Local Meta-models for ASM-MOMA

Martin Pilát[1,2] and Roman Neruda[2]

[1] Department of Theoretical Computer Science and Mathematical Logic, Faculty of Mathematics and Physics, Charles University in Prague, Malostranské náměstí 25, Prague, Czech Republic

[2] Institute of Computer Science, Academy of Sciences of the Czech Republic, Pod Vodárenskou věží 2, Prague 8, Czech Republic

Martin.Pilat@mff.cuni.cz, roman@cs.cas.cz

Abstract. Evolutionary algorithms generally require a large number of objective function evaluations which can be costly in practice. These evaluations can be replaced by evaluations of a cheaper meta-model of the objective functions. In this paper we describe a multiobjective memetic algorithm utilizing local distance based meta-models. This algorithm is evaluated and compared to standard multiobjective evolutionary algorithms as well as a similar algorithm with a global meta-model. The number of objective function evaluations is considered, and also the conditions under which the algorithm actually helps to reduce the time needed to find a solution are analyzed.

Keywords: Multiobjective optimization, meta-model, evolutionary algorithm.

1 Introduction

Many real life optimization tasks require optimizing multiple conflicting objectives at once. It has been shown and widely accepted that multiobjective evolutionary algorithms (MOEA) are among the best methods for multiobjective optimization. In the past years several multiobjective evolutionary algorithms [3,12,1] were proposed and used to deal with these problems. However, most of them require lots of evaluations of each objective function, which makes them problematic to use for solving real life problems. These problems may have complex objective functions whose evaluations are expensive (either in terms of time or money).

The use of the meta-models aims at lowering the number of objective function evaluations which are needed to obtain the final solution. The meta-model is a simplified and cheaper approximation of the real objective function. Meta-models can be used in several ways to augment the multiobjective evolutionary algorithms. In one of the first approaches [10] its authors used the NSGA-II [3] and replaced the objective functions with their meta-models. In [7] and [8] authors describe an aggregate meta-model based on various SVM architectures. Although the memetic variant is also possible in multiobjective setting, only a few references were found in the literature which deal with meta-model assisted multiobjective memetic algorithms [5].

The paper is organized as follows: In the next section 2. The tests and their results are described in sections 3 and 4. Section 5 concludes the paper and provides ideas for future research.

D.-S. Huang et al. (Eds.): ICIC 2011, LNBI 6840, pp. 79–84, 2012.

2 Algorithm Description

In one of our previous papers [9] we proposed a multiobjective memetic algorithm with aggregate meta-model (ASM-MOMA). This algorithm was able to reduce the number of required evaluations of the objective functions by the factor of 5 to 10 on most problems. ASM-MOMA uses a single global meta-model trained after each generation as a fitness function during the local search.

In this paper, we propose a new variant of ASM-MOMA with local models instead of a single global one. We call this variant LAMM-MMA. The main difference between LAMM-MMA and other multiobjective evolutionary algorithms is the addition of a special memetic operator, which performs local search on some of the newly generated individuals (the generation of the new individuals is handled by the respective MOEA to which is this operator added). The operator uses the meta-model constructed based on previously evaluated points in the decision space, for which the values of objective functions are known. The meta-model is trained to predict the distance to the currently known Pareto front. Moreover, as an addition to ASM-MOMA, in LAMM-MMA the points do not not have the same weight, as those that are closer to the locally optimized one are considered more important during the model building phase, see Equation 1 for details.

The main idea is that points closer to the known Pareto front are more interesting during the run of the algorithm and the memetic operator moves the individuals closer to the Pareto front. The purpose of the meta-model is not to precisely predict the value but rather provide a general direction in which the memetic search should proceed. To obtain a training set for the meta-models we also added an external archive of individuals with known objective values. This archive is updated after each generation when new individuals are added and at the same time the archive is truncated to ensure it does not grow indefinitely – random individuals are removed to match the limit on the number of individuals, see [9] for analysis of this approach.

The following sections detail the important parts of the algorithm. The main loop is essentially a generic MOEA with added memetic operator. We train a dedicated model for each individual I which shall be locally optimized by the memetic operator. For such an individual I we create a weighted training set

$$T_I = \left\{ \langle (x_i, y_i), w_i \rangle | y_i = -d(x_i, P), w_i = \frac{1}{1 + \lambda d(x_i, I)} \right\} \tag{1}$$

where $d(x, y)$ is the Euclidean distance of individuals x and y in the decision space, P is the set of non-dominated individuals in the archive and $d(x, P)$ is the distance of individual x to the closest point in the set P. λ is a parameter which controls the locality of the model, larger values of λ lead to more local model, whereas lower values lead to more global one.

The points which are closer to the individual I are more important during the training of the model. This distance weighting adds some locality to the models trained for each individual. The training set is constructed in such a way, that for the individuals closer to the currently known Pareto front the meta-model should return larger values. This fact is used during the local search phase (which uses the meta-model as a fitness function).

Table 1. Times needed for training and evaluation of selected meta-models, in seconds

Model	Training (T_t)	Evaluation (T_m)
Linear regression	0.142	8.46×10^{-7}
Support vector regression	0.328	7.14×10^{-7}
Multilayer perceptron	3.75	1.80×10^{-5}

In the local search phase we use another evolutionary algorithm (this time it is only a single objective one) to find better points in the surroundings of each individual. The algorithm runs only for a few generations and it uses only meta-model evaluations. The newly found individuals are placed back to the population. During the initialization of the local search the individual which should be optimized is inserted in the initial population and its variables are perturbed to create the rest of the initial population.

The algorithm uses quite large number of meta-model evaluations and even trainings. This might lead to significant overhead. To find out how large this overhead is, we run a few benchmarks (archive size/training set size of 400 individuals, Intel Core i7 920 (2.87Ghz) processor and 6GB RAM). Table 2 shows the results. We can see that the evaluations are faster than the training by several orders of magnitude and that each training takes only a fraction of a second. Even if there are 100 trainings per generation, it would mean an overhead of roughly 15 to 30 seconds per generation, which still might be faster than a single evaluation of the real objective function. Therefore the overhead of the training and evaluation is easily compensated by the reduced evaluations of the objective functions.

3 Test Setup

We tested our approach on the widely used ZDT [11] benchmark problems. These problems are all two dimensional, and we used 15 variables for each of them. In the local search phase we used various meta-models: namely multilayer perceptron, support vector regression, and linear regression. All the models use default parameters from the Weka framework [6] (which we used to run the experiments).

Table 2. Parameters of the multiobjective algorithm

Parameter	MOEA value	Local search value
Stopping criterion	50,000 objective evaluations	30 generations
Population size	50	50
Crossover operator	SBX	SBX
Crossover probability	0.8	0.8
Mutation operator	Polynomial	Polynomial
Mutation probability	0.1	0.2
Archive size	400	–
Memetic operator probability	0.25	–
Meta-model locality parameter λ	–	1

See Table 2 for the parameters of the main multiobjective algorithm and the internal single-objective algorithm.

As the base multiobjective evolutionary algorithm we used the NSGA-II and ϵ-IBEA with Simulated Binary Crossover [2] and Polynomial Mutation [4]. In the local search phase we used a simple single objective evolutionary algorithm with the same operators and the meta model as the fitness function.

To compare the results we use a measure we call H_{ratio}, it is defined as the $H_{ratio} = \frac{H_{real}}{H_{optimal}}$, where H_{real} is the hypervolume of the dominated space attained by the algorithm and $H_{optimal}$ is the hypervolume of the real Pareto set of the solutions. As the Pareto set is known for all the ZDT problems, we can compute this number directly. We use the vector $\mathbf{2} = (2, 2)$ as the reference point in the hypervolume computation. All points that do not dominate the reference point are excluded from the hypervolume computation. We compare the median number of function evaluations needed to attain the H_{ratio} of 0.5, 0.75, 0.9, 0.95, and 0.99 respectively.

4 Results

Table 3 shows the results of our algorithm compared to original ϵ-IBEA and ASM-MOMA. IBEA denotes the original ϵ-IBEA. LR, SVM, and MLP stands for the model used: linear regression, support vector regression and multilayer perceptron respectively. G denotes the single global model of ASM-MOMA and L stands for the local models as described in this paper.

The numbers in the table represent the median number of objective function evaluations needed to reach the specified H_{ratio} value. Twenty runs for each configuration were made.

From the results, we can see that the global models generally significantly decrease the number of required function evaluations, and the local models are even better than the global ones. Generally, linear regression gives better results than support vector regression and multilayer perceptrons. It probably creates simpler models which indicate the right general direction in which the local search should proceed. Moreover, we can see that the results of local models are almost always better than those of a single global model, thus we recommend using the faster models, i.e. linear regression or support vector regression instead of multilayer perceptrons.

On ZDT1 the global model was able to reduce the number of function evaluations to reach the $H_{ratio} = 0.95$ by a factor of 6.8 (LR) and the local model reduced it further, yielding the reduction factor of 7.3 for LR and even 7.7 with the SVM. For the $H_{ratio} = 0.99$ the reductions are not that large, but we can still see the number decreased by the factor of almost 4. In this case, local models did not improve the result.

On ZDT2 the results improved largely even for the $H_{ratio} = 0.99$. ASM-MOMA reduced the required number of objective function evaluations 8.4 times (SVM), while LAMM-MMA was able to reduce it almost 9 times with the same model and 9.6 times with the LR as the model.

Reaching the $H_{ratio} = 0.99$ was a problem for the original ϵ-IBEA on ZDT3, but both ASM-MOMA and LAMM-MMA were much more successful, both reducing the

Table 3. Median number of function evaluations needed to reach the specified H_{ratio} on ZDT1 test problem

	ZDT1					ZDT2				
H_{ratio}	0.5	0.75	0.9	0.95	0.99	0.5	0.75	0.9	0.95	0.99
IBEA	7400	13750	18200	20000	25550	750	2050	5150	7800	13000
IBEA-LR-G	1450	2500	2800	2950	7450	350	550	750	900	1650
IBEA-SVM-G	1400	2050	2700	3100	**6850**	350	550	850	1050	1550
IBEA-MLP-G	1800	2550	4000	4600	10100	450	650	950	1200	2700
IBEA-LR-L	**1300**	1900	2400	2750	7500	**300**	**500**	**700**	**850**	**1350**
IBEA-SVM-L	1350	1900	**2350**	**2600**	7100	350	550	800	1000	1450
IBEA-MLP-L	1400	**1850**	2450	3250	9650	350	550	750	900	1400
	ZDT3					**ZDT6**				
H_{ratio}	0.5	0.75	0.9	0.95	0.99	0.5	0.75	0.9	0.95	0.99
IBEA	650	1550	5400	8150	33350	10300	13650	18400	23150	34050
IBEA-LR-G	**350**	550	850	950	**1300**	3050	**6500**	13400	**17600**	32100
IBEA-SVM-G	**350**	550	850	1000	**1300**	**3000**	7250	14100	19250	34150
IBEA-MLP-G	450	800	1100	1250	1800	3500	7250	13250	18900	32450
IBEA-LR-L	**350**	**450**	**750**	**900**	**1300**	3050	6850	13050	18750	**31400**
IBEA-SVM-L	400	650	850	1050	1450	**3000**	**6500**	**12650**	17850	32550
IBEA-MLP-L	400	650	950	1150	1600	3400	7050	13300	18200	32950

number of evaluations over 25 times. For the $H_{ratio} = 0.95$ (which is not that difficult for ϵ-IBEA) we can see again reduction factors of 8.5 and 9 for ASM-MOMA and LAMM-MMA respectively (LR in both cases).

ZDT6, as in our previous paper [9], proved to be the most difficult problem. Although we can see reductions by the factor of 3.5 for the $H_{ratio} = 0.5$, this factor drops and there are only slight reductions of 6% for the $H_{ratio} = 0.99$. Note that LAMM-MMA again provided better reductions for this value, even though, the difference is not very large. Dealing with the difficulty of this problem is a motivation for further research.

5 Conclusions

In this paper we presented a memetic evolutionary algorithm for multiobjective optimization with local meta-models. We showed that the local models give better results than a single global model, usually reducing the number of needed function evaluations by another 10%. Although this difference may seem rather small it may greatly reduce the associated costs in practical tasks. We also showed that the algorithm is usable even for problems with quite simple objective functions, which take only milliseconds to evaluate, thus making it more widely usable. However, some problems are still difficult to solve with LAMM-MMA, and these provide the motivation for further research. Another open question is whether real life problems are among those easily solvable, or not.

We will continue the work on memetic multiobjective algorithms with aggregate meta-models. One of the goals is the reduction of the number of times the model is trained which is a problem especially for more expensive local models. These are trained multiple times in each generation. One possibility could be to cluster the individuals before the model is constructed and create a single local model for all the individuals in the cluster. Another open question is the effect of the degree of locality (represented by the λ parameter) on the evolution convergence speed and the possibility to change this parameter adaptively.

Acknowledgments. This research has been supported by the Ministry of Education of the Czech Republic under project no. OC10047, and by Czech Science Foundation project no. 201/09/H057.

References

1. Bader, J., Zitzler, E.: HypE: An Algorithm for Fast Hypervolume-Based Many-Objective Optimization. TIK Report 286, Computer Engineering and Networks Laboratory (TIK), ETH Zurich (November 2008)
2. Deb, K., Agrawal, R.B.: Simulated binary crossover for continuous search space. Complex Systems 9, 115–148 (1995)
3. Deb, K., Agrawal, S., Pratap, A., Meyarivan, T.: A fast elitist non-dominated sorting genetic algorithm for multi-objective optimisation: NSGA-II. In: Deb, K., Rudolph, G., Lutton, E., Merelo, J.J., Schoenauer, M., Schwefel, H.-P., Yao, X. (eds.) PPSN 2000. LNCS, vol. 1917, pp. 849–858. Springer, Heidelberg (2000)
4. Deb, K., Goyal, M.: A combined genetic adaptive search (geneas) for engineering design. Computer Science and Informatics 26, 30–45 (1996)
5. Georgopoulou, C., Giannakoglou, K.: Multiobjective metamodel-assisted memetic algorithms. Multiobjective Memetic Algorithms, 153–181 (2009)
6. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA data mining software: an update. SIGKDD Explorations 11(1), 10–18 (2009), `http://dx.doi.org/10.1145/1656274.1656278`
7. Loshchilov, I., Schoenauer, M., Sebag, M.: A mono surrogate for multiobjective optimization. In: Pelikan, M., Branke, J. (eds.) GECCO, pp. 471–478. ACM, New York (2010)
8. Loshchilov, I., Schoenauer, M., Sebag, M.: Dominance-based pareto-surrogate for multiobjective optimization. In: Deb, K., Bhattacharya, A., Chakraborti, N., Chakroborty, P., Das, S., Dutta, J., Gupta, S.K., Jain, A., Aggarwal, V., Branke, J., Louis, S.J., Tan, K.C. (eds.) SEAL 2010. LNCS, vol. 6457, pp. 230–239. Springer, Heidelberg (2010)
9. Pilát, M., Neruda, R.: ASM-MOMA: Multiobjective memetic algorithm with aggregate surrogate model. In: IEEE Congress on Evolutionary Computation. IEEE, Los Alamitos (to appear, 2011)
10. Voutchkov, I., Keane, A.: Multiobjective optimization using surrogates. Presented on Adaptive Computing in Design and Manufacture, ACDM 2006 (2006)
11. Zitzler, E., Deb, K., Thiele, L.: Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. Evolutionary Computation 8(2), 173–195 (2000)
12. Zitzler, E., Künzli, S.: Indicator-Based Selection in Multiobjective Search. In: Yao, X., Burke, E.K., Lozano, J.A., Smith, J., Merelo-Guervós, J.J., Bullinaria, J.A., Rowe, J.E., Tiňo, P., Kabán, A., Schwefel, H.-P. (eds.) PPSN 2004. LNCS, vol. 3242, pp. 832–842. Springer, Heidelberg (2004)