal configuration for a given problem instance prior to optimization and therefore can be classified as offline per-instance configuration. Hence, it is not surprising that such an intelligent automated configuration procedure requires quantifiable and cheap information characterizing the problem instance a priori. Ideally, this information is provided automatically without requiring the involvement of domain experts. This is commonly referred to as *Exploratory Landscape Analysis (ELA)* [5] in the domain of continuous single-objective black-box optimization. These ELA features, based on a small initial sample of the search and objective space, offer insights into various properties of a problem instance, e.g. the degree of multi-modality, variable scaling, and global structure. Their relevance and usefulness has been validated in several works, e.g. [3].

The related performance and ELA features are pre-processed, resulting in a dataset which only consists of the single best configuration for each problem instance and thereby serves as training

data for the upcoming machine learning inquiry. Several methods of constructing algorithm selectors are discussed in [4]. Some of the derived insights can be transferred to this work concerning automated algorithm configuration; especially, the choice of establishing a 'classification' or 'regression' task. In context of a classification scenario, each possible configuration out of the 4 608 resembles its own class. Considering the disparity between possible classes to predict (4 608) and the number of training samples (480), the prerequisites of such a multi-class classification scenario are not met.

A more promising and straightforward approach is to predict whether a module should be 'active' or 'inactive' given certain ELA features. This transforms the multi-class classification with over four thousand classes into a multi-label classification scenario. In other words, when predicting the optimal configuration for a given problem instance, a multi-label classification model predicts the option values for each of the eleven modules, e.g. 'active' for the first module, 'inactive' for the second, and so forth. The implemented multi-label classifier builds upon the work of [6] introducing *Classifier Chains (CC)*. The constructed CC is constituted of a set of 'base' classifiers (here random forests), where each of these base classifiers is responsible for exactly one specific CMA-ES module. Yet, instead of being independent from each other, each base classifier takes the output of all previous classifiers (as well as the ELA features) into consideration. This enables the CC to capture interactions between modules. Therefore, the order of these base classifiers is crucial for the CC's performance.

The resulting CC is able to close the performance gap – between the single best configuration for all problem instances and a hypothetical situation where always the best configuration is chosen for each problem instance – by over 80% (which also encompasses the additional costs imposed by the calculation of the ELA features). However, it should be noted that the full potential of the approach has not been exhausted yet. Rather, the current state serves as a proof-of concept for further endeavours in that direction. At the time of writing, the high performance cluster of the University of Muenster is utilized to evaluate several techniques regarding feature and model selection of base classifiers as well as hyper-parameter tuning of the base classifiers considered. It is expected that this inquiry yields even better results. Moreover, the scenario will be alternatively treated as a regression rather than a classification problem, where CMA-ES performance of all configurations for a given problem instance is predicted allowing for finally selecting the most promising configuration.

The interested reader might argue that the presented undertaking so far does not truly resemble an algorithm configuration task but rather an algorithm selection effort with 4 608 different distinct algorithms. Whether one considers this problem as one of the former or latter is a topic of debate. Certainly, the configuration space of the modular CMA-ES framework is significantly smaller than in other algorithm configuration scenarios. Simultaneously, this modular CMA-ES framework comprises many more 'algorithms' than commonly considered in algorithm selection inquiries. Essentially, it opens up the question what constitutes and limits an algorithm.

## References

1. Nikolaus Hansen. *The CMA Evolution Strategy: A Comparing Review*, pages 75–102. Springer Berlin Heidelberg, 2006.
2. Nikolaus Hansen, Steffen Finck, Raymond Ros, and Anne Auger. Real-Parameter Black-Box Optimization Benchmarking 2009: Noiseless Functions Definitions. Research Report RR-6829, INRIA, 2009.
3. Pascal Kerschke and Heike Trautmann. Automated Algorithm Selection on Continuous Black-Box Problems by Combining Exploratory Landscape Analysis and Machine Learning. *Evolutionary Computation (ECJ)*, 27(1):99–127, 2019.
4. Lars Kotthoff. *Algorithm Selection for Combinatorial Search Problems: A Survey*, pages 149–190. Springer International Publishing, 2016.
5. Olaf Mersmann, Bernd Bischl, Heike Trautmann, Mike Preuss, Claus Weihs, and Günter Rudolph. Exploratory Landscape Analysis. In *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation*, GECCO '11, pages 829–836, New York, NY, USA, 2011. ACM.
6. Jesse Read, Bernhard Pfahringer, Geoff Holmes, and Eibe Frank. Classifier Chains for Multi-Label Classification. *Machine Learning*, 85(3):333–359, 2011.
7. Sanders van Rijn, Hao Wang, Matthijs van Leeuwen, and Thomas Bäck. Evolving the Structure of Evolution Strategies. In *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–8, Dec 2016.

# Stochastic Joint Replenishment Planning Optimization with an Iterated Local Search

Rabin Sahu[1,2] and Nadarajen Veerapen[2]

[1] *Research Lab, Vekia, Euratechnologies-Bâtiment Urbawood*
*58 Allée Marie-Thérèse Vicot-Lhermitte, F-59000 Lille, France*
[2] *Univ. Lille, CNRS, Centrale Lille, UMR 9189 - CRIStAL - Centre de Recherche en Informatique Signal et Automatique de Lille, F-59000 Lille, France*
*{rsahu@vekia.fr, nadarajen.veerapen@univ-lille.fr}*

## 1. Context and Motivations

In this extended abstract, we discuss optimized operations of supply chains dealing with multiple items. Examples of such supply chains are omni-channel retailers, supermarkets and service supply chains, etc. Whenever more than one item is ordered from one supplier or they share the same transportation vehicle, *first order interactions* take place between the items. First order interaction can be due to a joint (common) fixed cost in addition to individual fixed costs. When the demands are known with certainty the problem is a deterministic Joint Replenishment Problem (JRP). On the other hand, if the demands are considered stochastic, it is called the stochastic joint replenishment problem. Other costs involved are the inventory holding costs and the shortage costs.

The periodic deterministic JRP is strongly NP-hard for infinite horizon and NP-hard for finite horizon (Cohen-Hillel and Yedidsion, 2018). No such proof is available yet for the stochastic JRP. However, it is commonly regarded as very hard due to its huge solution space. Structural results for the stochastic JRP show that the optimal solution cannot be expressed as a simple replenishment policy even for the two-item case (Ignall, 1969). Therefore, many authors have assumed some simple structures such as $(s, c, S)$ (Johansen and Melchiors, 2003), $(s, c, d, S)$ (Feng et al., 2015) and Periodic Review$(s, S)$ (Viswanathan, 1997), to facilitate practical implementation ($S$ = *Order up to level;* $s$ = *Reorder level;* $c$, $d$ = *Can order levels*). Even if the adopted method follows one of the above policies, any change in problem parameters requires re-evaluation of policy parameters. Such a process is also time consuming.

The real-world applications of stochastic JRP are numerous. Usually, for the stochastic JRP, replenishment planning solutions use heuristics that are independent for each item under the JRP or follow one of the joint policies mentioned in the previous paragraph. These methods are not optimal for the global problem. Since the stochastic JRP are very hard to solve optimally, we propose to use a stochastic local search (SLS) approach to find good quality solutions in reasonable time. The ordering decisions are then implemented under a rolling horizon planning environment. The quality of a solution obtained using our proposed methodology is always better than what can be obtained using item level heuristics because, it uses the item level heuristics for generating the initial solution and then conducts further improvements. We also conduct numerical studies to attest its performance against the joint policies.

## 2. Methodology

In contrast to the general assumption in the literature, non-stationary demand is encountered widely during application. Hence, our methodology follows a dynamic ordering strategy, i.e. orders are calculated during each ordering epoch. We first generate the initial solutions using different heuristics that are independent for each item then, use an iterated local search for further improvement. The details are explained in the following sections.

### 2.1. Representation of the Solution

The stochastic JRP is a multi-period inventory optimization problem involving multiple items. At the beginning of each review period (ordering epoch), the inventory levels of all items are recorded and order quantities for all items are decided. Under dynamic ordering strategy, the optimal order quantities during an ordering epoch can be expressed as a vector $\mathbf{Q}$, where each element of the vector corresponds

to the order quantity of the respective item. The order quantities can only take zero or positive integer values. Therefore, a solution of the stochastic JRP is a set of dynamically decided multi-item orders. It is optimal if it *minimizes the total expected cost over a period* for non-stationary demand or it *minimizes the expected cost per period* for stationary demand. Our proposed methodology is applicable in the cases of both stationary and non-stationary demands. The proposed SLS approach is presented next.

## 2.2. Proposed SLS Approach

**Steps of an iterated local search (ILS) for the stochastic JRP**

1. *Initialization* – The proposed iterated local search uses either of the two following initializations: Random initial solution and initial solution obtained with a combination of single item heuristics.
2. *Neighborhood* – The second step involves a local search that uses three neighborhood operators: Positive/negative increment to one element, $k$-exchange and synchronized neighborhood operator (discussed later).
3. *Perturbation* – Multiple applications of the neighborhood operator to one or more randomly selected element(s).

The application of a neighborhood operator to a solution affects the order quantities for different items. The fitness of the new solution can be computed incrementally, and therefore is not time consuming. The performance of the above ILS approach depends significantly on the underlying neighborhood relation, and specifically, on the size of the neighborhood. For the stochastic JRP with $N$ items, the size of the neighborhood for a $k$-exchange operator is in $\mathcal{O}(N^k)$, and for the simple increment/decrement (for example +1/-1) the number of neighbors is $2N$.

**Synchronized Neighborhood Operator (SNO)**

The idea behind SNO is derived from the analysis of structural properties of near optimal solutions for the stochastic JRP. It is observed that the near optimal solutions tend to generate orders for a group of items during each ordering epoch. Under stochastic environment it is impossible to determine the exact time until which the inventories would last. However, the property of a near-optimal solution is to always ensure that the inventories for a maximum number of items last until equivalent times. In other words items are ordered in groups to reduced the share of joint ordering cost, and in such a way that they are also ordered in group in the future.

The SNO works on the above principle. For $N$ items there are at least $q_{min}^N$ possible solutions, where $q_{min}$ is the minimum of all possible number of orders for different items. The SNO provides a set of neighbors where a group of items has non-zero orders and the order quantities are synchronous to ensure that the next order is also grouped. Therefore, it eliminates the unnecessary searches in sub-optimal solution space.

## 2.3. Results

Preliminary results suggest further improvement of 2-7% over the best available joint policies in the literature for the two-item problem. More elaborate results will be presented during the workshop.

# References


Tamar Cohen-Hillel and Liron Yedidsion. The periodic joint replenishment problem is strongly NP-hard. *Mathematics of Operations Research*, 2018.

Haolin Feng, Qi Wu, Kumar Muthuraman, and Vinayak Deshpande. Replenishment policies for multi-product stochastic inventory systems with correlated demand and joint-replenishment costs. *Production and Operations Management*, 24(4):647–664, 2015.

Edward Ignall. Optimal continuous review policies for two product inventory systems with joint setup costs. *Management Science*, 15(5):278–283, 1969.

Søren Glud Johansen and Philip Melchiors. Can-order policy for the periodic-review joint replenishment problem. *Journal of the Operational Research Society*, 54(3):283–290, 2003.

S Viswanathan. Note. periodic review (s, S) policies for joint replenishment inventory systems. *Management Science*, 43(10):1447–1454, 1997.

# Automatic Design of Hybrid Stochastic Local Search Algorithms applied to the permutation flowshop problem

Federico Pagnozzi, Thomas Stützle
Université Libre de Bruxelles, IRIDIA
e-mail: federico.pagnozzi@ulb.ac.be, stuetzle@ulb.ac.be

Metaheuristics and, in particular, Stochastic Local Search, SLS, methods such as Iterated Local Search (ILS), Iterated Greedy (IG), Tabu Search (TS), GRASP or Simulated Annealing (SA) have been used with good results for many NP-hard optimization problems. Typically, when tackling such problems, an algorithm designer has to decide which one of these methods to apply and how to adapt the chosen metaheuristic to the problem. So far, this process has been carried out manually using a trial and error approach that requires time and an expert designer in order to achieve good results.

An alternative to the manual algorithm engineering is the automated design of effective SLS algorithms through building flexible algorithm frameworks and using automatic algorithm configuration techniques to instantiate high-performing algorithms.

The concept of automatic algorithm design can be traced back to the introduction of automatic configuration tools and unified algorithm implementations. The former simplifies the configuration of algorithms with a big parameter set, while, the latter expose the design choices, when building a SLS, as parameters. By using the two together, we can automatize the design of specific SLS in a process called programming by optimization. We propose a way to adapt these ideas towards generating high-performing algorithms for important scheduling problems. The method is based on decomposing the SLS algorithms to components and to define a set of rules to describe how to combine them. Finally, an automatic configurator is used to find the best combination of components that satisfies the given rules. The presented system can choose either to instantiate an existing SLS method or to create a new one by hybridizing two, or more, SLS algorithms. More specifically, the automatic configurator is used to select the best components, the rules are expressed as a grammar, and a new framework, called EMILI, has been created to implement the components and to instantiate the algorithms.

EMILI has been designed to be an unified framework for the automatic SLS design. EMILI is based on (i) a decomposition of SLS algorithms into algorithmic components, (ii) an algorithm template from which many different types of SLS methods can be instantiated, (iii) a recursive definition of possible algorithm compositions that in turn allow to generate hybrid algorithms, and (iv) a strict separation between algorithm-related components and

problem related components. EMILI is a significant refinement over previous proposals in terms of ease of implementation and algorithm composition, the comprehensiveness of the implemented components, and the possibility of tackling problem classes rather than single optimization problems.

# Fast evolutionary algorithm for multi-objective industrial problems

A. Ouskova Leonteva, P. Parrend, A. Jeannin-Girardon and P. Collet

ICube CSTB, 67000 Strasbourg
anna.ouskova-leonteva@etu.unistra.fr
pierre.parrend@unistra.fr
anne.jeannin@unistra.fr
pierre.collet@unistra.fr

**Abstract.** This paper proposes fast and smart evolutionary algorithm to solve large scaled industrial multi-objective optimization problems (MOPs) with high accuracy in continuous search space.

## 1 Introduction

The main problem of many industrial systems is an impossibility to determine their optimal characteristics of components by analytical methods and real-life experimental tests due to their high cost and complexity. A possible approach is to appay multi-objective evolutionary algorithms (MOEAs). Many existing state-of-the-art MOEAs do not satisfy necessary requirements (runtime (RT), accuracy) for resolving such kind of problems. Proposed MOEA provides high accuracy in short RT due to its low complexity and needs small number of estimations of function evaluations in order to approximate the set of Pareto optimal solutions.

## 2 Proposed approach

Inspired by an archive-based MOEA ASREA [1], the proposed algorithm, called FastEMO, has low complexity $O(man)$ (where $m$ is the number of objectives, $a$ is the size of archive, $n$ is the population size). As ASREA, FastEMO reduces the overall RT due to computation of a small Pareto Front, limited to the size of the archive. The optimal archive size is fixed as $15m$. Unlike ASREA, FastEMO has a much simpler architecture and suggests :

- Elitist archive update strategy, which retains only non-dominated solutions of a current generation to the next generation ;
- New adaptive crossover operator, which ensures continuously-increasing accuracy during generations. It is based on properties of BLX-$\alpha-\beta$ [2] and Arithmetic crossovers [3]. For adaptive control of parameter values for different generations, we propose a dynamic function, which exploits a number of the current generation and a number of non-dominated solutions in the previous population ;
- Self-adaptive Gaussian mutation one-step mechanism, which works efficiently on a large dimensional space and is described here [4] ;
- Increase of archive size to offspring population size on the last generation, in order to obtain more non-dominated solutions and to raise accuracy.

## 3 Experiments

Set of experiments are conducted to investigate the performance of FastEMO. During these experiments were used following resources :

- MOEAs for comparative study versus FastEMO : MOEA-D, NSGA2, NSGA3, IBEA, ASREA and CDAS ;

– Performance metrics of diversity and convergence : Hypervolume (HV) and Inverted Generational Distance (IGD) ;
– Benchmark suites (2/3-objective) : ZDT, DTLZ, WFG, UF, BBOB-biobj 2018 COCO (Comparing Continuous Optimizers)[5].

In order to estimate performance in BBOB COCO tests the following method is used : the central performance measure is the RT in terms of the number of evaluations conducted on a given problem until a given target value is hit (HV). For other benchmarks suites, total CPU time execution for given number of generations and mean value of RT per one generation is used. Evaluation of the test results was also carried out by analysing graphs of Pareto fronts.

Results of comparative experiments of FastEMO on fifty-five BBOB-biobj functions (with the search space dimension = 40) versus the referenced algorithm (RA) in COCO platform are shown in Table 1.

**Table 1.** Results of comparative experiments on fifty-five BBOB-biobj functions

| Number of results | Precision | | |
|---|---|---|---|
| | $1e-01$ | $1e-03$ | $1e-05$ |
| Positive | 48 | 43 | 53 |
| Negative | 7 | 12 | 2 |

Results of comparative experiments of FasteEMO on ZDT, DTLZ, WFG, UF tests (31 functions, 1000 individuals, 500 generations) versus MOEAD-D, NSGA2, NSGA3, IBEA, ASREA, CDAS are shown Table 2.

**Table 2.** Results of comparative experiments on ZDT, DTLZ, WFG, UF(thirty-one functions)

| Number of results | CPU RT | HV | IGD |
|---|---|---|---|
| Positive | 31 | 28 | 29 |
| Negative | 0 | 3 | 2 |

## 4 Conclusion

According to experimental results, FastEMO has advantages over mainstream state-of-the-art algorithms in values of performance metrics and in CPU RT. Also FastEMO shows better values of RT and HV precision versus RA of COCO on the separable, moderate and illconditions functions on high search space dimension. During analyse of properties of the algorithm it was revealed, that with fixed number of evaluations and with an increase of the population size, values of performance metrics are improving with a slight increase of CPU RT due to properties of FastEMO: low complexity, simplified architecture and efficiency of genetic operators. By these properties, FastEMO can be easily implemented on GPU.

## References

1. Sharma, D., Collet, P. (2010, July). An archived-based stochastic ranking evolutionary algorithm (ASREA) for multi-objective optimization. In Proceedings of the 12th annual conference on Genetic and evolutionary computation (pp. 479-486). ACM.
2. Herrera, F., Lozano, M., Snchez, A. M. (2003). A taxonomy for the crossover operator for realcoded genetic algorithms: An experimental study. International Journal of Intelligent Systems, 18(3), 309-338.
3. Furqan, M., Hartono, H., Ongko, E., Ikhsan, M. (2017). Performance of Arithmetic Crossover and Heuristic Crossover in Genetic Algorithm Based on Alpha Parameter. IOSR Journal of Computer Engineering (IOSR-JCE), 19(1), 31-36.
4. Back, T. (1992). Self-adaptation in genetic algorithms. In Proceedings of the first european conference on artificial life (pp. 263-271). MIT Press.
5. COCO: The Bi-objective Black-Box Optimization Benchmarcking Test Suite. https://numbbo.github.io/coco-doc/bbob-boobj/functions

# Automatic Configuration of a Dynamic Hill Climbing Algorithm

Weerapan Sae-dan[1] , Marie-Eléonore Kessaci[1],Nadarajen Veerapen[1], and Laetitia Jourdan[1]

Université de Lille, CNRS, UMR 9189 - CRIStAL 59000 Lille, France
{weerapan.saedan,mkessaci,nadarajen.veerapen,laetitia.jourdan}@univ-lille.fr

## 1 Introduction

In this paper, we propose an automatic algorithm configuration (AAC). It can manage many parameters and an online method produces a deterministic algorithm which parameters modified during execution. We explore the benefits of both methods by proposing a dynamic framework that switches between different configurations during execution and adapted AAC protocol. We define a control mechanism for parameters as a combination of three questions [1]: what is to be controlled, how the control is performed, and when the parameter is changed? Our method is tested on single-objective permutation scheduling flowshop problem with an automatic configuration of a dynamic hill-climbing.

## 2 Automatic Algorithm Configuration of Dynamic Framework

Our contribution is to propose a new method to parameterize a dynamic algorithm. To describe what is called a dynamic framework, C. Pageau et al. [2] propose using parameter configuration that changes at predetermined times, as show in figure 1. We will propose more flexibility on the execution times for each configuration. This concept enables the use of AAC instead of the parameter control mechanisms.

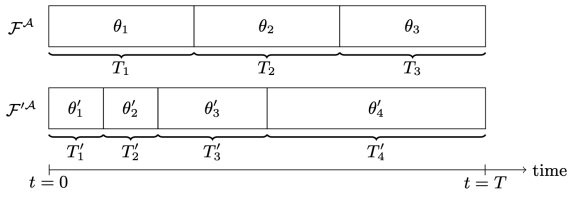



Fig. 1: Example of two configuration schedules [2]

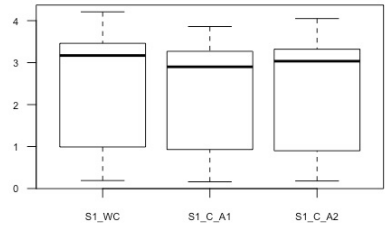
In this paper, the automatic configuration can fix the duration of each time split ($s_i$) in the list $\{10, 25, 50, 75, 90\}$ and each time split is calculated as follows. $T_1$ is equal to $(T \times s_1)/100$, $T_2$ is equal to $((T - T_1) \times s_2)/100$, $T_i$ is equal to $((T - \sum(T_i - T_{i-1})) \times s_i)/100$, and $T_k$ is the remaining budget where $T$ is the time budget.

## 3 Experiments and Conclusion

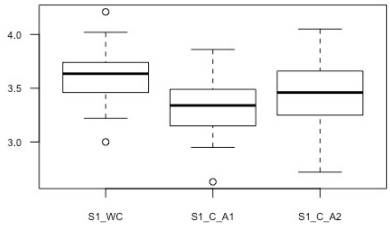
We investigated the proposed method to solve the single-objective permutation flowshop scheduling problem. We use Irace [3] to implement the AAC and to find the configuration of hill-climbing [4] and its components (exploration neighborhood, operation neighborhood, neighborhood order, and perturbation) best adapted to the instances of the problem to solve. We use the Taillard instances [5], we use different instance of N jobs, M machines: 20×5, 20×10, 20×20, 50×5, 50×10, 50×20, 100×5, 100×10, 100×20, 200×10, and 200×20. For each size, 10 instances are used.

In this paper, two dynamic framework parameters have been implemented: $K$ as the number of time splits and $\{T_1, T_2, \ldots, T_K\}$ the associated time budget. 3 time budget configurations are

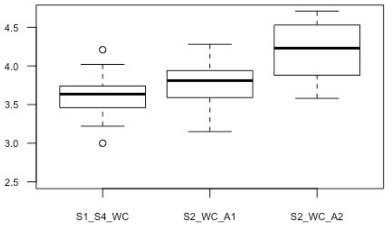
possible, corresponding to the different time splits $\{T\}$, $\{T_1, T_2\}$ and $\{T_1, T_2, T_3\}$. 24 configurations for our algorithm are present per time split. We test 3 scenarios in which $K$ is equal to 1, 2 or 3. Since $|s| = |\{10,25,50,75,90\}| = 5$, the number of time configurations possible are 1, 5 and 25 for $K = 1$, 2 and 3 respectively. If $K = 1$, then there are 24 different configurations, $K = 2$ includes approximately $2.9 \times 10^3$ different configurations $(24+5\times24^2)$ and, $K = 3$ a total of approximately $3.5 \times 10^5$ different configurations $(24 + 5 \times 24^2 + 25 \times 24^3)$. We set to $N^2 \times M$ milliseconds, the stopping criterion of the algorithm.



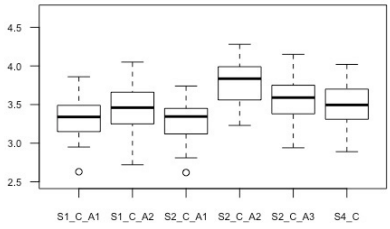

(a) all instances




(b) without and with control




(c) without parameter control




(d) with parameter control

Fig. 2: Instances 100 jobs and 20 machines. C-with control, WC-without control,$S_1$ = all instances, $S_{2-4}$-subset of instances, $A_{1-4}$-specific parameter configuration. Smaller values are better.

For the comparison between with control and without control, the experiments presented in figure 2 (a)(b) show that the value of control. Moreover, the experiments presented in (c)(d) show that it is more interesting without control to learn from all instances while with control performance is better with learning on instances of the same size. We hypothesize that parameter control makes the algorithm better adapted to the local properties of the instance.

## References

1. A.E. Eiben, R. Hinterding, and Z. Michalewicz. Parameter control in evolutionary algorithms.3(2):124141,1999.
2. Camille Pageau, Aymeric Blot, Holger Hoos, Marie-Elonore Kessaci, and Laetitia Jourdan. Configuration of a Dynamic MOLS Algorithm for Bi-objective Flowshop Scheduling: 10th In- ternational Conference, EMO 2019, East Lansing, MI, USA, March 10-13, 2019, Proceedings, pages 565577. 01 2019.
3. L*ó*pez-Ib*áñ*ez, M., Dubois-Lacoste, J., C*á*ceres, L. P., Birattari, M., and St*ü*tzle, T. The irace package: Iterated racing for automatic algorithm configuration. Operations Research Perspectives, 3:43 58, 2016.
4. Cohen, W., Greiner, R., Schuurmans, D. Probabilistic Hill-Climbing. Computational Learning Theory and Natural Learning Systems, Vol. II. Edited by Hanson, S., Petsche, T., Rivest R., Kearns M. MITCogNet, Boston:1994.
5. Taillard, E. Benchmarks for basic scheduling problems. European Journal of Operational Research, 64(2):278285,1993.

# Sampled Walk: A Partial Neighborhood Search Strategy

Sara Tari, Matthieu Basseur, Adrien Goëffon

Laboratoire d'Etude et de Recherche en Informatique d'Angers, LERIA, EA 2645, SFR MathSTIC, UNIV Angers. 2 boulevard Lavoisier 49045 Angers cedex 01 FRANCE
{sara.tari, matthieu.basseur, adrien.goeffon}@univ-angers.fr

## 1 Introduction

This work aims to obtain insights to conceive local search algorithms. We focus on establishing links between optimization problem structure and efficiency of local searches rather than tackling and optimizing a specific problem.

More precisely, we present and focus on *partial neighborhood local search* algorithms, simple solution-based local searches which explore a sample of the neighborhood at each step of the search. In our experiments, we perform a parameter sensitivity analysis on partial neighborhood local searches and compare them to state-of-the-art local searches on several types of binary and permutation fitness landscapes.

Fitness landscapes, first introduced by Wright [2] in the field of theoretical biology, are used in various fields to apprehend the behavior of complex systems better. In particular, in evolutionary computation, the study of combinatorial and continuous search spaces through fitness landscapes analysis helps to understand and predict the behavior of evolutionary algorithms. Such a model can help to observe difficulties induced by a given problem when tackled with an optimization method. Focusing on such landscapes facilitates the extraction of links between their properties and search algorithms behavior. Our experimental analysis highlights some links between ruggedness and both overall efficiency of considered methods as well as parameter sensitivity of partial neighborhood local searches.

## 2 Sampled Walk

The key concept of Sampled Walk (`SW`) is to ignore if a move improves or not the current fitness value while maintaining a selection pressure. `SW` is a local search based on randomly sampled neighborhoods. At each step of the search, `SW` evaluates $\lambda_{\texttt{SW}}$ random neighbors of the current solution and selects the one with the highest fitness value. Except $\lambda_{\texttt{SW}}$, the only conceptual choice to make concerns the stopping criterion. $\lambda_{\texttt{SW}} = 1$ corresponds to a random walk whereas $\lambda_{\texttt{SW}} = N$ ($N$ being the neighborhood size) corresponds to a tabu search mechanism with an empty tabu list.

Due to the extreme simplicity of `SW`, its implementation is easy and does not require many design choices which depend upon the considered neighborhood function. Moreover, `SW` simplicity greatly facilitates its analysis and allows many specific advanced variants. Note that `SW`, which is defined in a local search context, can also be viewed as a (1,$\lambda$) evolution strategy (with $\lambda = \lambda_{\texttt{SW}}$).

`ID-Walk` (Intensification/Diversification Walk) [1] is based upon a similar concept. Like `SW`, `ID-Walk` can be considered as a *partial neighborhood search* since it consists of evaluating (at most) $\lambda_{ID}$ solutions at each step of the search. However, `ID-Walk` selects the first encountered improving neighbor and therefore considers the fitness of the current solution to select the move to apply. When no improving solution is found among the $\lambda_{ID}$ neighbors, the selected one depends upon the considered variant. $\texttt{ID}_{best}$ selects the best one among the $\lambda_{ID}$ deteriorating neighbors, whereas $\texttt{ID}_{any}$ randomly selects one of them.

It is obvious that these partial neighborhood local searches (`SW`, $\texttt{ID}_{best}$, $\texttt{ID}_{any}$), which use randomly generated subneighborhoods, leads to similar behaviors. As stated by Neveu *et al.* [1], `ID-Walk` was proposed to combine intensification and diversification during the search process. Although `SW` follows the same principle, it emphasizes that the sign of the fitness variation does not determine explicitly the diversification aspect brought by the partial neighborhood. Moreover, `SW` does not consider the fitness of the current solution for the selection process. The experiments in the next section investigate empirically `SW`, and determine if it can be efficient on various landscapes.

## 3 Analysis on Fitness Landscapes

In order to accurately assess the capacity of partial neighborhood local searches to lead toward good quality solutions, we compared the three variants `SW`, $\texttt{ID}_{best}$ and $\texttt{ID}_{any}$ to two widely-used local searches: tabu search (TS) and iterated local search (ILS). Like `SW`, the classic tabu search does not use the current fitness for the selection process, but considers almost the whole neighborhood at each step of the search. ILS separates intensification and diversification phases. We choose here to use the first-improvement strategy during hill-climbing (intensification) phases, as first-improvement regularly reaches better local optima than best-improvement on landscapes difficult to climb.

ILS, TS, $\texttt{ID}_{best}$, $\texttt{ID}_{any}$ and `SW` require to set two parameters: a stopping criterion and their respective aforementioned parameter. For each method, we performed runs using several parameter values in order to establish appropriate settings. In our experiments, we consider four types of fitness landscapes: NK landscapes, UBQP landscapes, QAP landscapes, and FSP landscapes. The two first types use binary string solution representation, whereas the two lasts use permutations.

First, a parameter sensitivity analysis coupled with landscape analysis shows that best $\lambda_{\texttt{SW}}$ values correspond to a portion of the whole neighborhood varying from 3% and 15% of its size. These values tend to be higher on highly rugged landscapes and tends to be smaller for large instances. Moreover, this size slightly increases when the maximal number of evaluations is greater.

A comparative analysis of the results shows in most cases that on the considered NK landscapes, the sampled walk `SW` leads toward best solutions on average. `SW` efficiency does not seem to be affected by the ruggedness of the landscapes. On every considered landscape, results obtained by `ID-Walk` are very close to those obtained by `SW`, but `SW` often statistically dominates on large landscapes.

On UBQP landscapes, results differ from those obtained on NK landscapes. On large UBQP landscapes, the tabu search often leads toward the best average fitness values. This fact might be resulting from a particular structure of these landscapes. Considering NK landscapes, analogies between the evolution of ruggedness indicators indicate that such landscapes have a uniform ruggedness repartition. On the contrary, it appears that UBQP landscapes have a less uniform ruggedness repartition, which we can described as a local ruggedness and a global smoothness. This specificity could also explain why smaller UBQP instances are easy to solve by local search as long as some diversification is applied.

On permutation problems (QAP and FSP), `SW` and $\texttt{ID}_{best}$ outperform other variants and are often statistically not comparable.

## 4 Conclusion

We have investigated partial neighborhood local searches and, more particularly, the sampled walk algorithm which can be viewed as a local search transposition of a (1,$\lambda$)-ES. We show that the sampled walk is efficient to tackle standard binary and permutation landscapes. Conducted experiments on NK landscapes highlighted the fact that the sampled walk behavioral parameter can be principally set according to the landscape ruggedness. Experiments also show that such a method is globally competitive in comparison to metaheuristics like tabu search and iterated local search. Even if a tabu search outperforms the sampled walk on UBQP, we can establish links between respective efficiency of methods and ruggedness repartition thanks to the $k$-ruggedness indicator.

This family of local searches based on a random sampling of the neighborhood are not deeply investigated in the metaheuristics literature despite their simplicity. There are thus many ways to use the sampled walk principle and to improve its efficiency, for instance, by adapting its parameter during the search according to landscapes features.

## References

1. Bertrand Neveu, Gilles Trombettoni, and Fred Glover. Id walk: A candidate list strategy with a simple diversification device. In *International conference on principles and practice of constraint programming*, pages 423–437. Springer, 2004.
2. Sewall Wright. The roles of mutation, inbreeding, crossbreeding, and selection in evolution. In *Proceedings of the Sixth International Congress on Genetics*, volume 1, pages 356–366, 1932.

# A Hybrid Heuristic for the Synchronized Team Orienteering Problem with Time Windows

Ala-Eddine Yahiaoui[1], Aziz Moukrim[1], Mehdi Serairi[1] and Marc Demange[2]

[1] Sorbonne universités, Université de technologie de Compiègne, CNRS, Heudiasyc UMR 7253, CS 60 319, 60 203 Compiègne cedex
{ala-eddine.yahiaoui,aziz.moukrim,mehdi.serairi}@hds.utc.fr
[2] School of Science, RMIT University, Melbourne, Australia
marc.demange@rmit.edu.au

**Keywords.** Team Orienteering Problem, Time windows, Synchronization, Iterated Local Search, Set Cover.

## 1 Introduction

In this paper, we address a recent variant of the Team Orienteering Problem (TOP) where additional constraints, namely time windows and synchronized visits, are considered. This variant has been initially proposed in the context of planning protection tasks of a number of strategic assets (buildings, facilities, bridges, etc.) endangered by wildfire progression. This problem is modeled using a fleet of heterogeneous vehicles used to visit the set of assets. Each asset is associated with a time window representing fire fronts progression. An asset is also assigned a service time duration which models the time necessary to perform the protection activities. Finally, each asset has a resource requirements which are expressed by the number and type of vehicles required. Due to these constraints, protecting all the assets might be impossible. Hence, a value called profit is associated with each asset in order to distinguish between different assets according to their relative importance. In order to gain the profit of a given asset, it must be visited by the required vehicles in a synchronized manner, i.e. the visits should be simultaneously performed by the vehicles within the corresponding time window. As a result, the objective function aims at maximizing the amount of profit collected. We call this problem the Synchronized Team Orienteering Problem with Time Windows (STOPTW).
The STOPTW was first proposed in [1] under the name of Asset Protection Problem During Escaped Wildfire. The authors introduced a mixed integer programming model for the problem which was demonstrated on a realistic wildfire scenario in Tasmania. The authors in [2] proposed an Adaptive Large Neighborhood Search Heuristic (ALNS) for the problem along with a new set of benchmark instances.
To solve this problem, we propose in this paper an effective hybrid heuristic. Our method combines between local search heuristics and a set cover formulation in order to achieve higher performance. Experimental results conducted on benchmark instances [2] have shown that our method outperforms the ALNS in terms of both, solution quality and computational times.

## 2 Contribution

We propose in this paper an iterative heuristic to solve the STOPTW. This heuristic is composed of three major components.
The first component is an Adaptive Iterated Local Search (A-ILS) with an embedded *Destructive/ Constructive* local search. The construction phase is performed by using a *candidate-list* based insertion heuristic, whereas the destruction phase is carried out by an adaptive removal operator. The ordering of assets in the candidate list is calculated based on a specific criteria that takes into account the width of time windows, the number of vehicles and the profit. In addition, these factors are weighted using three parameters generated randomly during the search progress and used to calculate a new candidate list after each iteration. This feature allows the heuristic to cover a larger part of the search space.
Solutions produced by the A-ILS are improved by the second component that acts as an *Improvement Procedure* (IP). The IP invokes several efficient local search operators adapted

to the case of STOPTW. Four operators have been developed: *relocate*, *2-opt**, *exchange* as well as the *insertion* operator described earlier. Although *relocate* and *2-opt** operators do not increase the total profit, they are used to reduce the distance/ time of the solution and hence, provide eventual opportunity to the other operators to improve the solution quality. The two previous components generate throughout the solving process a set of good solutions which are saved in a pool. The third component of the heuristic is a Route Recombination (RR) procedure. It aims at finding the best combination of routes that maximizes the total objective value. This is achieved using a set covering formulation solved on the pool of routes. The solution obtained by RR is further improved by applying the local search operators of the IP component.

## 3 Experimentation and results

Computational tests have been carried out on 240 instances generated by Roozbeh et al. (2018) in [2]. We compared the proposed heuristic with the ALNS presented in [2]. Benchmark instances are all composed of 200 vertices in addition to the depot, and each instance was used to derive an additional instance of medium size by truncating the first 100 vertices. We followed the same protocol used in [2] to evaluate the ALNS, that is, ten runs for each instance, and we recorded the best solution as well as the average objective value.
Table 1 depicts the results of our method compared to ALNS. It reports computational times ($CPU$) in seconds, the best ($Best$) and the average run ($AVG$), both represented by the percentage of protected assets, and finally, the improvement achieved by our method compared to ALNS ($GAP$).

**Table 1.** Comparison with the literature

| Nb. Assets | ALNS | | | Our method | | | |
|---|---|---|---|---|---|---|---|
| | CPU(s) | Best (%) | AVG (%) | CPU(s) | Best (%) | AVG(%) | GAP(%) |
| 100 | 141.66 | 68.03 | 66.37 | 49.22 | 76.47 | 75.18 | −12.41 |
| 200 | 578.93 | 64.04 | 62.68 | 176.36 | 74.40 | 73.80 | −16.18 |

Regarding instances with 100 assets, our method succeeded to achieve substantial improvement compared to ALNS. Computational times were divided by a factor of 2.9, decreasing from 141.66$s$ for ALNS to only 49.22$s$. The overall percentage of protected assets also was substantially improved, rising from 68.03% to 76.47%, yielding an overall improvement gap of −10.41%. In the case of 200 assets, overall computational time achieved by our method are 3.3 times faster than that of ALNS, with 176.36$s$ against 578.93$s$. The overall percentage of saved assets was also improved which attains 74.40% against 64.04% realized by ALNS, yielding an improvement gap of −16.18%.

***Acknowledgements.*** The authors would like to thank the Hauts-de-France region and the European Regional Development Fund (ERDF) 2014/2020 for the funding of this work. This work was carried out in the framework of GEOSAFE (Geospatial Based Environment For Optimization Systems Addressing Fire Emergencies) and of Labex MS2T funded through the program "Investments for the Future" managed by the National Agency for Research (Reference ANR-11-IDEX-0004-02).

## References

1. van der Merwe, M., Minas, J. P., Ozlen, M., & Hearne, J. W. (2014). A mixed integer programming approach for asset protection during escaped wildfires. Canadian Journal of forest research, 45(4), 444-451.
2. Roozbeh, I., Ozlen, M., & Hearne, J. W. (2018). An adaptive large neighbourhood search for asset protection during escaped wildfires. Computers & Operations Research, 97, 125-134.

# Evolving Stochastic Hill Climbers

Vincent Hénaux, Adrien Goëffon, and Frédéric Saubion

Laboratoire d'Etude et de Recherche en Informatique d'Angers, LERIA, EA 2645,
SFR MathSTIC, UNIV Angers
{firstname.name}@univ-angers.fr

## 1 Seeking for alternative fitness functions

Given an optimization problem defined by an objective function $f_{\text{obj}} : \mathcal{X} \to \mathbb{R}$ on a search space $\mathcal{X}$ the purpose of Local Search (LS) algorithms [2] is to find a solution $x \in \mathcal{X}$ maximizing (or minimizing) the objective value. LS algorithms guide the search by means of a neighborhood relation, a fitness function and a move strategy. The design of an efficient algorithm often focuses on the move strategy, given a natural neighborhood relation and the objective function $f_{\text{obj}}$ as fitness function—that is, from a fitness landscape directly derived from the problem.

Among the most basic move strategies, the (first improvement) hill climber randomly selects improving neighbors until a local optimum is reached. The difficulty for an $f_{\text{obj}}$-based climber to find good solutions (and in particular the global optimum) is obviously correlated with the number of local optima induced by $f_{\text{obj}}$ and the neighborhood relation. Improving such algorithm to reach better solutions usually involves to refine the move strategy with diversification techniques.

On the contrary, we propose here to modify the fitness function so that a simple climber can achieve better solutions. Our aim is more precisely to iteratively improve the LS algorithm by changing the fitness landscape through mutations on the fitness function, while keeping unchanged the neighborhood relation and the move strategy.

Indeed, once the LS process has been set, there exist better fitness functions than $f_{\text{obj}}$ to reach more efficiently the global optimum, ie. fitness function whose associated fitness landscape is easier to climb from a random starting solution [1]. For instance, an optimal fitness function $f_{\text{opt}}$ could be $f_{\text{opt}}(x) = -d(x, x_{\text{opt}})$, where $d(\cdot, \cdot)$ is a distance and $x_{\text{opt}}$ the global optimum. Of course, finding an optimal fitness function is equally difficult that finding the global optimum, but we investigate in this paper to find better hill climbers than a climber guided by means of $f_{\text{obj}}$. We propose here a search process (in the space of fitness functions) which consists to improve fitness functions and thus hill climbers (operating in the original search space). The aim is to determine an alternative climber LS($f_{\text{alt}}$) that globally reaches better solutions (evaluated w.r.t. $f_{\text{obj}}$) than the natural climber LS($f_{\text{obj}}$).

## 2 Local search algorithm and preliminary experiments

Algorithm 1 presents the general strategy. An initial alternative fitness function $f_{\text{alt}}$ is randomly generated and then evolves by means of a mutation operator, describing a local search process (first improvement hill climbing). Moreover, the fitness function evaluation stage consists itself of iterating first improvement hill climbers for collecting local optima (w.r.t. $f_{\text{alt}}$). These solutions are then evaluated by using the objective function $f_{\text{obj}}$. Climbers using current and neighboring (muted) functions are compared by means of a Mann–Whitney U test. A muted function is selected when the corresponding hill climbing algorithm statistically outperforms the current one.

In this work we use NK landscapes [3] as binary combinatorial optimization problems to get a proof of concept. As first experiments we choose NK functions both as objective functions and as alternative fitness functions. The aim is then to determine an NK function able to climb the landscape at least as efficiently as the original one. A NK landscape mutation is obtained by randomly modifying variable dependencies and/or fitness contributions.

Let us precise that we do not currently focus on the computational cost of such a strategy, but we rather validate the concept of navigating through the search space by following the adaptive paths of an alternative landscape, itself constructed in an adaptive way. One can observe that this evolutionary process is able to build a fitness function which is as equally adapted or even more that the original fitness functions. This proof of concept will serve as a basis for future work about the adaptive construction of local search algorithms.

**Algorithm 1** Fitness function evolution

**Parameters**: a fitness function model ($\mathcal{M}$), a mutation parameter ($\zeta$), a solution space ($\Omega$), a neighborhood function ($\mathcal{N}$), a statistical test ($\mathcal{T}$), a statistical significance threshold ($z_{\min}$), the number of local optima hill-climbing runs for evaluate fitness functions ($p$), the maximum number of consecutive non-improving mutations (maxmut).
**Input**: an objective function $f_{\text{obj}}$.
**Output**: an alternative fitness function $f_{\text{alt}}$.

$f_{\text{alt}} \leftarrow \text{initialize}_{\mathcal{M}}()$
**repeat**
  **for** $j \leftarrow 1$ to $p$ **do**
    randomly select $s \in \Omega$
    $S_j \leftarrow \text{hill\_climbing}_{(\Omega,\mathcal{N},f_{\text{alt}})}(s)$ // *$S$ is a vector of local optima w.r.t. $f_{\text{alt}}$*
    $F_j \leftarrow f_{\text{obj}}(S_j)$ // *$F$ is a vector of scores*
  **end for**
  nbmut $\leftarrow 0$
  **repeat**
    improve $\leftarrow$ false
    $f'_{\text{alt}} \leftarrow \text{mutate}_{\mathcal{M},\zeta}(f_{\text{alt}})$
    **for** $j \leftarrow 1$ to $p$ **do**
      randomly select $s \in \Omega$
      $S'_j \leftarrow \text{hill\_climbing}_{(\Omega,\mathcal{N},f'_{\text{alt}})}(s)$
      $F'_j \leftarrow f_{\text{obj}}(S'_j)$
    **end for**
    **if** $\text{z\_score}_{\mathcal{T}}(F', F) \geqslant z_{\min}$ **then** // *significant score of hypothesis* $\text{LS}(f'_i) \succ \text{LS}(f)$
      $f_{\text{alt}} \leftarrow f'_{\text{alt}}$
      improve $\leftarrow$ true
      nbmut $\leftarrow 0$
    **else**
      nbmut $\leftarrow$ nbmut $+ 1$
    **end if**
  **until** improve or (nbmut = maxmut)
**until** not improve
**return** $f_{\text{alt}}$

| | Avg HC score | |
|---|---|---|
| $f_{\text{obj}}$ | $f_{\text{obj}}$ based | $f_{\text{alt}}$ based |
| $\text{NK}_{256,1}$ | 0.691 | 0.695 |
| $\text{NK}_{256,6}$ | 0.725 | 0.719 |
| $\text{NK}_{256,12}$ | 0.705 | 0.702 |

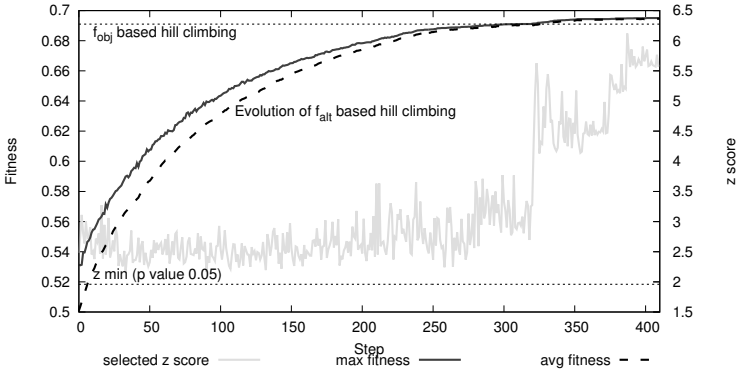



**Fig. 1.** Performance of $f_{\text{alt}}$ based hill climbers. Reference $f_{\text{obj}}$ are NK functions with $N = 256$ and $K \in \{1, 6, 12\}$. $f_{\text{alt}}$ are evolved NK functions with $N = 256$ and $K = 1$. The figure shows the evolution of the $f_{\text{alt}}$ score considering the first $f_{\text{obj}}$ function ($N = 256, K = 1$)

## References

1. Matthieu Basseur and Adrien Goëffon. Climbing combinatorial fitness landscapes. *Applied Soft Computing*, 30:688 – 704, 2015.
2. Holger H Hoos and Thomas Stützle. *Stochastic local search: Foundations and applications*. Elsevier, 2004.
3. Stuart A Kauffman and Edward D Weinberger. The NK model of rugged fitness landscapes and its application to maturation of the immune response. *Journal of theoretical biology*, 141(2):211–245, 1989.

# A learning-based algorithm for continues optimization

Hojjat Rakhshani, Lhassane Idoumghar, Julien Lepagnot, and Mathieu Brévilliers

Université de Haute-Alsace, IRIMAS-UHA, F-68093 Mulhouse, France

## 1 Introduction

In this work, we investigate the use of a deep learning (DL) model as an alternative tool to tackle the optimization tasks. Feature extraction is the main adopted concepts in MaNet that helps the algorithm to skip irrelevant or partially relevant parameters and use those design variables which contribute most to the overall performance in each iteration. The proposed MaNet adopts a Convolutional Neural Network (CNN); which are regularized version of fully-connected neural networks inspired from biological visual systems [1]. In MaNet, multiple convolution layers are stacked which allows convolution layers to be applied to the output of the previous layer, results in a hierarchically set of more decomposed features. Finally, a Dense layer (or fully-connected) with linear activation function will be used to form the final solution vector. As it can be seen from Fig. 1, MaNet has a very simple structure and can benefit from the advantage of having a fast network training process[1]. Indeed, it has only 3,742 trainable parameters compared to state-of-the-art models [2] which have millions or billions of parameters. This could facilitate the application of MaNet for optimization tasks where a small amount of data (i.e., population) is available.

MaNet is composed of two similar architectures which are subjected to different optimization procedures. The first one uses a batch size of one and the other uses 64 as its batch size. The batch size is a hyperparameter of gradient descent that should be tuned for each optimization task. To do so, MaNet integrates a reinforcement strategy inspired from SDCS [3]. Technically speaking, SDCS is a simple metaheuristic algorithm which toggles continually between two snap and drift modes to enhance reinforcement and stability. Based on this idea, MaNet introduces a self-adaptive strategy to tune the batch size hyperparameter. More precisely, it is looking to see if the best cost function stops improving after some number of epochs, and if so then it restarts the optimization process and continues the search by the architecture which obtained a higher overall performance so far. Finally, it is worth mentioning to note that the initial population will remain unchanged during training the network and the algorithm will evolve a set of filters. The goal of MaNet then, is to transfer the initial population on one end to evolved solutions on the other hand. This is one of the main differences between MaNet and evolutionary algorithms.

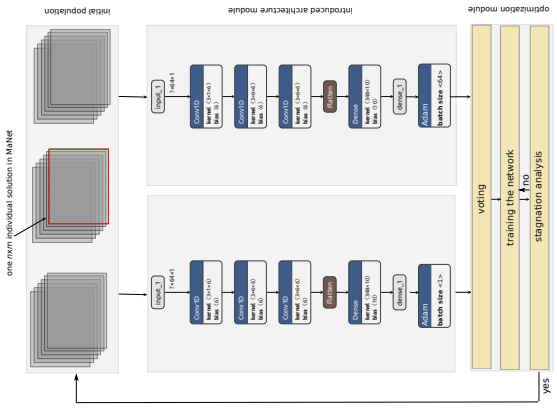



Fig. 1: An overview of the proposed optimization architecture

[1] Netron Visualizer is used to illustrate the model. The tools is available online at: https://github.com/lutzroeder/netron

### 1.1 Experimental results

We use a set of 9 benchmark functions given in CEC 2017 [20] to evaluate the performance of the proposed algorithm[2]. The aforementioned problems are adopted on the GPU so as to be linked with machine learning libraries. We used jSO [4] algorithm for the purpose of comparison which is the second ranked algorithm in CEC2017 competitions for the single objective optimization track. In order to make a fair comparison, all the experiment conditions are the same. In MaNet, we have 3 convolution layers which are sequentially connected to each other. In each layer, the number of filters and the filter size are 6 and 3, respectively. The MaNet is a CNN model and needs a lot of input data to be well trained and so the population size is fixed to $n = 5,000$. Moreover, $m$ is considered to be 64 for all the problems. The MaNet will be optimized using the Adam algorithm [5]. The results in Table 1 confirm that MaNet has a competitive results in comparison with one of the best designed algorithm for the CEC2017 problems. This is quite interesting because MaNet doesn't borrow any search strategy or component from the previously proposed methods for the CEC problems including L-SHADE [6] and jSO.

Table 1: The obtained results by MaNet and jSO for 50 dimensional problems over 51 runs [7]. The results of the Wilcoxon rank sum test are also reported at the 95% confidence level. The results for jSO are directly taken from the original paper [4].

| Function | Algorithm | Best | Worst | Mean | Median | Std. | Sign |
|---|---|---|---|---|---|---|---|
| 1 | MaNet | $3.67e+02$ | $2.06e+03$ | $1.39e+03$ | $1.46e+03$ | $3.71e+02$ | − |
| | jSO | $0.00e+00$ | $0.00e+00$ | **0.00e+00** | $0.00e+00$ | $0.00e+00$ | |
| 3 | MaNet | $9.80e+04$ | $1.42e+05$ | $1.23e+05$ | $1.25e+05$ | $8.88e+03$ | − |
| | jSO | $0.00e+00$ | $0.00e+00$ | **0.00e+00** | $0.00e+00$ | $0.00e+00$ | |
| 4 | MaNet | $3.10e-06$ | $1.53e-03$ | **8.22e-04** | $9.96e-04$ | $4.46e-04$ | + |
| | jSO | $1.32e-04$ | $1.42e+02$ | $5.62e+01$ | $2.85e+01$ | $4.88e+01$ | |
| 5 | MaNet | $1.99e+00$ | $1.09e+01$ | **6.15e+00** | $5.97e+00$ | $2.20e+00$ | + |
| | jSO | $8.96e+00$ | $2.39e+01$ | $1.64e+01$ | $1.62e+01$ | $3.46e+00$ | |
| 6 | MaNet | $0.00e+00$ | $0.00e+00$ | **0.00e+00** | $0.00e+00$ | $0.00e+00$ | = |
| | jSO | $0.00e+00$ | $0.00e+00$ | **0.00e+00** | $0.00e+00$ | $0.00e+00$ | |
| 7 | MaNet | $5.49e+01$ | $5.65e+01$ | **5.58e+01** | $5.59e+01$ | $3.62e-01$ | + |
| | jSO | $5.75e+01$ | $7.42e+01$ | $6.65e+01$ | $6.66e+01$ | $3.47e+00$ | |
| 8 | MaNet | $1.99e+00$ | $8.95e+00$ | **5.41e+00** | $5.97e+00$ | $1.99e+00$ | + |
| | jSO | $9.95e+00$ | $2.41e+01$ | $1.70e+01$ | $1.70e+01$ | $3.14e+00$ | |
| 9 | MaNet | $0.00e+00$ | $0.00e+00$ | **0.00e+00** | $0.00e+00$ | $0.00e+00$ | = |
| | jSO | $0.00e+00$ | $0.00e+00$ | **0.00e+00** | $0.00e+00$ | $0.00e+00$ | |
| 10 | MaNet | $1.86e+04$ | $1.88e+04$ | $1.87e+04$ | $1.87e+04$ | $6.25e+01$ | − |
| | jSO | $2.40e+03$ | $3.79e+03$ | **3.14e+03** | $3.23e+03$ | $3.67e+02$ | |

## References


1. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems. (2012) 1097–1105
2. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
3. Rakhshani, H., Rahati, A.: Snap-drift cuckoo search: A novel cuckoo search optimization algorithm. Applied Soft Computing **52** (2017) 771–794
4. Brest, J., Maučec, M.S., Bošković, B.: Single objective real-parameter optimization: Algorithm jso. In: 2017 IEEE congress on evolutionary computation (CEC), IEEE (2017) 1311–1318
5. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
6. Tanabe, R., Fukunaga, A.S.: Improving the search performance of shade using linear population size reduction. In: 2014 IEEE congress on evolutionary computation (CEC), IEEE (2014) 1658–1665
7. Awad, N., Ali, M., Liang, J., Qu, B., Suganthan, P.: Problem definitions and evaluation criteria for the cec 2017 special session and competition on single objective real-parameter numerical optimization. Tech. Rep. (2016)


[2] The codes for CEC problems and the jSO algorithm are publicly available at: http://www.ntu.edu.sg/home/EPNSugan/index_files/CEC2017/CEC2017.htm

# PRM-based differential evolution for UAV path planning

S. Ghambari[1], L. Idoumghar[1], L. Jourdan[2], and J. Lepagnot[1]

[1] University of Haute-Alsace, IRIMAS, F-68093 Mulhouse, France
soheila.ghambari, lhassane.idoumghar, julien.lepagnot@uha.fr
[2] University of Lille, CRIStAL, UMR 9189, CNRS, Centrale Lille, France
laetitia.jourdan@univ-lille.fr

## 1 Introduction

Nowadays, unmanned aerial vehicles (UAVs) are ideally suited for many real-world applications such as reconnaissance and surveillance. One of the most concerning issues in this field is autonomous path planning, which refers to planning an optimal path from the start location to a desired destination while avoiding obstacles. Previous studies have presented a series of techniques to tackle the aforementioned problem based on different necessities such as collision avoidance, real-time planning, and safety maximization. They took hints from different research areas; like mathematics for classical graph-based and probabilistic approaches [1], physics for potential field algorithm [2], or computer science for artificial intelligence methods [3].

In recent years, general-purpose stochastic local search algorithms, also known as meta-heuristics, received most of the research effort in the intelligent UAVs community [3, 4]. They are important approaches thanks to: a) their flexibility to solve large-scale complex problems, b) applying different learning strategies to perform an effective search towards the global optimum, c) permitting to utilize kinematic constraints to the path, and d) employing for both single and multiple UAVs. However, in practice, when the available computational resources are limited, they are not always the best choice due to their slow rate of convergence. A majority of meta-heuristics use random solutions to initialize their population. In some cases in UAV path planning problem, using random initial solutions may result in not finding any feasible path after several iterations, especially in large complex environments. Hence, the performance of these algorithms can be improved considerably if initial solutions are feasible for such condition.

In this study, we show how incorporating some heuristic information into the meta-heuristics can ease their usage for the aforementioned problem. Without loss of generality, the differential evolution (DE) and probabilistic roadmap (PRM) methods are considered as two baselines, where the initial population of DE is fed by the PRM's heuristic information. More precisely, the PRM method as a classical approach is utilized to generate a set of feasible paths for setting up the initial population of the DE algorithm to improve its convergence performance. The basic idea behind PRM is to take admissible random points for constructing a graph of configuration space that capture the underlying topology of the free space [5]. Thereafter, it uses a local planner in order to link adjacent points in free space with edges, and then using the graph to plan pass through the space. In this way, the algorithm converges faster to a global best path which can reduce the computational time considerably. We suppose that the search space is known and a grid-based map is used to represent the environment. The buildings in the environment with different polygon shapes are denoted as static obstacles. In order to understand how these polygon shapes occupy the grid cells, polygon triangulation method is used to decompose a polygon area into a set of triangles with pairwise non intersecting interiors. Then, it checks whether a grid cell lies inside a triangle or not. The efficiency of the suggested algorithm is examined on a realistic urban scenario. Evaluations exhibited desirable performance of the proposed algorithm in terms of solution accuracy and convergence rate.

### 1.1 Experimental results

In order to assess the performance of the suggested method, a realistic urban scenario is considered. The selected environment is the map of university of *Haute-Alsace* which is located in a small region of the Mulhouse city in France. The map file is extracted from *OpenStreetMap*, defined by geographical coordinates in terms of latitude and longitude (see Fig. 1 (a-c)). The configuration

parameters of the algorithm can be divided into two categories: environment and algorithm parameters which are listed in Table 1. These parameters have a great impact on the performance of the algorithms. Hence, instead of using a trial-and-error approach to identify good values for them, we applied *irace* software package [6] to get very high-performing algorithmic variants. Finally, we performed a comparison between the presented algorithm, standard DE, and PRM method. All the experiments were executed over 15 independent runs which are reported in Table 2. The results are compared with each other in terms of the best, mean and standard deviation (S.D.) of the path length and computational time, respectively. As it was expected, the hybrid approach considerably performs better than the standard DE algorithm. Also, PRM outperforms the other competitors in terms of computational time. However, the proposed algorithm is able to find paths with shortest length in comparison with two other methods. Moreover, the obtained optimal path is displayed in Fig. 1 (d). Generally, the suggested approach illustrated promising results in finding a superior solution with proper accuracy and minimum computational time. Our future work mainly consists in employing other algorithms that are more appropriate in the presence of moving obstacles, and combinations of other motion planning with meta-heuristics in order to reduce the computational time.

**Table 1.** The setting parameters

| Environment | DE algorithm | PRM |
|---|---|---|
| Latitude = [47.7279242, 47.7332576] | No.population in [1, 100] | No.sample [100, 200, 300, 400] |
| Longitude= [7.3070263, 7.3156652] | Crossover Rate in [0.1, 1] | No.edges in [5, 10] |
| Grid space= (87 * 54) | Scaling Factor in [0.1, 2] | Max edge length in [10, 20, 30] |
| Start point= [x= 1, y= 1] | No.strategy in [1, 5] | - |
| Goal point= [x= 71, y= 47] | Max Iteration = 250 | - |
| Grid size= 1 | Max Run = 15 | Max Run = 15 |
| No.obstacle= 49 | - | - |

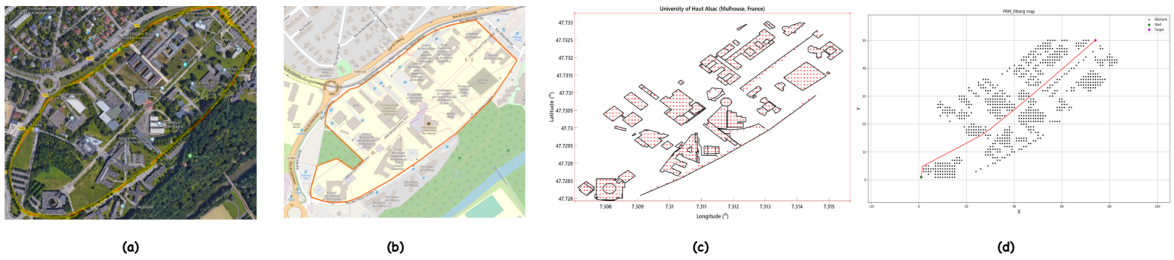



**Fig. 1.** A realistic urban scenario: (a) Google map picture, (b) Openstreetmap view, (c) Occupancy map (d) The obtained best path with the proposed method

**Table 2.** The obtained results of algorithms over 15 independent runs

| | Path length | Path length | Computational time (s) |
|---|---|---|---|
| Algorithm | Best | mean ± S.D. | mean ± S.D. |
| Proposed method | 7.97e+01 | 1.03e+02 ± 9.72e+00 | 3.28e+01 ± 1.73e+00 |
| DE | 1.74e+02 | 2.01e+02 ± 1.83e+02 | 6.65e+01 ± 3.81e+00 |
| PRM | 8.30e+01 | 1.10e+02 ± 1.05e+01 | 1.04e+00 ± 8.63e-02 |

## References

1. J. Li and X.-x. Sun, "A route planning's method for unmanned aerial vehicles based on improved a-star algorithm [j]," *Acta Armamentarii*, vol. 7, pp. 788–792, 2008.
2. Y.-b. Chen, G.-c. Luo, Y.-s. Mei, J.-q. Yu, and X.-l. Su, "Uav path planning using artificial potential field method updated by optimal control theory," *International Journal of Systems Science*, vol. 47, no. 6, pp. 1407–1420, 2016.
3. Y. Zhao, Z. Zheng, and Y. Liu, "Survey on computational-intelligence-based uav path planning," *Knowledge-Based Systems*, vol. 158, pp. 54–64, 2018.
4. S. Ghambari, J. Lepagnot, L. Jourdan, and L. Idoumghar, "A comparative study of meta-heuristic algorithms for solving uav path planning," in *2018 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, 2018, pp. 174–181.
5. L. Kavraki, P. Svestka, and M. H. Overmars, *Probabilistic roadmaps for path planning in high-dimensional configuration spaces*. Unknown Publisher, 1994, vol. 1994.
6. M. Lopez-Ibanez and T. Stutzle, "The automatic design of multiobjective ant colony optimization algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 16, no. 6, pp. 861–875, 2012.

# An engine for configurations of simulated annealing algorithm

İ. Sevim[1]

*1. Yıldız Teknik Üniversitesi, Beşiktaş Yerleşkesi, Barbaros Bulvarı, 34349*
*isevim@yildiz.edu.tr*

**Keywords**: automated configuration, simulated annealing, artificial neural networks.

## 1 Introduction

Simulated Annealing (SA) is a metaheuristic algorithm mimics the annealing process of solid materials [1]. It is based on the idea of heating and slowly cooling a solid material to have a durable crystal structure [2]. By following this analogy, SA allows the deterioration of fitness function value to escape from local optimality during the search process. At a given iteration, SA always accepts an improving solution, where it accepts a deteriorating solution with probability. This probability is based on two stochastic component: (1) Cooling schedule, and, (2) initial temperature. Performance of an SA implementation depends on these two components under the assumption of fixed initial solution.

In the literature, various studies discuss the correlation between features of problem instances and algorithmic performance [3,4]. Authors usually adopt the following approach for analysis: Run different algorithms for a set of instances of a particular problem and compare the results to choose the *best* algorithm. In this study, we analyse the same relationship following a different perspective: Instead of comparing different algorithms, compare the performances of different configurations of an algorithm on different instances of a particular problem. We work on an SA implementation for Travelling Salesman Problem (TSP) to illustrate the concept. The implementation follows the template of Al-Talbi [2]. Random datasets of TSP are obtained by using the approach given in [5].

In this work, we propose an artificial neural network (ANN) approach to learn the relationship between instance features and SA configurations (variations two stochastic components), and, algorithmic performance. We aim to work on adequately large number of TSP instances to train an ANN in an offline fashion and obtain an engine to predict the optimal configurations for a newly given TSP instance.

## 2 Method and Data Structure

Let **I** ={I1,I2,…,IN} be the set of TSP instances, **F**={F1,F2,…,FN} be the features of corresponding instances, and **C**={C1,C2,…,CM} be the set of configurations of SA. Then, the structure of the raw data is given in table 1 where f(N,M) is the fitness function value of the instance N with configuration M.

Table 1. Structure of raw data

| # | Instance | Configuration | Fitness Function Value |
|---|---|---|---|
| **1** | I1 | C1 | f(1,1) |
| **2** | I1 | C2 | f(1,2) |
| **…** | … | … | … |
| **M** | I1 | CM | f(1,M) |
| **…** | … | … | … |
| **…** | IN | C1 | f(N,1) |
| **…** | .. | … | … |
| **NM** | IN | CM | f(N,M) |

After obtaining the fitness function values for the instance and configuration pairs given in table 1, we are ready to build the dataset to train the ANN. The structure of this data set is given in table 2.

Table 2. Structure of the dataset to train the ANN

| # | Instance | Features | Configuration Index |
|---|---|---|---|
| **1** | I1 | F1 | t : f(1,t) = max(f(1,1),f(1,2),…,f(1,M)) |
| **2** | I2 | F2 | t : f(2,t) = max(f(2,1),f(2,2),…,f(2,M)) |
| **…** | … | … | … |
| **N** | IN | FN | t : f(N,t) = max(f(N,1),f(N,2),…,f(N,M)) |

Once the entries of table 2 is on hand, the problem is boiled down to a classification problem: among all configuration combinations choose the *best* one. We choose to solve this classification problem with ANN.

## 3 Conclusion

In this study, we discuss that choosing the *best* configuration for the stochastic components of an SA implementation for TSP can be boiled down to a classification problem. By exploiting this idea, we propose an ANN to learn the relationship between instance features and configurations, and, algorithmic performance. We also propose that the trained ANN can be used as an engine to predict the optimal configurations for a newly given TSP instance.

## References


[1] S. Kirkpatrick, C. D. Gelatt et M. P. Vecchi (1983), Optimization by simulated annealing, *Science*, 671-680.

[2] E.-G. Talbi (2009), *Metaheuristics: from Design to Implementation*, John Wiley & Sons.

[3] L. Xu, F. Hutter, H. H. Hoos et K. Leyton-Brown (2007), SATzilla-07: the design and analysis of an algorithm portfolio for SAT, in *International Conference on Principles and Practice of Constraint Programming*, Providence, RI.

[4] K. M. Malan et A. P. Engelbrecht (2017), Fitness landscape analysis for metaheuristic performance prediction, in *Recent advances in the theory and application of fitness landscapes*, Springer, 103-132.

[5] T. Fischer, T. Stützle, H. Hoos et P. Merz (2005), An analysis of the hardness of TSP instances for two high performance algorithms, in *Proceedings of the Sixth Metaheuristics International Conference*, Vienna.