Exploratory Analysis of Stochastic Local Search Algorithms in Biobjective Optimization

Manuel López-Ibáñez, Luis Paquete, and Thomas Stützle

IRIDIA – Technical Report Series

Technical Report No.
TR/IRIDIA/2009-015

May 2009

IRIDIA – Technical Report Series
ISSN 1781-3794

Published by:
IRIDIA, *Institut de Recherches Interdisciplinaires et de Développements en Intelligence Artificielle*
Université Libre de Bruxelles
Av F. D. Roosevelt 50, CP 194/6
1050 Bruxelles, Belgium

Technical report number TR/IRIDIA/2009-015

Exploratory Analysis of Stochastic Local Search Algorithms in Biobjective Optimization

Manuel López-Ibáñez, Luís Paquete, and Thomas Stützle

Abstract This technical report introduces two Perl programs that implement graphical tools for exploring the performance of stochastic local search algorithms for biobjective optimization problems. These tools are based on the concept of the empirical attainment function (EAF), which describes the probabilistic distribution of the outcomes obtained by a stochastic algorithm in the objective space. In particular, we consider the visualization of attainment surfaces and differences between the first-order EAFs of two algorithms. This visualization allows to identify certain algorithmic behaviors in a graphical way. We explain the use of these visualization tools and illustrate them with examples arising from practice.

1 Introduction

Experiments in computer science often produce large amounts of data, mainly because experiments can be setup, performed and repeated with relative facility. Given the amount of data, exploratory data analysis techniques are one of the most important tools that computer scientists may use to support their findings. In particular, specialized graphical techniques for representing data are often used to perceive trends and patterns in the data. For instance, there exist techniques for the extraction of relevant variables, the discovery of hidden structures, and the detection of outliers and other anomalies. Such exploratory techniques are mainly used during the design of an algorithm and when comparing the performance of various algorithms.

Luís Paquete
CISUC, Department of Informatics Engineering, University of Coimbra, Portugal
e-mail: paquete@dei.uc.pt

Manuel López-Ibáñez, Thomas Stützle
IRIDIA, CoDE, Université Libre de Bruxelles, Brussels, Belgium
e-mail: manuel.lopez-ibanez@ulb.ac.be, stuetzle@ulb.ac.be

Even before testing more formal hypotheses, the algorithm designer has to find patterns in experimental data that provide further insights into new ways of improving performance.

In this technical report, we focus on the graphical interpretation of the quality of the outcomes returned by Stochastic Local Search (SLS) algorithms [Hoos and Stützle, 2005] for biobjective optimization problems in terms of Pareto-optimality. This notion of optimality is tied to the notion of *dominance*. We say that a solution dominates another one if it is at least as good as the latter for every objective and strictly better for at least one objective. For these problems, the goal is to find the set of *nondominated* solutions among all feasible solutions. The mapping of these solutions in the objective space is called *Pareto-optimal front*. For the particular case of multiobjective combinatorial optimization problems (MCOPs), fundamental results about their properties and complexity can be found in Ehrgott [2000b].

Each run of an SLS algorithm produces a *nondominated set*, a random set of mutually nondominated objective vectors that approximates the Pareto-optimal front of an MCOP. Currently, there are two widely used techniques for assessing the performance of these algorithms with respect to the solution quality: graphical examination of multiple outcomes and scalar quality indicators. Unfortunately, these two approaches present several drawbacks that have been discussed in the literature [Knowles and Corne, 2002, Zitzler et al., 2003].

The empirical attainment function (EAF) [Grunert da Fonseca et al., 2001] is a middle ground between directly plotting the complete output and the extreme simplification of quality indicators. The EAF is a summary of the outcomes of multiple runs of an SLS algorithm, and, at the same time, it is sufficiently complex to detect whether and where an algorithm is better than another. By plotting and comparing the EAFs of different SLS algorithms, we are able to pin-point several performance behaviors that otherwise would be hidden when using other performance assessment approaches.

The technical report is organized as follows. Section 2 introduces basic concepts about multiobjective optimization in terms of Pareto optimality. Section 3 briefly summarizes the concept of attainment function and its empirical counterpart. Sections 4 and 5 introduce plotting techniques for exploring algorithm performance based on the empirical attainment function, and describe two Perl programs, `eafplot.pl` and `eafdiff.pl`. Section 6 presents three examples of applications of these programs. Finally, we present conclusions and further work in Sect. 7.

2 Stochastic Local Search for Multiobjective Problems

SLS algorithms iteratively search for good quality solutions using the local knowledge provided by the definition of a neighborhood or a set of partial solutions. Since they are based on a randomized search process, it is not expected that the same outcome is returned for different runs with different random seeds of the random number generator. *Metaheuristics* are general-purpose SLS methods that can be adapted to various optimization problems. Well known metaheuristics are Simulated Annealing, Tabu Search, Iterated Local Search, Variable Neighborhood Search, Ant Colony Optimization, and Evolutionary Algorithms. An overview of these methods is given by Hoos and Stützle [2005].

In this technical report, we focus on SLS algorithms that provide approximations to the optimal set of solutions for multiobjective combinatorial optimization problems (MCOPs). The quality of the solutions of MCOPs is evaluated by an *objective function vector* $\mathbf{f} = (f_1, \ldots, f_d)$, where d is the number of objectives.

Without preference information, the main goal of solving an MCOP is to find a set of feasible solutions that "minimizes" the objective function vector $\mathbf{f}$ according to the notion of Pareto-optimality. Let $\mathbf{u}$ and $\mathbf{v}$ be vectors in $\mathbb{R}^d$. We say that $\mathbf{u}$ *dominates* $\mathbf{v}$ ($\mathbf{u} \prec \mathbf{v}$) if and only if $\mathbf{u} \neq \mathbf{v}$ and $u_i \leq v_i,\ i = 1, \ldots, d$. In addition, we say that $\mathbf{u}$ and $\mathbf{v}$ are *nondominated* if and only if $\mathbf{u} \not\prec \mathbf{v}$ and $\mathbf{v} \not\prec \mathbf{u}$. For simplicity, we shall use the same relations among solutions when the relation between their image in the objective space is as described above. Formally, given two feasible solutions s and s' to a MCOP, we say that $s \prec s'$ if and only if $\mathbf{f}(s) \prec \mathbf{f}(s')$.

The notion of optimal solution in multiobjective optimization clearly differs from the single-objective counterpart. We say that a feasible solution s is a *Pareto optimum* if and only if there is no other feasible solution s' such that $\mathbf{f}(s') \prec \mathbf{f}(s)$. In addition, we say that a set of feasible solutions is the *Pareto optimal set* if and only if it contains *only* and *all* Pareto optimum solutions. The image of the Pareto optimal set in the objective space is often called *Pareto frontier*.

Finding the Pareto-optimal set in MCOPs is known to be a hard challenge in optimization [Ehrgott, 2000a]. For many problems, the size of the Pareto-optimal set is too large to be enumerated. Therefore, depending on the time constraints, it could be preferable to have an approximation to the Pareto-optimal set in a reasonable amount of time. Such an approximation is always a *nondominated set*, that is, a set of solutions that are mutually nondominated.

The Pareto-optimality principle can be extended to compare pairs of nondominated sets [Hansen and Jaszkiewicz, 1998, Zitzler et al., 2003] and be used as a natural way of comparing algorithm performance. Given two arbitrary nondominated sets in a d-dimensional objective space, A and B, we can say that A *is better than* B in terms of Pareto-optimality ($A \lhd B$) if and only if every $\mathbf{b} \in B$ is dominated by or equal to at least one $\mathbf{a} \in A$, and $A \neq B$. If we have that $A \ntriangleleft B$, $B \ntriangleleft A$, and $A \neq B$, we say that A and B are *incomparable* ($A \parallel B$), and neither set can be preferred over the other according only to the notion of Pareto-optimality.

SLS algorithms have been shown to be state-of-the-art methods for generating very good approximations to the Pareto-optimal set for many MCOPs. As a result, when comparing two SLS algorithms, we often need to compare nondominated sets that are incomparable in the Pareto sense. There are mainly two approaches for summarizing and comparing SLS algorithms with respect to solution quality: direct examination of multiple nondominated sets and scalar quality indicators.

As an example of direct examination, we plot in Fig. 1 the outcomes obtained by ten runs of the same SLS algorithm applied to an instance of a biobjective optimization problem. On the left plot, we plot the objective vectors as points. Points with

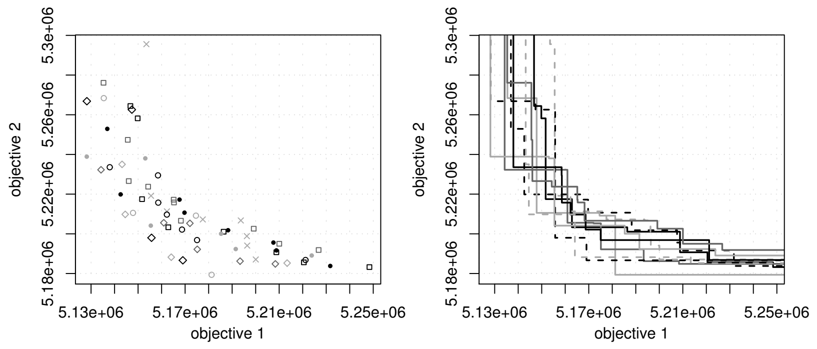

Fig. 1 Ten independent outcomes obtained by an SLS algorithm applied to an instance of a biobjective optimization problem. In the right plot, the same outcomes are shown but points belonging to the same run are joined with a line.

the same shade of gray and shape were obtained in the same run. On the right plot, objective vectors from the same run are joined with stair-case lines delimiting the area dominated by them. Even with only ten runs, it is difficult to visualize the algorithm behavior. With a larger number of runs, directly plotting the outcomes quickly becomes impractical. A direct comparison of the outcomes of different algorithms is similarly difficult.

On the other extreme are scalar quality indicators, which are scalar values computed for each nondominated set (or pairs of nondominated sets) [Knowles and Corne, 2002, Zitzler et al., 2003]. Quality indicators are surrogate measures of particular quality aspects of the nondominated sets (e.g. closeness to the best-known solutions, spread, diversity) that are generally acknowledged as desirable. Hence, several quality indicators are often examined simultaneously, but this, in turn, complicates the interpretation of results. The values returned by quality indicators often do not reflect by how much and in which aspects a nondominated set is better than another. Moreover, recent theoretical work has shown that no combination of unary quality indicators is in general able to detect whether an output set is better than another and, at most, they can detect whether an output set is not worse [Zitzler et al., 2003]. In fact, many quality indicators may provide an answer that contradicts the Pareto-optimality principle [Knowles and Corne, 2002, Zitzler et al., 2003].

The empirical attainment function (EAF) provides a compromise between these two extremes. On the one hand, the EAF summarizes the outcomes of multiple runs of an SLS algorithm. On the other hand, it is able to illustrate where in the objective space and by how much the outcomes of two algorithms differ.

3 The Empirical Attainment Function

The Pareto-optimality relations provide a reliable way to compare the output of two SLS algorithms. If the outcomes of one algorithm are better (according to the $\triangleleft$-relation) than the outcomes of another algorithm, we can confidently say that the former gives a better approximation to the Pareto-optimal set. If this first step shows no clear advantage of one algorithm over the other, we have a strong indication that their outcomes are incomparable most of the time. Therefore, we are now interested in knowing where the outcomes of the algorithms differ and how large this difference is. This is exactly the information that can be provided by the attainment function.

The attainment function represents the probability that an arbitrary objective vector in the objective space is *attained* (that is, dominated or equal) in a single run of a particular algorithm [Grunert da Fonseca et al., 2001]. In practice, this function is unknown, but it can be estimated using data collected from several independent runs of an SLS algorithm. Such an estimate is called the empirical attainment function (EAF).

The EAF can be seen as a distribution of the solution quality after running a particular algorithm for a specific amount of computation time. In this sense, it is an extension of the solution quality distribution of SLS algorithms for the single objective case [Hoos and Stützle, 2005]. The further developments described in the literature have led to methods for statistical inference based on the EAF [Fonseca et al., 2005, Grunert da Fonseca and Fonseca, 2002, Paquete and Fonseca, 2001, Shaw et al., 1999].

In this technical report, we focus on the first-order EAF, which gives the empirical frequency of attaining an objective vector in the objective space in a single optimization run. Higher order EAFs characterize the dependence structure within each outcome, and thus, they allow to study the frequency of two or more objective vectors being attained in the same run [Fonseca et al., 2005]. In the following, we will always refer to the first-order EAF simply as EAF.

In the biobjective case, the EAF is both fast to compute and easy to visualize. We will consider two different visualizations. First, plots of the $k\%$-attainment surfaces are used to characterize the behavior of a single SLS algorithm. Second, the performance of two SLS algorithms is compared by plotting the location of the differences with respect to their EAFs [Paquete et al., 2004].

4 Examination of the Attainment Surfaces

Fonseca and Fleming [1996] proposed the notion of *attainment surface*, which corresponds to a boundary which separates the objective space in two regions: those objective vectors that are attained by (dominated by or equal to) the outcomes returned by the SLS algorithm, and those that are not. This notion is formalized in the concept of $k\%$-attainment surface, which is the line separating the objective space attained by k percent of the runs of an SLS algorithm. In other words, the $k\%$-attainment surface corresponds to the $k/100$ percentiles of the empirical frequency distribution. For example, the *median* attainment surface delimits the region attained by 50 percent of the runs. Similarly, the *worst* attainment surface delimits the region attained by all runs (100%-attainment surface), whereas the *best* attainment surface corresponds to the limit between the region attained by at least one run and the objective vectors never attained by any run.

Given m runs, the computation of the EAF is equivalent to the computation of all $k\%$-attainment surfaces with $k = i \cdot 100/m$, $i = 1, \ldots, m$. In fact, the $k\%$-attainment surface is sufficiently defined by the nondominated objective vectors from the set of all objective vectors that are attained by k percent of the runs.

The attainment surfaces allow to summarize the behavior of an SLS algorithm in terms of the location of the objective vectors obtained. For example, if we were interested in the objective vectors that are attained by at least half of the runs, then we could examine the median attainment surface. Similarly, the worst-case results of an algorithm are described by the worst attainment surface, whereas the best results ever achieved are given by the best attainment surface. Sections 4.2 and 6.1 give examples.

4.1 The `eafplot.pl` Program

The program `eafplot.pl` is a Perl program that produces a plot of attainment surfaces given an input file that contains a number of nondominated sets. If several input files are given, then one plot is produced for each input file, all plots with the same range on the axes. The plots produced are encapsulated postscript (EPS) files.

These programs require a Perl installation, the statistical environment R [R Development Core Team, 2008], and an external program for computing the EAF.[1]

Input files may contain multiple sets of nondominated objective vectors. Each objective vector is given in a line as two columns of floating-point numbers. Different sets are separated by at least one blank line.

The attainment surfaces plotted by `eafplot.pl` can be specified in several ways:

- By default, `eafplot.pl` plots the best, median, and worst attainment surfaces.
- Option `--iqr` plots the 25%, 50% (median), and 75% attainment surfaces.
- Option `--percentile=INT[,INT]` plots the given percentiles of the attainment surface. For example, `eafplot.pl --percentile=25,50,75` is equivalent to `eafplot.pl --iqr`.
- Option `--extra=FILE` will add objective vectors from `FILE` to the plot as points. This may be useful for comparing the outcome of an SLS algorithm against a reference set.

[1] The program for computing the EAFs provided by us is based on the original code written by Carlos M. Fonseca available at `http://www.tik.ee.ethz.ch/pisa/`.

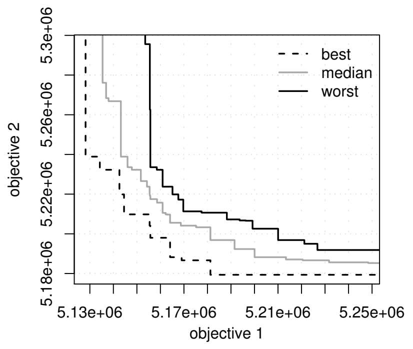

Fig. 2 Best, median and worst attainment surfaces for the data described in Fig. 1.

The program accepts other parameters that are not discussed in this technical report but are explained by the option `--help`.

4.2 Example Application of `eafplot.pl`

Given the input data shown in Fig. 1, the corresponding best, median and worst attainment surfaces are shown in Fig. 2. This plot was generated by the command:

```
eafplot.pl example1_dat
```

As an alternative to the best and worst attainment surfaces, one may prefer to plot other percentiles that are more robust with respect to the number of runs. The dependence of the best and worst attainment surfaces on the number of runs is illustrated by Fig. 3, where the same algorithm is run 15 (left), 50 (middle), and 200 (right) times with different random seeds. As more runs are performed, the locations of the best and worst attainment surfaces change strongly, while the locations of the 25% and 75% attainment surfaces are rather stable. It is well-known from classical statistics that the sample best and worst are biased estimators for the population best and worst. The three plots in Fig. 3 were produced by running:

```
eafplot.pl --best --median --worst  \
    --percentiles=25,75  r15_dat r50_dat r200_dat
```

5 Examining the differences between EAFs

The EAF is also the basis for a graphical technique that gives visual information on the pairwise comparison of two SLS algorithms. The main idea is to plot the location of the differences between the outcomes of two algorithms with respect

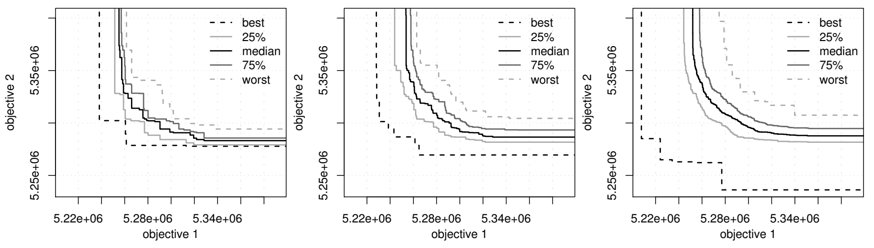

Fig. 3 Three plots of attainment surfaces for 15 (left), 50 (middle), and 200 (right) independent runs of the same algorithm on the same problem instance.

to their corresponding EAFs. The EAF of an algorithm estimates the probability of attaining each point in the objective space. If the difference of the estimated probability values of two SLS algorithms at a certain point is large, this indicates a better performance of one algorithm over another at that point. The sign of the difference gives information about which algorithm performed better.

The differences between the EAFs of two algorithms can be computed by first computing the EAF of the union of the outcomes of both algorithms. Then, for each point in the objective space where the value of the EAF changes, one needs to compute the value of the EAF of the first algorithm at that point minus the value of the EAF of the second algorithm. This can be done by counting how many runs of each algorithm attained that point. Finally, positive and negative differences are plotted separately, and the magnitudes of the differences between the EAFs are encoded using different shades of grey: the darker a point, the larger is the difference.

Figure 4 illustrates this performance assessment method. The two plots in the top part of Fig. 4 give the EAFs associated to two algorithms that were run several times with different random seeds on the same problem instance. Points in the EAFs are assigned a gray level according to their probability. In addition, we plot four different attainment surfaces. The lower line on both plots connects the best set of points attained over all runs of both algorithms (*grand* best attainment surface), and the upper one the set of points attained by any of the runs (*grand* worst attainment surface). Any differences between the algorithms are contained within these two lines. The dashed line corresponds to the median attainment surface of each algorithm, which is given to facilitate the comparison of the two sides of the plot.

The bottom plots of Fig. 4 show the location of the differences between the EAFs of the two algorithms. On the left are shown points where the EAF of Algorithm 1 is larger by at least 20 percent than that of Algorithm 2, and on the right are given the differences in the opposite direction (positive differences between the EAF of Algorithm 2 over the one of Algorithm 1). The amount of the differences is encoded in a grey scale shown in the legend of the plot. To facilitate comparison, the same attainment surfaces are plotted as for the top plots. From these plots, we can observe that Algorithm 1 performs better in the center and towards the minimization of objective 1, whereas Algorithm 2 performs better towards high quality solutions for the second objective (low values on the y-axis).

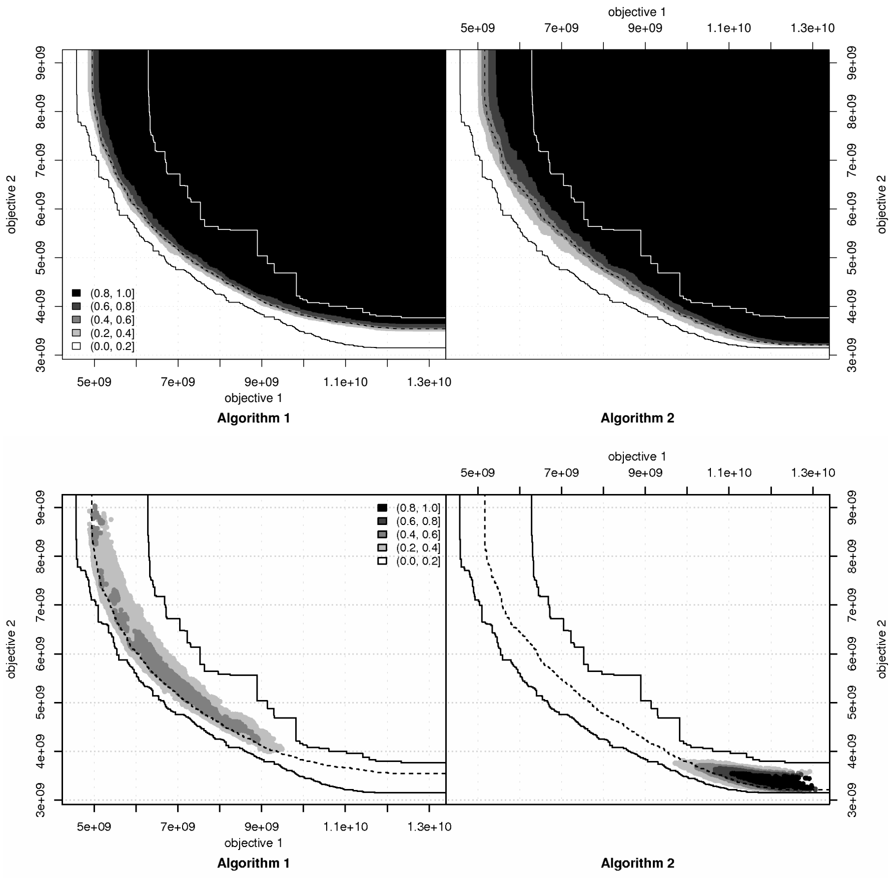

Fig. 4 Visualization of the EAFs associated to the outcomes of two algorithms (*top*) and the corresponding differences between the EAFs (*bottom left*: differences in favour of Algorithm 1; *bottom right*: differences in favour of Algorithm 2). In the top, the gray level encodes the value of the EAF. In the bottom, the gray level encodes the magnitude of the observed difference.

Note that these differences in performance would be ignored by most scalar quality indicators. In this particular example, each of the two data sets contains 90 runs with an average of 500 objective vectors per run. The generation of the plot from these data sets required less than 10 seconds of computation time on a Intel Core^{TM} 2 CPU with 1.83 GHz.

Output plot	*file1-file2*.eps
Grand best attainment surface	*file1-file2*.best
Grand worst attainment surface	*file1-file2*.worst
Differences between EAFs	*file1-file2*.diff
Full EAF of input file `fileX`	*fileX*.eaf
Median attainment surface of file `fileX`	*fileX*.med

Table 1 Output files produced by `eafdiff.pl` given input files `file1` and `file2`.

5.1 The `eafdiff.pl` Program

The program `eafdiff.pl` is a Perl program that takes two input files, each of which contains a number of nondominated sets, and produces a plot of the differences between the first-order EAFs of the two input files.

The `eafdiff.pl` program can produce two types of plots:

- A side-by-side plot of the full EAF of each of the input files. This type of plot can be requested by using the option `--full`. For example, the top plot of Fig. 4 was produced by the commandline:

```
eafdiff.pl --full --left="Algorithm 1" ALG_1_dat \
                  --right="Algorithm 2" ALG_2_dat
```

- A side-by-side plot of the differences in the EAFs between the two input files. This is the default. For example, the bottom plot of Fig. 4 was generated by:

```
eafdiff.pl --left="Algorithm 1" ALG_1_dat \
           --right="Algorithm 2" ALG_2_dat
```

By default, `eafdiff.pl` plots also the grand best and grand worst attainment surfaces as solid black lines, and the median attainment surface corresponding to each input file as dashed lines. The program accepts other parameters that are not discussed in this technical report but are explained by the option `--help`.

Apart from the plot *file1-file2*.eps, the `eafdiff.pl` program produces several output files. These are listed in Table 1.

6 Examples

In this section, we illustrate the use of the graphical techniques on several examples. As we will see, these tools allow us to discover particular algorithm behaviors.

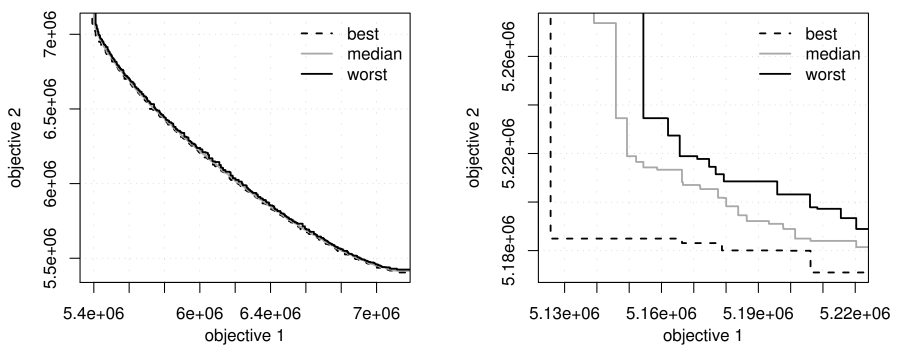

Fig. 5 The same algorithm is applied to two BQAP instances with correlation -0.75 (left) and 0.75 (right). The plots show the best, median and worst attainment surfaces of 10 runs.

6.1 Effect of Problem Structure

Problem structure has a clear effect on algorithm performance. In multiobjective optimization, the correlation between the objectives is often an example of this, since we expect that a smaller correlation between the objectives may induce a smaller number of Pareto-optimal solutions, and vice versa [Mote et al., 1991].

We reproduce here experiments described by López-Ibáñez et al. [2006]. We consider two instances of the Biobjective Quadratic Assignment Problem (BQAP) with different correlations between the flow matrices, which translate into different correlations between the corresponding objectives. (See López-Ibáñez et al. [2006] for a more thorough explanation of this problem).

We plot in Fig. 5 the best, median and worst attainment surfaces of the outcomes obtained by the same algorithm when applied to two BQAP instances with correlation -0.75 (left) and 0.75 (right). In each case, ten independent runs of the algorithm were performed with different random seeds. Even thought both instances are similar in terms of size and range of values, the range of the nondominated sets obtained for correlation -0.75 (left) is much wider than for correlation 0.75 (right), as can be seen in the range of the objective values in each of the plots. The plots show a strong effect of the correlation on the location of the outcomes obtained by the algorithm.

The two plots in Fig. 5 were produced by running:

```
eafplot.pl n75_dat

eafplot.pl p75_dat
```

6.2 Differences in algorithmic performance

Several approaches to biobjective problems consist in solving several weighted scalarizations of the objective function vector. Usually, the components of the weight vectors are real numbers in $[0,1]$ and sum to one. With a subset of weight vectors evenly distributed in the full set of possible weight vectors, we may expect to find a well spread set of objective vectors.

Two distinct algorithmic behaviors may be expected either by increasing the number of weights or by running the underlying stochastic algorithm for a longer time for each scalarization. Intuitively, by increasing the number of weights, the algorithm should be able to obtain a larger number of nondominated objective vectors distributed along the Pareto front, which gives a better approximation of the shape of the Pareto-optimal front. On the other hand, by giving more time to each scalarization, the resulting nondominated objective vector is typically of a higher quality. Because of limits in computation time, the algorithm designer has to examine the trade-off between these two settings in order to improve solution quality.

We describe an experiment that examines the trade-off between different parameter settings of WRoTS (Weighted Robust Tabu Search) for the Biobjective QAP, such as described by López-Ibáñez et al. [2006]. In WRoTS, several scalarizations of the BQAP objective function vector are solved by repeated runs of the (single-objective) Robust Tabu Search (RoTS) algorithm [Taillard, 1991]. A single run of the RoTS algorithm is stopped after $l \cdot n$ iterations, where n is the instance size and l is a parameter. Each scalarization uses a different weight, taken from a set of maximally dispersed weights. We denote the number of weights by w. Upon termination of the main search process, all solutions returned are filtered to obtain a set of nondominated objective vectors.

Figure 6 shows the differences in EAFs between two algorithm configurations of WRoTS. The left plot shows differences in favor of performing few scalarizations and long runs of RoTS ($l = 100$, $w = 10$); whereas the right plot shows differences in favor of performing many short runs of RoTS ($l = 10$, $w = 100$). Note that the total number of iterations performed by each algorithm is the same. There are very strong differences in three particular regions of the right plot (many scalarizations and short runs of RoTS). These regions are probably difficult to attain with the coarse set of weights examined by the left configuration. On the other hand, the use of a larger number of iterations of RoTS (left plot) does not lead to better individual objective vectors, except for the extreme objective values.

The plots in Fig. 6 were generated by the command:

```
eafdiff.pl --left="WRoTS, l=100, w=10"  \
           --right="WRoTS, l=10, w=100" \
           wrots_l100w10_dat wrots_l10w100_dat
```

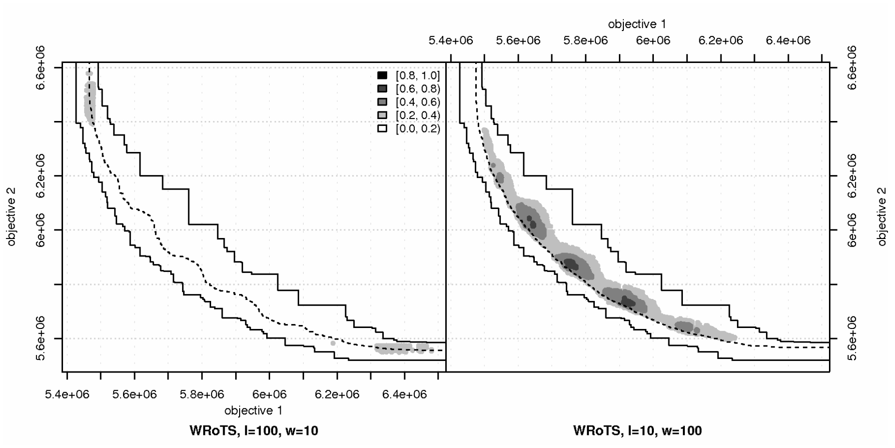

Fig. 6 EAF differences for two configurations of W-RoTS. The left plot shows differences in favor of long runs of RoTS and few scalarizations ($l = 100$, $w = 10$); the right plot shows differences in favor of short runs of RoTS and many scalarizations ($l = 10$, $w = 100$).

6.3 Biased Behavior

An algorithm may be focusing too much on a particular region of the objective space in detriment of other equally relevant regions, which may be due to some algorithmic choice. In this example, we present a case in which an algorithm is more biased towards one objective.

We describe an algorithm that was proposed by Paquete and Stützle [2003], called two-phase local search (TPLS): the first phase consists of finding a good solution to one single objective, using an effective single objective algorithm. This phase provides the starting solution for the second phase, in which a local search algorithm is applied to a sequence of different scalarizations of the objectives. The underlying idea for the second phase is that successive scalarizations are treated as a chain: a scalarization modifies slightly the emphasis given to the different objectives when compared to the previous scalarization; the local search for each scalarization is started from the local optimum solution that was returned by the previous scalarization. The main question is whether this strategy can have comparable performance to a restart strategy (Restart), which starts from a randomly generated solution at every scalarization.

The experimental analysis was performed for the Biobjective Travelling Salesman Problem. Both algorithms, TPLS and Restart, have the same underlying stochastic local search, an Iterated Local Search [Stützle and Hoos, 2001].

Figure 7 shows the EAF differences between TPLS and Restart. TPLS is able to obtain good solutions with respect to the second objective. However, TPLS is not able to improve the solutions obtained by Restart with respect to the first objective.

The plots in Fig. 7 were obtained with the command:

```
eafdiff.pl --left="TPLS" --right="Restart" tpls rest
```

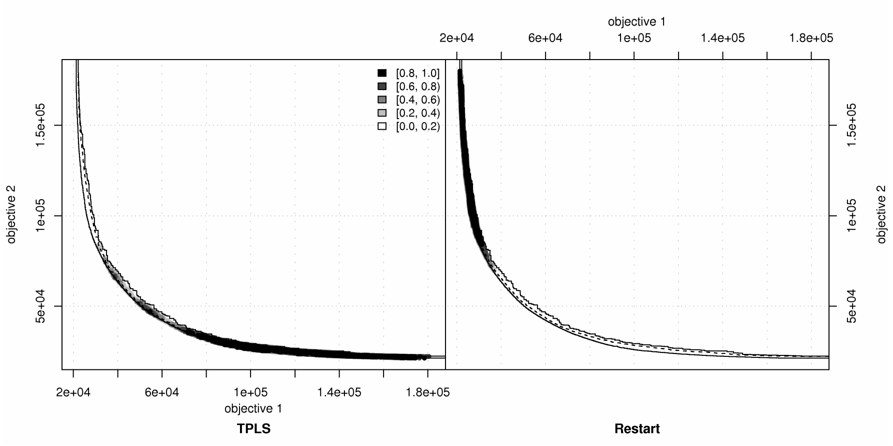

Fig. 7 EAF differences for TPLS versus Restart.

7 Summary and Outlook

In this technical report, we have described graphical techniques for summarizing and comparing the quality of SLS algorithms for biobjective problems in terms of Pareto-optimality. We have also described two programs that implement these techniques. In addition, examples of the usage of these programs were provided throughout the report. These examples can be reproduced by using the programs and data available at `http://iridia.ulb.ac.be/supp/IridiaSupp2009-002/`.

These graphical techniques are based on the first-order EAF for biobjective optimization problems. For more than two objectives, the graphical technique of parallel coordinates [Inselberg, 1985] has been used by Paquete and Stützle [2009]. However, the interpretation of the plots is more difficult than in the biobjective case. Therefore, other ways to present the information given by the first-order EAF may be worth investigating. Finally, new techniques could be developed based on the information provided by higher-order EAFs [Fonseca et al., 2005].

Acknowledgements This work was supported by the META-X project, an *Action de Recherche Concertée* funded by the Scientific Research Directorate of the French Community of Belgium. Thomas Stützle acknowledges support from the Belgian F.R.S.-FNRS, of which he is a Research Associate. The authors also acknowledge Carlos M. Fonseca for providing us the code for computing the EAF as well as for the helpful discussions on the main topic of this technical report.

References

M. Ehrgott. Hard to say it's easy - Four reasons why combinatorial multiobjective programmes are hard. In Y. Y. Haimes and R. E. Steuer, editors, *Research* *and Practice in Multiple Criteria Decision Making*, volume 487 of *Lecture Notes in Economics and Mathematical Systems*, pages 69–81. Springer Verlag, Berlin, Germany, 2000a.

M. Ehrgott. *Multicriteria Optimization*, volume 491 of *Lecture Notes in Economics and Mathematical Systems*. Springer Verlag, Heidelberg, Germany, 2000b.

C. M. Fonseca and P. Fleming. On the performance assessment and comparison of stochastic multiobjective optimizers. In H.-M. Voigt, W. Ebeling, I. Rechenberg, and H.-P. Schwefel, editors, *Proceedings of PPSN-IV, Fourth International Conference on Parallel Problem Solving from Nature*, volume 1141 of *Lecture Notes in Computer Science*, pages 584–593. Springer Verlag, Berlin, Germany, 1996.

C. M. 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, A. H. Aguirre, and E. Zitzler, editors, *Evolutionary Multi-criterion Optimization (EMO 2005)*, volume 3410 of *Lecture Notes in Computer Science*, pages 250–264. Springer Verlag, Berlin, Germany, 2005.

V. Grunert da Fonseca and C. M. Fonseca. A link between the multivariate cumulative distribution function and the hitting function for random closed sets. *Statistics & Probability Letters*, 57(2):179–182, 2002.

V. Grunert da Fonseca, C. M. Fonseca, and A. Hall. Inferential performance assessment of stochastic optimizers and the attainment function. In E. Zitzler, K. Deb, L. Thiele, C. C. Coello, and D. Corne, editors, *Evolutionary Multi-criterion Optimization (EMO 2001)*, volume 1993 of *Lecture Notes in Computer Science*, pages 213–225. Springer Verlag, Berlin, Germany, 2001.

M. P. Hansen and A. Jaszkiewicz. Evaluating the quality of approximations to the non-dominated set. Technical Report IMM-REP-1998-7, Institute of Mathematical Modelling, Technical University of Denmark, Lyngby, Denmark, 1998.

H. Hoos and T. Stützle. *Stochastic Local Search – Foundations and Applications*. Morgan Kaufmann Publishers, San Francisco, CA, 2005.

A. Inselberg. The plane with parallel coordinates. *Visual Computer*, 1(4):69–91, 1985.

J. Knowles and D. Corne. On metrics for comparing non-dominated sets. In *Proceedings of the 2002 Congress on Evolutionary Computation (CEC'02)*, pages 711–716. IEEE Press, Piscataway, NJ, 2002.

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*, 5(1):111–137, 2006.

J. Mote, I. Murthy, and D. L. Olson. A parametric approach to solving bicriterion shortest path problems. *European Journal of Operational Research*, 53(1):81–92, 1991.

L. Paquete and C. M. Fonseca. A study of examination timetabling with multiobjective evolutionary algorithms. In *Proceedings of the Fourth Metaheuristics International Conference*, pages 149–154, Porto, Portugal, 2001.

L. Paquete and T. Stützle. Design and analysis of stochastic local search algorithms for the multiobjective traveling salesman problem. *Computers & Operations Research*, 36(9):2619–2631, 2009.

L. Paquete and T. Stützle. A two-phase local search for the biobjective traveling salesman problem. In C. M. Fonseca, P. Fleming, E. Zitzler, K. Deb, and L. Thiele, editors, *Proceedings of the Evolutionary Multi-criterion Optimization (EMO 2003)*, volume 2632 of *Lecture Notes in Computer Science*, pages 479–493. Springer Verlag, Berlin, Germany, 2003.

L. Paquete, M. Chiarandini, and T. Stützle. Pareto local optimum sets in the biobjective traveling salesman problem: An experimental study. In X. Gandibleux, M. Sevaux, K. Sörensen, and V. T'kindt, editors, *Metaheuristics for Multiobjective Optimisation*, volume 535 of *Lecture Notes in Economics and Mathematical Systems*, pages 177–200. Springer Verlag, Berlin, Germany, 2004.

R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2008. URL `http://www.R-project.org`.

K. J. Shaw, C. M. 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 1999 Congress on Evolutionary Computation (CEC'99)*, volume 1, pages 34–75. IEEE Press, Piscataway, NJ, 1999.

T. Stützle and H. Hoos. Analysing the run-time behaviour of iterated local search for the travelling salesman problem. In P. Hansen and C. Ribeiro, editors, *Essays and Surveys on Metaheuristics*, pages 589–611. Kluwer Academic Publishers, Boston, MA, 2001.

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

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.