Towards the Empirical Analysis of SLS Algorithms for Multiobjective Combinatorial Optimization Problems through Experimental Design

Luis Paquete[*][†] Thomas Stützle[*] Manuel López-Ibáñez[‡]

[*]Fachgebiet Intellektik, Fachbereich Informatik, Technische Universität Darmstadt
Hochschulstraße 10, 64289 Darmstadt, Germany
{lpaquete,tom}@intellektik.informatik.tu-darmstadt.de

[†]Faculdade de Economia, Universidade do Algarve
Campus de Gambelas, 8005-139 Faro, Portugal

[‡]Centre for Emergent Computing, School of the Built Environment, Napier University
10 Colinton Road, EH10 5DT Edinburgh, UK
m.lopez-ibanez@napier.ac.uk

1 Introduction

Stochastic Local Search (SLS) algorithms for Multiobjective Combinatorial Optimization Problems (MCOPs) typically involve the selection and parameterization of many *algorithm components* whose role with respect to their overall performance and relation to certain instance features is often not clear. This fact is becoming more problematic because of the recent trend in solving MCOPs that is giving increasing attention to hybrid methods.

In this extended abstract we use a *modular* approach for the design of SLS algorithms for MCOPs that are solved in terms of Pareto optimality. We assume that SLS algorithms can be seen as combinations of *algorithm components* that can be coupled together to solve a given problem. These algorithm components have different targets that are important for successful approaches to solve MCOPs. Our approach offers the possibility of analyzing SLS algorithms by experimental design techniques. In fact, each algorithm component is considered a *factor* in experimental design, *i.e.*, it is an abstract characteristic of SLS algorithms that can affect random variables that describe the solution quality returned and the required computation time. Each factor has associated *levels* which are possible instantiations of the component. We show that different choices for algorithm components can affect the SLS algorithm in various ways, and that even the same choices can lead to different behavior in dependence of various instance features.

We illustrate our analysis using SLS algorithms for the biobjective Quadratic Assignment Problem (bQAP). The bQAP is defined as follows: given n facilities and n locations, one $n \times n$ matrix $\mathbf{A}$ where a_{ij} is the distance between locations i and j, and two $n \times n$ matrices $\mathbf{B}^1$ and

$\mathbf{B}^2$ where b^1_{rs} is the first flow and b^2_{rs} is the second flow between facilities r and s, the objective function in the bQAP can be stated as

$$\min_{\phi \in \Phi} \begin{cases} \sum_{i=1}^{n} \sum_{j=1}^{n} a_{\phi_i \phi_j} b^1_{ij} \\ \\ \sum_{i=1}^{n} \sum_{j=1}^{n} a_{\phi_i \phi_j} b^2_{ij}, \end{cases} \tag{1}$$

where Φ is the set of all permutations of $\{1, 2, ..., n\}$, ϕ_i gives the location assigned to facility i in the solution $\phi \in \Phi$, and "min" refers to the notion of Pareto optimality.

For our analysis, we implemented *straightforward* extensions of simple SLS algorithms for the single-objective QAP to tackle the bQAP by solving several scalarizations of the objective function vector for instances of different size, structure of the input data, and correlations between the flow matrices. As the underlying SLS algorithms we considered the use of an Iterative Improvement algorithm (II) based on the 2–exchange neighborhood for the QAP and, alternatively, the usage of Robust Taboo Search (RoTS) [17]. Performance assessment is carried out by means of attainment functions [5], which do not incur any of the known limitations of unary quality indicators for multiobjective algorithms [19].

2 Algorithm Components

We use scalarizations of the objective function vector based on a weighted sum formulation For tackling MCOPs in terms of Pareto optimality [15]. For each weight vector, we do one run of the underlying local search algorithm. Given its simplicity, this framework allows us to incorporate, in a straightforward way, the following algorithm components:

Dispersion Policy. A usual requirement on the set of solutions returned by SLS algorithms for MCOPs is that they are *spread* in the objective space. SLS algorithms with scalarizations of the objective function often use *maximally dispersed* weight vectors [15], which is also done here as follows: given Q objectives and $\binom{z+Q-1}{Q-1}$ distinct weight vectors, each vector $\lambda = (\lambda_1, \dots, \lambda_Q)$ is normalized such that $\sum_{q=1}^{Q} \lambda_q = 1$ and its components has values in $\{\frac{i}{z} \;/\; i = 0, \dots, z\}$.

Intensification Mechanism. Besides spreadness, also the quality of the individual solutions is an important factor in the performance assessment. High solution quality for each of the scalarizations can be obtained by, for example, increasing the number of iterations of the underlying SLS algorithm. This can be seen as an intensification mechanism for the search process whereas the dispersion policy introduces diversification.

Search Strategy. We consider two search strategies: The first starts the underlying SLS algorithm for each scalarization from a randomly generated initial solution (`Restart` strategy). The second strategy consists of the following two phases (`TPLS`): (i) obtain a high quality solution for one of the objectives; and (ii) solve a sequence of scalarized problems by starting from the best solution found in the previously tackled scalarization—in the first iteration the starting point is the solution returned in the first phase (see [13] for more details). The change of the weights in this second phase is done as follows: given two objectives, the weight vector in the first and last position in the sequence of weight vectors that are examined are $(1, 0)$ and $(0, 1)$, respectively; two successive weight vectors differ by only $\pm 1/z$ in any two components. This approach for generating the weight vectors can be easily extended to an arbritrary number of objectives using an algorithm to generate all compositions of z in Q parts in a Gray Code order as in [8]. (Note that by different strategies for changing the weights by, for example, allowing larger steps than of size $\pm 1/z$, different search behavior may result; however, such a study is beyond the scope of this article.)

Component-wise Step. Since the number of solutions of the two search strategies is bounded by the number of scalarizations, an extension is to accept non weakly dominated solutions in the neighborhood of each solution returned by a scalarization [13]; we call this additional procedure *component-wise step.*

3 Performance Assessment Methodology

The performance assessment and comparison of algorithms for MCOPs is by far more complex than in the single-objective case and fundamental critics have been raised against the use of unary quality indicators for the performance of algorithms for MCOPs [19]. These critics do not apply to the analysis of the solution quality of multiobjective opimizers by means of the attainment functions methodology [5]. This methodology associates the performance of a SLS algorithm for MCOPs to the probability of attaining *an arbritrary point* in the objective space; the function that characterizes this probability is called *attainment function* [5] and can be seen as a generalization of the distribution function of solution cost [6] to the multiobjective case. These probabilities can be estimated empirically from the outcomes obtained in several runs of an SLS algorithm by the *empirical attainment function* (EAF). Using EAFs, we can apply statistical hypothesis tests on the EAFs of several algorithms for a certain problem instance. Suitable test statistics are the maximum absolute distance between two EAFs for the two-sample case and the maximum absolute distance between k EAFs, for the k-sample case [1]; if the latter is rejected, pairwise comparisons between the k EAFs can be performed, where the returned p-values are corrected by Holm's procedure [7]. Since the distribution of these test statistics is not known, permutation tests [4] based on the above test statistics have to be performed [14]. If the null hypothesis (H_0) of the equality of the EAFs is rejected, a visualization of the performance difference between pairs of algorithms can be done by identifying regions of the objective space where the maximum absolute difference between EAFs is large (we defined large differences as above 20%) [5]; in addition, the sign of the differences indicates which algorithm performs better in which region. When the experimental goal is to study the main effects of several factors, the permutation procedure can be changed to eliminate the masking effect of other factors; in such a case, given a factorial experiment, the permutation test can be performed by restricted randomization [4]. Differently, the CPU time taken by the various configurations, which results to be a univariate distribution, can be analyzed by parametric statistics, if the assumptions of independence of error terms, equal variance and normality are verified [2]. Moreover, since we do not expect large differences of CPU time among instances of the same type (same correlation between flow matrices or same size), we can extend the notion of block used in our experimental design to include all instances of the same type.

4 Experimental Analysis

The bQAP instances were generated using the instance generator in [9]. We considered the following parameters: instance size $n \in \{25, 50, 75\}$, correlation between flow matrices $\rho \in \{-0.75, 0.0, 0.75\}$, and unstructured or structured instances (the unstructured instances are generated as those of class Taixxa [18] and the structured ones as the Taixxb [18] instances). We generated 3 instances for each combination, resulting in a total of 54 instances. We considered the following values for each algorithm component: $\{n, 5n, 10n\}$ scalarizations, $\{\text{II}, 50n, 100n\}$ tabu iterations (where II means that RoTS is replaced by an Iterative Improvement algorithm), `Restart` and `TPLS` strategies, and the use or not of the component-wise step; since `TPLS` should start from a good solution to one of the objectives, its first phase takes $10n^2$ tabu iterations. For each combination of these parameters, 5 trials were done on each instance in a full factorial experiment. In the following we give a summary of the findings in our statistical analysis using a significance level $\alpha = 0.05$.

4.1 Computation Times

In our experimental design, we allow for different settings of the levels of several algorithm components and we treat the computation time as a variable whose value depends on the particular configuration tested. There is also one more specific reason of leaving the computation time variable: When fixing computation time, without an exact cost model for the dependence between computation time and parameter settings, it is very difficult to choose the algorithm components and parameter settings in such a way as to guarantee termination of the algorithm within the given time limits and to still have a balanced experimental design. When analyzing the computation time, it is obvious that if everything is the same except, for example, the number of tabu search iterations is increased, also the computation times increase. While some of these tradeoffs are clear, we are more interested in analyzing the interactions between factors or to answer questions like whether the component-wise step does give a significant overhead in computation time if it is used. To answer these questions, we performed an ANOVA with respect to computation time. In order to meet the required assumptions for the ANOVA analysis, we divided the data into structured and unstructured input data; in addition, we defined instance size and the correlation of the flow matrices as *crossed blocks* [2]. Two interactions in common to both analysis were detected. These are (i) *tabu iterations* $\times$ *scalarizations* $\times$ *size* and (ii) *tabu iterations* $\times$ *scalarizations* $\times$ *search strategy*. The meaning of interaction (i) is the following: since the instance size affects the size of the neighborhood, this is reflected in the computation time resulting by changes in the number of tabu iterations or the number of scalarizations. Interaction (ii) means that search strategies behave differently with the change of tabu iterations and the number of scalarizations. Rather obviously, as also indicated by ANOVA, the number of iterations and scalarizations have the largest effect on the computation times, whereas, interestingly, the component-wise step has very little effect. Two more observations are noteworthy. The first is that `Restart` is faster than `TPLS` under the conditions set here. The reason is mainly due to the long first phase of `TPLS`. The second is that, when using II as the local search algorithm, there do not exist statistically significant differences between the `TPLS` search strategies with an increasing number of scalarizations (between n and $5n$ on unstructured instances, and among all scalarizations tested here on structured instances).

4.2 Solution Quality

As a next step, we analyze the influence of the various algorithm components on the solution quality as it is evaluated using attainment functions.

Search strategies. The null hypothesis (H_0) on the equality of the EAFs associated to the two search strategies was always rejected for all instances tested, which means that both search strategies produce statistically different outcomes. We detected large differences in the plots associated to both search strategies, which means that `Restart` and `TPLS` are performing better in different regions of the objective space; all plots showed differences in favor of `TPLS` in the region of the objective space with lowest values for the objective to which the first phase of `TPLS` was applied; as the size of the structured instances grows, the difference between `TPLS` and `Restart` becomes even larger in that objective. The correlation between flow matrices also plays apparently a strong role in the shape of the approximation to the Pareto front. This directly translates into a performance advantage of `Restart`, since the differences in favor of it cover a wide region on the instances with negatively correlated objectives. Finally, we observed large differences between search strategies in structured instances favoring `Restart`.

Component-wise step. Except for two unstructured instances of size 25 and the unstructured instances of size 50 and 75 with positively correlated flow matrices, H_0 was always rejected. This means that on most of the instances, in particular, all structured instances, the usage of the component-wise step resulted in improved quality. In addition, we only found differences in favor of the usage of the component-wise step. As the correlation of the flow matrices decreases in unstructured instances, also for these the benefit of using the component-wise step becomes more relevant.

Tabu iterations. H_0 was rejected for most unstructured instances (few exceptions occur when comparing $50n$ with $100n$ tabu iterations) and for structured instances of size 25, while for the structured instances of size 50 and 75 it could not be rejected. Looking at the location of large differences, one can observe that the influence of this component depends strongly on the structure of input data: while for unstructured instances the largest differences are found between II and $50n$ tabu iterations, for structured instances, in the few cases that H_0 is rejected, they occur between $50n$ and $100n$ tabu iterations.

Number of scalarizations. H_0 is mostly rejected, except when comparing $5n$ against $10n$ scalarizations in larger structured instances; this could indicate some limiting behavior above $5n$ scalarizations. The increase on the number of scalarizations corresponds to a roughly constant improvement in the front of non-dominated solutions, although the differences are not so well marked as with different tabu iterations.

Summary and general discussion. Note that the mentioned differences in the attainment functions can also be visualized in plots; some examples, which were also discussed above, are given in Figure 1. Through the systematic analysis of experimental designs, we could statistically validate the influence of different choices for the algorithm components and the instance features on the computation time and the solution quality found by specific kinds of SLS algorithms for MCOPs. For example, we identified the number of tabu iterations as the main responsible for the main marked differences both in terms of computation time and solution quality in unstructured instances; therefore, we have a clear trade-off of solution quality vs. computation time between different choices of this component (having the other components fixed): II returns lower solution quality, but takes very little time, whereas RoTS returns higher solution quality, but takes much more time. On the other hand, on structured instances, the increase of the number of tabu iterations is negligible in terms of the solution quality for large instances; in fact, even II seems to return reasonably good solution quality for those instances in much less time than RoTS variants. For structured instances, however, the increase of the number of scalarizations from n to $5n$ significantly improved solution quality, while increasing further the number of scalarizations to $10n$ diminishes this effect. Hence, on these instances $5n$ scalarizations should be preferable to $10n$ scalarizations. Another interesting aspect of this analysis is the interaction between the component-wise step and the correlation between flow matrices, with respect to solution quality; the relevance of this component grows when going from positive to negative correlations between flow matrices. Additionally, given the very minor (if noticeable at all) increase in computation time, we recommend its use in general.

The statistical differences between search strategies in all instances indicate that `Restart` and `TPLS` are far from producing similar outcomes; a noticeable difference is found in structured instances, indicating a better performance of `Restart`. This suggests that `Restart` is a good candidate as a baseline algorithm for comparing other strategies, given its good performance here. `TPLS` performs quite well on unstructured instances with positive correlation, although this is mainly due to the longer first phase of this search strategy. A variant of `TPLS` with the component-wise step was previously shown to be a state-of-the-art algorithm for the biobjective Traveling Salesman Problem (bTSP) [13]. One may have expected that it would perform better than a simple `Restart` strategy also for the bQAP, which was not the case here, at least for structured instances. There are several reasons for this. Firstly, the bQAP itself has a very different structure regarding the clustering of solutions as shown in [11]. Secondly, the usage of RoTS on structured bQAP instances is not ideal, since it is known to show also relatively poor performance on structured, single-objective QAP instances when compared to other algorithms like, for example, iterated local search. We compared `TPLS` against other high-performance algorithms for the bQAP and we found that for unstructured instances it was competitive to state-of-the-art algorithms [10]. For structured instances, the current version of `TPLS` is worse than state-of-the-art, but once one changes the local search from RoTS to a high-performing SLS algorithm for (single-objective) structured QAP instances, like iterated local search [16], again results competitive to the state-of-the-art are obtained [10]. Performance results that corroborate these claims will be shown in an extended version of this paper.

5 Conclusions and Further Work

This extended abstract shows that SLS algorithms for MCOPs can be analyzed by their algorithm components and that statistical statements about their performance can be provided by means of an experimental design methodology, both with respect to computation time and solution quality. Additionally, we show that certain instance features can have dramatic influence on the choice of the best levels of these components; therefore, we think that further experimental studies on SLS algorithms should focus on interactions between algorithm components and instance features in order to identify pairs of key-success factors leading to

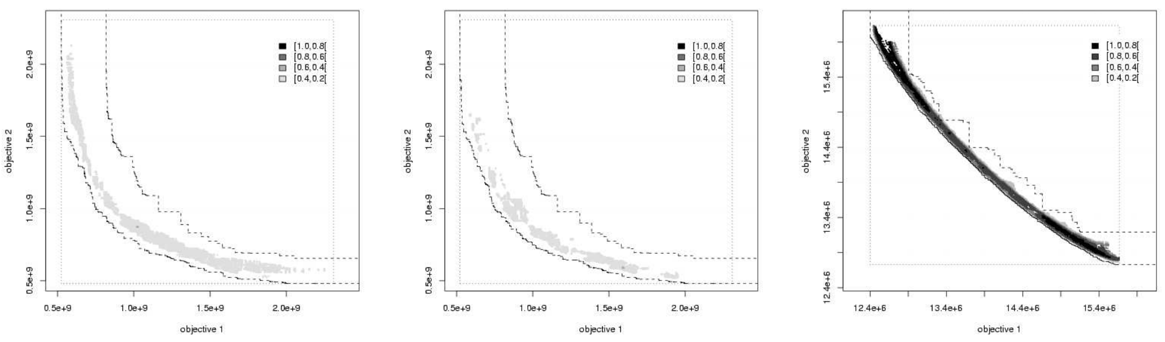

Figure 1: In the plots are given the locations of the differences above 0.2 (20%) in the EAFs comparing selected results of our analysis; the amount of differences are indicated by different shades of grey. *Left side:* Advantage of using $5n$ over n scalarizations on a structured instance with $n = 50$ and $\rho = 0.0$. *Middle:* Advantage of using the component-wise step over not using it on the same instance. *Right side:* Advantage of using $50n$ tabu search iterations over II for an unstructured instance with $n = 75$ and $\rho = -0.75$. The differences between the EAFs with a different sign are not shown, since all them were below 0.2.

high-performance SLS algorithms for MCOPs.

The current experimental methodology can be extended by considering also aspects of stochastic dominance between EAFs and the number of objectives as another blocking factor. In addition, other experimental scenarios can be tested, *e.g.*, the study of the development of the solution quality as a function of an algorithm's run-time; in this case, time can be considered as another objective [12]. Moreover, we could also study the *variance* of the outcomes in the objective space [3]. Finally, it would be interesting to apply systematic experimental designs also to more complex algorithms like multiobjective evolutionary algorithms or hybrids of it, in order to understand the contribution of various of their components like populations or the type of selection criteria.

References

[1] J. Conover. *Practical Nonparametric Statistics.* John Wiley & Sons, New York, 1980.

[2] A. Dean and D. Voss. *Design and Analysis of Experiments.* Springer Verlag, 1999.

[3] C. Fonseca, V. Grunert da Fonseca, and L. Paquete. Exploring the performance of stochastic multiobjective optimisers with the second-order attainment function. In C. C. Coello et al., editor, *Evolutionary Multi-criterion Optimization (EMO 2005)*, LNCS 3410, pages 250–264. Springer Verlag, 2005.

[4] P. I. Good. *Permutation Tests: A Practical Guide to Resampling Methods for Testing Hypothesis.* Spinger Series in Statistics. Springer Verlag, New York, 2000.

[5] V. Grunert da Fonseca, C. Fonseca, and A. Hall. Inferential performance assessment of stochastic optimizers and the attainment function. In E. Zitzler et al., editor, *Proceedings of EMO 2001*, LNCS 1993, pages 213–225. Springer Verlag, 2001.

[6] H. Hoos and T. Stützle. *Stochastic Local Search—Foundations and Applications.* Morgan Kaufmann Publishers, San Francisco, CA, 2004.

[7] J. Hsu. *Multiple Comparisons - Theory and Methods.* Chapman & Hall/CRC, 1996.

[8] Klingsberg. A gray code for compositions. *Journal of Algorithms*, 3:41–44, 1982.

[9] J. Knowles and D. Corne. Instance generators and test suites for the multiobjective quadratic assignment problem. In C. Fonseca, P. Fleming, E. Zitzler, K. Deb, and L. Thiele, editors, *Evolutionary Multi-criterion Optimization*, volume 2632 of *Lecture Notes in Computer Sience*, pages 295–310. Springer Verlag, 2003.

[10] M. López-Ibáñez, L. Paquete, and T. Stützle. Hybrid population-based algorithms for the bi-objective quadratic assignment problem. *Journal of Mathematical Modelling and Algorithms*, 2005. Accepted for publication.

[11] L. Paquete. *Stochastic Local Search Algorithms for Multiobjective Combinatorial Optimization: Methodology and Analysis.* PhD thesis, Technische Universität Darmstadt, Fachbereich Informatik, 2005. Forthcoming.

[12] L. Paquete and C. Fonseca. A study of examination timetabling with multiobjective evolutionary algorithms. In *4th Metaheuristics International Conference (MIC 2001)*, pages 149–154, Porto, 2001.

[13] L. Paquete and T. Stützle. A two-phase local search for the biobjective traveling salesman problem. In C. Fonseca et al., editor, *Evolutionary Multi-criterion Optimization (EMO 2003)*, LNCS 2632, pages 479–493. Springer Verlag, 2003.

[14] K. J. Shaw, C. Fonseca, A. L. Nortcliffe, M. Thompson, J. Love, and P. J. Fleming. Assessing the performance of multiobjective genetic algorithms for optimization of a batch process scheduling problem. In *Proceedings of the Congress on Evolutionary Computation (CEC'99)*, volume 1, pages 34–75, 1999.

[15] R.E. Steuer. *Multiple Criteria Optimization: Theory, Computation and Application.* John Wiley & Sons, New York, 1986.

[16] Thomas Stützle. Iterated local search for the quadratic assignment problem. *European Journal of Operational Research*, To appear.

[17] É. D. Taillard. Robust taboo search for the quadratic assignment problem. *Parallel Computing*, 17:443–455, 1991.

[18] É. D. Taillard. Comparison of iterative searches for the quadratic assignment problem. *Location Science*, 3:87–105, 1995.

[19] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. Grunert da Fonseca. Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Transactions on Evolutionary Computation*, 7(2):117–132, 2003.