2014 IEEE Congress on Evolutionary Computation (CEC)
July 6-11, 2014, Beijing, China

The Effect of Different Local Search Algorithms on the Performance of Multi-Objective Optimizers

Martin Pilát Roman Neruda

***Abstract*—Several contemporary multi-objective surrogate-based algorithms use some kind of local search operator. The search technique used in this operator can largely affect the performance of the multi-objective optimizer as a whole, however, little attention is often paid to the selection of this technique. In this paper, we compare three different local search techniques and evaluate their effect on the performance of two different surrogate based multi-objective optimizers. The algorithms are evaluated using the well known ZDT and WFG benchmark suites and recommendations are made based on the results.**

I. Introduction

Multi-objective evolutionary algorithms have gained a lot of attention in the recent decades. They are considered one of the best multi-objective optimizers and are widely used both in theory and practice. However, their usefulness is limited by the large number on objective function evaluations they require. In practice, these evaluations can be rather costly, both in terms of computational power and money.

This limitation is most often overcome by the introduction of so called surrogate models. These are a fast approximation of the real expensive objective function and can be used in lieu of the real objective to reduce the number of times it needs to be evaluated, thus decreasing the cost of the optimization considerably. Most often, these models are obtained by training a machine learning regression model. The model is trained using the previously evaluated points in the search space and is used to predict the values of the new points.

The surrogate model may be used in different ways during the local search. In some cases, individuals are improved using the models by starting a local search algorithm from these individuals [1]. In other cases [2], [3], more individuals are generated, evaluated by the model, and only the best of them are selected for evaluation and inclusion in the next generation. Yet another approach [4], [5] switches between the optimization of the real objectives and the optimization of the surrogate model. The algorithms used for the optimization in both phases can be the same, or different.

If we generalize the notion of local search just a little bit, all of the above described techniques can be viewed as a special case of local search. When more individuals are generated than evaluated (i.e. some filtering, or pre-screening, of the individuals is used), the process which generates the individuals can be considered a local search algorithm. Similarly, algorithms which switch between the two phases of optimization may also be viewed as performing local search of the model – the phase which uses only the surrogate model evaluations is a local search.

In this work, by *surrogate model based multi-objective evolutionary algorithm* we mean any evolutionary algorithm which uses any kind of modeling of the fitness function and uses the evaluations of the model with the goal to reduce the overall number of evaluations of the fitness function. Moreover, by *local search* we denote any way of generating new individuals other than the usual uninformed (i.e. random) genetic operators.

Although local search techniques are often used in surrogate-based multi-objective algorithms, the reasoning about the selection of a particular local search technique is rarely discussed. However, selection of a proper local search can improve the performance of the algorithm considerably, as we shall see later. In this work, we compare three different popular local search techniques and discuss, how the selection of a given search technique affects the performance of the multi-objective evolutionary algorithm as a whole. For the experiments, we use two of our surrogate-based multi-objective EAs – ASM-MOMA and HO-MOMA.

Of course, as per the no free lunch theorem (NFL) [6] we cannot expect that there will be a single local search algorithm working the best for the whole range of problems we can obtain. However, there should be at least some identifiable classes of problems for which a given algorithm will have the best performance. In fact, it may even be the case, that the optimization problems which come from the formulation of a local search are not general enough for the no free lunch theorem to apply, and there may, in fact, be an optimal algorithm for them.

The rest of the paper is organized as follows: in the next section, we start by the overview of related work in the field of surrogate based multi-objective optimization with the focus on the different local search strategies employed. Further, we describe two of our recent multi-objective optimizers, which we use as a testing ground for different local search techniques. A brief overview of such techniques is provided in a latter section. Next, the experiments we performed are described and the results are commented on. Based on these results, we recommend useful local search strategies and discuss the decision which should be made while choosing local search heuristics for the use in conjunction with a surrogate-based multi-objective optimizers.

Martin Pilát is with the Charles University in Prague, Faculty of Mathematics and Physics, Malostranské náměstí 25, 118 00, Prague, Czech Republic. (email: Martin.Pilat@mff.cuni.cz)

Roman Neruda is with the Institute of Computer Science, Academy of Sciences of the Czech Republic, Pod Vodárenskou věží 271/2, 182 07, Prague, Czech Republic. (email: roman@cs.cas.cz)

Roman Neruda was supported by the Ministry of Education of the Czech Republic project COST LD 13002.

978-1-4799-1488-3/14/$31.00 ©2014 IEEE

II. Related Work

Many surrogate-based multi-objective evolutionary algorithms were proposed in the past. Among the first such algorithms are two versions of the surrogate based NSGA-II [4], [5]. Although there are differences in the ways the models are trained and the types of models used, both of these algorithms switch between two phases – one, in which the real objectives are optimized, and one in which the surrogate models, built for each of the objectives separately, are optimized using NSGA-II.

A Gaussian process [7] based version of MOEA/D [8], called MOEA/D-EGO, was also proposed [9]. MOEA/D uses the idea of decomposition to solve multi-objective optimization problems. It transforms the multi-objective problem into several single-objective problems by using weighted sums of the objectives. In MOEA/D-EGO, a Gaussian process model is build for each of the objectives and models for the single-objective sub-problems are derived from them. The models are then optimized using MOEA/D with the goal to find the largest expected improvement.

Loshchilov *et al*. [2], [3] also proposed two surrogate-based algorithm. In this case, the surrogate models are aggregated – there is only one surrogate, which predicts the quality of the individual as a whole, instead of predicting the values of each objective separately. In the earlier version, the surrogate used ideas from OneClass SVM and support vector regression [10] to characterize the areas of the search space which can contain new non-dominated individuals. In the later version, the model was based on ranking SVM [11] and was trained to predict the dominance relation. In both cases, a larger number of individuals was generated and then only the best of them (according to the model) were evaluated with the real objectives.

Finally, we have also proposed a surrogate-based multi-objective algorithm with an aggregate model – ASM-MOMA [12]. The model in the algorithm is based on the distance to the currently non-dominated set. Individuals in this algorithm are locally optimized using another single-objective evolutionary algorithm. More details are presented in the next section, as this is one of the algorithms we use for comparison of the different local search strategies.

III. Surrogate-based multi-objective algorithms

In this section, we describe two different multi-objective surrogate-based evolutionary algorithms. One of them is the multi-objective memetic algorithm with aggregate surrogate model (ASM-MOMA), the other is a multi-objective memetic algorithm, which is based on the optimization of the hyper-volume contribution of each individual with respect to its neighbors on the non-dominated front (HO-MOMA) (the neighbors are the two individuals on the Pareto front of a problem with two objectives which are closest to the left and right from the optimized individual, see bellow for more rigorous definition). The main drawback of HO-MOMA is the fact, that it only works for bi-objective problems due to the local search fitness construction.

Both the algorithms can be explained as a modification of an existing multi-objective evolutionary algorithm. We use NSGA-II here, but any other multi-objective algorithm can be used. Both of the algorithms we used during experiments use single-objective optimization during the local search, however, they differ in the way the single-objective function is constructed.

In ASM-MOMA, only one model is used. When the training set is created, the individuals in the non-dominated set are assigned a value of 0, while the other individuals are assigned a value equal to the negative value of their distance from the non-dominated set. The model is trained on this training set, and as such, it discriminates between the areas of the search space, which have been already explored (they should have negative values), and the yet unexplored areas (they should have positive values thanks to generalization of the model), the line between these two areas should correspond to the current non-dominated set (and the model should predict 0 for them).

In each generation a quarter of individuals in the population are improved using a local search procedure, which starts with the selected individual and tries to maximize the value of the model. The best found individual replaces the original one in the population.

The surrogate model in ASM-MOMA is rather simple, and therefore it is easily trained. We experimented with different machine learning techniques to find one, which would provide the best results during the optimization, and we found out that support vector regression based models provide the most robust results with respect to the convergence rate and the quality of the resulting non-dominated sets.

The other algorithm we used for experiments (HO-MOMA) is currently under development, however, it can be used for these experiments (although, it is possible, some aspects of the algorithm will change slightly before it will be published). In fact, the comparison of the local search techniques was partially inspired by the need to choose one for this algorithm.

In HO-MOMA the construction of the model is different. First, based on previously evaluated values of the objective function, a surrogate model is build for each of the objectives. Let us denote the individuals in the sorted (by the values of the first objective) non-dominated set as $x_1, \ldots, x_n$. For each individual x_i a single-objective the following function is optimized:

$$\max_x \qquad (f_2(x_{i-1}) - \hat{f}_2(x))(f_1(x_{i+1}) - \hat{f}_1(x)),$$

where f_1 and f_2 are the two objective functions and $\hat{f}_1$ and $\hat{f}_2$ are their models. A negative value is taken in cases x where both $f_1(x) > f_1(x_{i+1})$ and $f_2(x) > f_2(x_{i-1})$, so that the equation is positive only if the point x contributes to the hyper-volume of the non-dominated points. After the local search completes, the individual x_i is replaced by the vector, which maximizes the above equation according to the model – i.e. it has the largest hyper-volume contribution.

The algorithm runs in two phases which switch after each generation. In one of them, one iteration of NSGA-II is performed, in the other, the local search is executed for individuals in the non-dominated front.

The fact that the hyper-volume contribution is computed with respect to the neighbors of the individual ensures that the optimized individuals will be well spread over the whole current non-dominated set.

IV. Local search algorithms

We compare different local search algorithms in this work. Most of them are well-known and often used, but for the sake of self-containedness and also to define them more precisely (which is especially needed in the case of the simple evolutionary algorithm), we briefly present them here.

A. Random search

Random search is the most simple search heuristic. It is quite often used in algorithms which tend to generate much larger populations of pre-offspring and then use a surrogate model to select only the promising ones. It generates a number of individuals around the optimized one, and the one with the best fitness (according to a surrogate model) is then selected as the new individual.

B. Gradient search

Gradient search is one of the most basic continuous optimization algorithms. As it requires the gradient of the optimized function, it is generally not usable for black-box optimization. However, when surrogate models are used, it is possible to obtain the gradient of the model, either numerically, or, in some cases, even analytically. In this work, we choose the numerical way which is more general.

The gradient search algorithm works by iteratively moving the individual towards the areas with better fitness – it uses the gradient information to find such areas. For maximization, the creation of the individual x_{t+1} from individual x_t can be expressed as:

$$x_{t+1} = x_t + \lambda \nabla f(x_t),$$

where λ is the step size parameter, and $\nabla f(x_t)$ is the gradient of the function (model) f in the point x_t.

This step is repeated for several iterations and the individual from the last iteration is used as the new, optimized one.

C. Simple evolutionary algorithm

Evolutionary algorithms have a large number of variants. In this work, by a simple evolutionary algorithm we denote an algorithm which is basically NSGA-II [13] with only a single objective. It uses the same operators – i.e. SBX crossover [14] and polynomial mutation [15]. The environmental selection merges the populations of parents and offspring, sort them according to the (surrogate-based) fitness function, and selects the best half of the individuals to fill the parent population in the next generation. The operators of the algorithm are separable – each of the variables in the individual are optimized independently of the others. This may lead to problems when optimizing non-separable functions.

We used such a simple evolutionary algorithm in our original work with ASM-MOMA and its later variants.

V. Experimental setup

The algorithms are evaluated using the well-known ZDT [16] and WFG [17] benchmark suites.

In the experiments, the multi-objective algorithms use 100 individuals in the population and were given the computational budget of 30,000 objective function evaluations. They use the SBX crossover with probability 0.9, and polynomial mutation with probability $1/N$, where N is the number of variables for each of the problems ($N = 30$ for ZDT1, ZDT2, and ZDT3, $N = 10$ for ZDT4 and ZDT6, and $N = 24$ for all the WFG problems – 4 of these are position related and 20 are distance related parameters). The algorithms use NSGA-II selection with the hyper-volume contribution as the secondary sorting criterion.

The SVM-based surrogate model was based on support vector regression from the Weka [18] toolbox. It uses the Gaussian kernel with bandwidth parameter 0.01. The complexity parameter was set to $C = 1$.

We also use a "perfect model" in some of the experiments – it is, in fact, not a model at all. It uses the real objective functions instead of any model. However, these evaluations are not included in the overall evaluation count. In a sense, it presents and ideal model, one which has zero error and learned the objectives perfectly. It is useful as a comparison to the SVM model and also allows us to evaluate the performance of the local search algorithms from a different – more theoretical – point of view. Moreover, it may be also interesting to observe the differences in the performance of the SVM model and the perfect model.

The random search samples 1,000 individuals around the optimized one – a number from normal distribution with the standard deviation equal to $\sigma = 0.01(u_i - l_i)$, is added to the i-th variable, where u_i and l_i are the lower and upper bound of the i-th variable respectively.

The gradient search adds $0.01\nabla f$ to the values of the variables in each iteration, where ∇f is the (numerically) computed gradient of the fitness. It is executed for 10 iterations. We experimented with larger number of iterations in preliminary test, but we did not observe any significant difference.

The simple evolutionary algorithm uses the SBX crossover and polynomial mutation with the same probabilities as the external algorithm. It has a population of 50 individuals, generated around the optimized one by cloning the individual and applying the polynomial mutation 100 times to each of the clones. It runs for 30 generations.

All the local search algorithms use only surrogate model evaluations and no evaluations of the real objective functions.

The algorithms are evaluated using the ΔH metric – it expresses the difference between the hyper-volume of the

μ-optimal distribution [19] for $\mu = 100$ and the hyper-volume found by the algorithm. The value of the metric was recorded after each generation and used to create the graphs in the following section. We executed the algorithms for 15 independent runs for each of the configurations and provide the median of the performance, together with the first and third quartile in the graphs presented in this paper. We obtained a large number of data during the experiments, and all the data would not fit in the paper due to the space limitation. However, all the data is available on the author's web page as supplementary material[1]. The supplementary material also contains numerical and statistical comparison of the different algorithms which is evaluated with the Mann-Whitney U-test (Wilcoxon rank-sum test).

VI. Results and discussion

In all the graphs presented in the following sections the names of the algorithm consist of two parts – the first part expresses the type of algorithm (ASM for ASM-MOMA, and svmHO and pmHO for HO-MOMA with SVM based model and perfect model respectively). The second part, separated by a dash from the first part, denotes the type of local search ("Grad" for gradient search, "Rand" for random search, and "EA" for simple evolutionary algorithm). When we say, some algorithm is significantly better than another one, it means the p-value of the Mann-Whitney U-test (Wilcoxon rank-sum test) is less than 0.001.

A. ASM-MOMA

First, there is almost no difference among the different local search strategies when employed in ASM-MOMA (cf. Figure 1). In fact, we have not observed any statistically significant difference among the different local search algorithms (their performance was statistically evaluated after 1,000; 5,000; 10,000; 20,000; and 30,000 evaluations of the objectives). The small differences among the algorithms is mainly due to the fact that the surrogate model in ASM-MOMA – based only on the distance of points from the non-dominated set – is rather simple, without local optima. Thus, it is easily optimized by all the local search algorithms.

B. HO-MOMA with SVM model

For HO-MOMA, with its more complicated, hyper-volume contribution based model, the situation is different. The differences among the local search heuristics (see Figure 3) are much larger and thus the choice of a good local search heuristic is more challenging

For the ZDT problems, we can see that the gradient-based local search is able to converge quicker than the other local search heuristics in most cases. The ZDT problems are quite easy, with one of the objectives being a linear function of the first variable (except for ZDT6, where the objective is more complicated). This of course translates to simpler models for the objectives and thus easier optimization of such models. Therefore, the gradient search is significantly better in the first few thousands of evaluations than the other local searches on the ZDT benchmarks. ZDT4 is the only exception, in this case, the performance of all the optimizers is almost the same, none of them is able to converge in the 30,000 objective evaluations the algorithms were given. The evolutionary algorithm has the slowest convergence rate in this case. We can also see an interesting behavior for ZDT6, here, the evolutionary algorithm is the only one, which is not able to converge to values of $\Delta\mathrm{H}$ lower than 0.01. All the other local search algorithms find better approximations. It is possibly due to the fact that the algorithm uses the same operators as the external NSGA-II algorithm, whereas the other algorithms work differently. The different operators, or local search techniques, may hybridize better with NSGA-II and are able to better exploit the information provided by the model. The value of approx. 0.01 (found by the EA local search), is also the value towards which the NSGA-II without surrogate converges.

The WFG problems are much more complicated and it can also be observed in the results. The gradient search is no longer the best among the tested algorithms, in fact, it struggles to optimize the more complicated objective functions with lots of local optima, and is often the slowest of the algorithms, often even slower than the random search. An interesting situation is observed for the WFG2 function, where the EA local search performs much better than the other algorithms (EA is able to obtain almost $\Delta\mathrm{H} = 0.1$, while the other algorithms are around $\Delta\mathrm{H} = 1$).

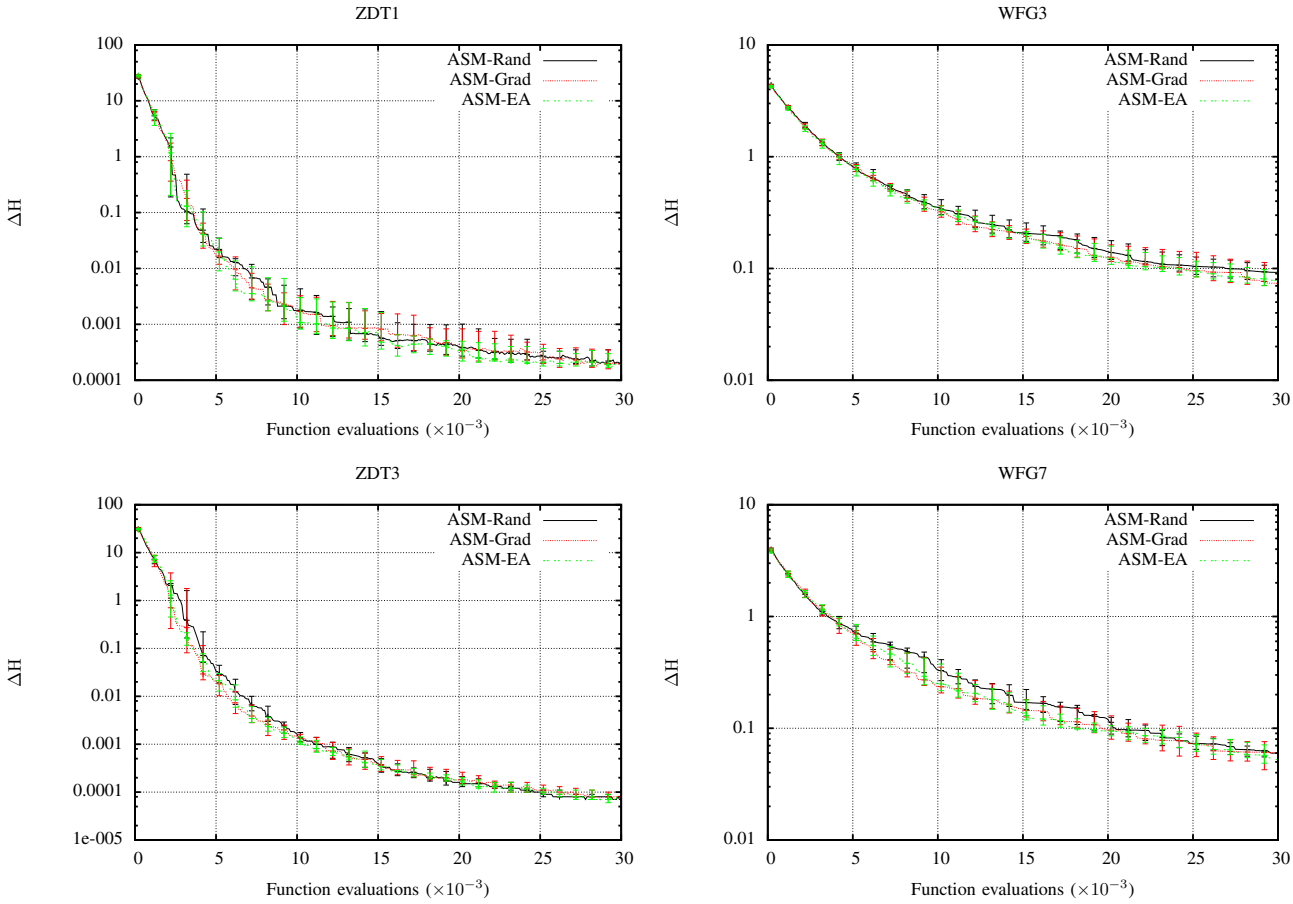
C. HO-MOMA with perfect model

The version of HO-MOMA with perfect model is rather a theoretical one, as the perfect model uses the evaluations of the real objectives (which are not included in the overall count). However, the results (see Figure 3) are interesting and show how the choice of model can affect the performance. One could expect that the results with the perfect model should be better than the ones with the SVM model, but this is not always the case. In some cases, the results of the SVM models are better.

For ZDT problems, the results of the perfect model roughly correspond to the results with the SVM models. The convergence is little slower, but the order of the local search algorithms stays the same. Gradient search is again the fastest of the tested algorithms, especially in the earlier phases of the evolution, and all the local search algorithms converge to similar values of $\Delta\mathrm{H}$. In this case, even the EA of ZDT6 is able to find the same quality of resulting Pareto sets as the other local search algorithms (although we observe slower convergence and rather large plateau around the value of $\Delta\mathrm{H} = 0.01$ – the one the algorithms was able to obtain with the SVM model). For ZDT4, the EA is the fastest of the algorithms during the first 15,000 function evaluations, where the gradient search starts to win (before they converge to the same value around the 25,000 mark). The random search is the slowest of the algorithms in all cases, except ZDT6 as already mentioned.

[1] http://martinpilat.com/images/pdf/cec2014-supplementary.pdf

Figure 1. The performance of the different local search heuristics when used with ASM-MOMA. Median of fifteen runs together with first and third quartile.



For the WFG benchmark, the EA local search is in almost all cases the best among the local searches. For some of the problems (WFG1, WFG3, WFG4, WFG7, WFG9 and partially also WFG5) the difference between the EA search and the others is rather large. Most of these problems are separable (WFG1, WFG4, WFG5, and WFG7) and the evolutionary algorithm uses operators which can exploit the separability. For WFG3 the convergence of EA is faster mainly in the beginning, and the difference between the algorithms stays almost the same later in the evolution – it has approximately 10 times lower value of ΔH for numbers of evaluations between 5,000 and 30,000.

As already mentioned, the situation on WFG2 is the most interesting, mainly when compared with the SVM based models. For the SVM based models, the EA local search is the best among the tested, however, with the perfect model, the evolutionary algorithm does not work better than the other local search techniques. This shows, that the surrogate modeling can improve the convergence rate of the algorithm even when compared with the perfect model. The reasons for this performance difference should be further studied in the future. In WFG2, one of the objectives is uni-modal, while the other is multi-modal, both are non-separable functions. It seems the SVM models are able to grasp the global structure of the multi-modal WFG2 objective better and make it more easily exploitable by the EA.

VII. Conclusions

We compared three widely used local search algorithms – random search, gradient search, and simple evolutionary algorithm. These algorithms were employed as local search in two surrogate based multi-objective algorithms.

The results indicate that for algorithms with simpler models (ASM-MOMA) the choice of the local search does not affect the result – every local search algorithms is able to exploit the surrogate model. However, with more complicated local surrogate models (HO-MOMA) the local search algorithms can greatly affect the performance. Therefore, the choice of the local search (and also of the surrogate model) shall be carefully considered when designing new surrogate-based multi-objective algorithms, not only with regard to the type of model used but also with regard to the complexity of optimized functions.

Moreover, we have shown that surrogate models can improve the convergence rate of the algorithm even more than a "perfect model" – a "model" which uses the objectives directly. The models can grasp the global structure of the

Figure 2. The performance of the different local search heuristics with HO-MOMA with SVM-based model. Median of fifteen runs, together with the first and third quartile.

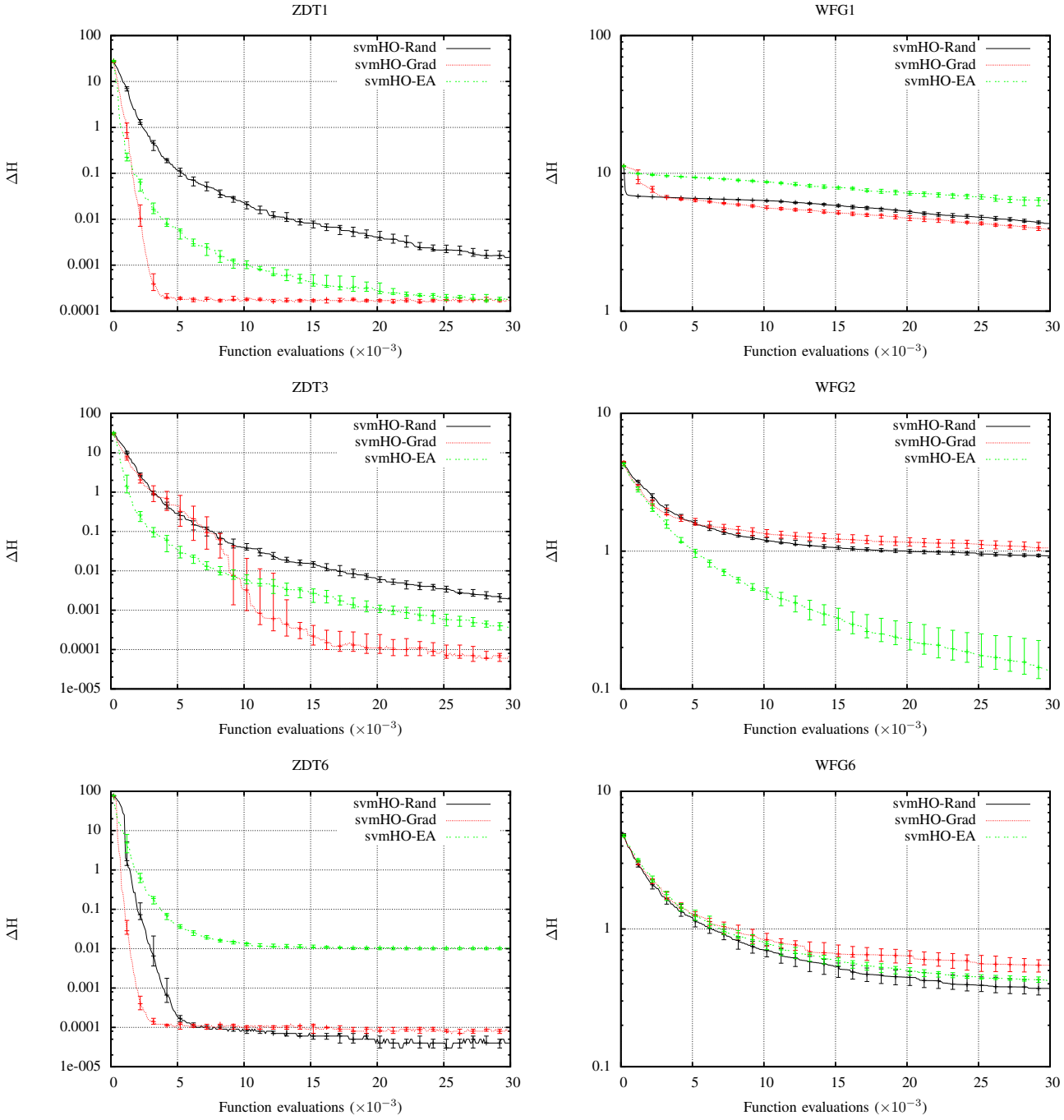
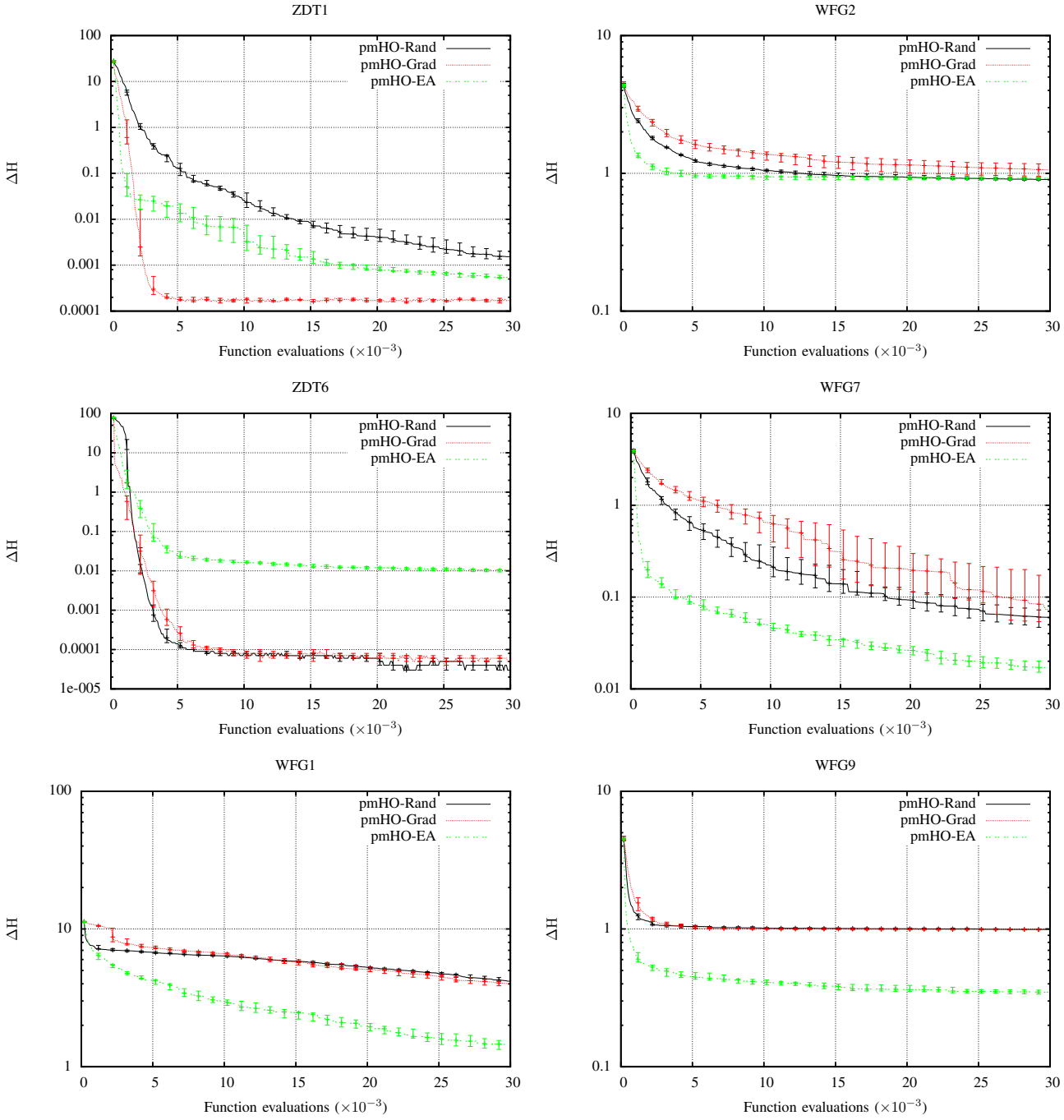


Figure 3. The performance of the different local search heuristics with HO-MOMA with SVM-based model. Median of fifteen runs, together with the first and third quartile.



problem and remove some of the local optima, thus making the optimization easier.

In the future, these results can be used in the design of new surrogate-based multi-objective algorithms. They can also lead to the improvement of some of the existing algorithms – i.e. if the local search algorithm they use is replaced by a more suitable one.

We would like to work on ways, which would provide the possibility for automated selection of local search algorithm in run-time. This selection might be based on the evaluation of the different local search algorithms and their performance while solving the problem at hand, e.g. each of the local searches can be used in some generations, with the ones with faster improvement being used more often.

REFERENCES

[1] M. Pilát and R. Neruda, "Aggregate meta-models for evolutionary multiobjective and many-objective optimization," *Neurocomputing*, vol. 116, pp. 392–402, 2013.

[2] I. Loshchilov, M. Schoenauer, and M. Sebag, "A mono surrogate for multiobjective optimization.," in *GECCO* (M. Pelikan and J. Branke, eds.), pp. 471–478, ACM, 2010.

[3] I. Loshchilov, M. Schoenauer, and M. Sebag, "Dominance-based pareto-surrogate for multi-objective optimization.," in *SEAL* (K. Deb, A. Bhattacharya, N. Chakraborti, P. Chakroborty, S. Das, J. Dutta, S. K. Gupta, A. Jain, V. Aggarwal, J. Branke, S. J. Louis, and K. C. Tan, eds.), vol. 6457 of *Lecture Notes in Computer Science*, pp. 230–239, Springer, 2010.

[4] I. Voutchkov and A. Keane, "Multiobjective optimization using surrogates," in *Proceedings of the 7th International Conference on Adaptive Computing in Design and Manufacture*, pp. 167–175, The M.C.Escher Company, April 2006.

[5] P. K. Nain and K. Deb, "A multi-objective optimization procedure with successive approximate models," *KanGAL report*, no. 2005002, 2005.

[6] D. Wolpert and W. Macready, "No free lunch theorems for optimization," *Evolutionary Computation, IEEE Transactions on*, vol. 1, pp. 67–82, Apr 1997.

[7] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005.

[8] Q. Zhang and H. Li, "Moea/d: A multiobjective evolutionary algorithm based on decomposition," *Evolutionary Computation, IEEE Transactions on*, vol. 11, no. 6, pp. 712–731, 2007.

[9] Q. Zhang, H. Li, D. Maringer, and E. Tsang, "Moea/d with nbi-style tchebycheff approach for portfolio management," in *Evolutionary Computation (CEC), 2010 IEEE Congress on*, pp. 1–8, 2010.

[10] H. Drucker, C. J. C. Burges, L. Kaufman, A. J. Smola, and V. Vapnik, "Support vector regression machines," in *NIPS* (M. Mozer, M. I. Jordan, and T. Petsche, eds.), pp. 155–161, MIT Press, 1996.

[11] T. Joachims, "A support vector method for multivariate performance measures," in *Proceedings of the 22nd international conference on Machine learning*, ICML '05, (New York, NY, USA), pp. 377–384, ACM, 2005.

[12] M. Pilát and R. Neruda, "Multi-objectivization and surrogate modelling for neural network hyper-parameters tuning," in *ICIC (3)* (D.-S. Huang, P. Gupta, L. Wang, and M. M. Gromiha, eds.), vol. 375 of *Communications in Computer and Information Science*, pp. 61–66, Springer, 2013.

[13] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *Evolutionary Computation, IEEE Transactions on*, vol. 6, no. 2, pp. 182–197, 2002.

[14] K. Deb and R. B. Agrawal, "Simulated binary crossover for continuous search space," *Complex Systems*, vol. 9, pp. 1–34, 1994.

[15] K. Deb and M. Goyal, "A combined genetic adaptive search (geneas) for engineering design," *Computer Science and Informatics*, vol. 26, no. 4, pp. 30–45, 1996.

[16] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: Empirical results," *Evol. Comput.*, vol. 8, pp. 173–195, June 2000.

[17] S. Huband, P. Hingston, L. Barone, and L. While, "A review of multiobjective test problems and a scalable test problem toolkit," *Evolutionary Computation, IEEE Transactions on*, vol. 10, no. 5, pp. 477–506, 2006.

[18] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: an update," *SIGKDD Explor. Newsl.*, vol. 11, pp. 10–18, Nov. 2009.

[19] A. Auger, J. Bader, D. Brockhoff, and E. Zitzler, "Theory of the hypervolume indicator: optimal μ-distributions and the choice of the reference point," in *Proceedings of the tenth ACM SIGEVO workshop on Foundations of genetic algorithms*, FOGA '09, (New York, NY, USA), pp. 87–102, ACM, 2009.