A parallel, hybrid Meta Optimization for Finding better Parameters of an Evolution Strategy in Real World Optimization Problems

Thomas E. Koch, Volker Scheer, Jürgen Wakunda, Andreas Zell
Lehrstuhl Rechnerarchitektur, Wilhelm-Schickard-Institut für Informatik
Eberhard-Karls-Universität Tübingen, Köstlinstr. 6, D-72074 Tübingen
{`koch,wakunda,zell`}`@informatik.uni-tuebingen.de`

1 MOTIVATION

Evolution Strategies (ES) and Genetic Algorithms (GA) are two main members of Evolutionary Algorithms (EA), which have become widely used to solve optimization problems of real world applications. We used especially ES of our **Ev**olutionary **A**lgorithms Toolkit (EvA) [WZ97] for the following real world problems: (a) to forecast the minimum energy 3D structure of proteins, (b) motor optimization, (c) to position clamps for workpiece adjustment in wood-working industries [KMSZ99]. In this paper we deal mainly with the latter. This application is multi-objective, n-dimensional and must satisfy given constraints.

The problem is to find the best parameters for the ES and its optimizing application (here: optimal clamp positioning). It is very difficult to determine an exact mathematical function for the problem, because the evaluation of a found clamp configuration is only possible with the knowledge of experts. Therefore we want to know if our constructed fitness function models the real world good enough.

For this task we used a two-tier Meta-ES in a master/slave relationship: the secondary ES as master, which runs primary ESs as slaves. Running sequentially, the performance is unacceptable (> 10 hours). So we used parallel processing.

2 APPLICATION: OPTIMAL CLAMP POSITIONING

Figure 1 shows a result of this geometric optimization problem known in wood-working industries. The special task is to find an optimal positioning of vacuum clamps to fix a flat wood piece. These wood boards are fixed on a working table by vacuum clamps in order to treat them with a robot (shaping, sawing, snorring, boring, glueing ...). The goal is the optimal positioning of these clamps to avoid interference with robots or other clamps and to avoid fluttering in the work process. Objectives like *clamps should be near working traces* or *clamps should be equally distributed* are rating solutions found by a fitness function, which has to be optimized. This task requires processing in limited response time because it is embedded in a manufacturing workflow.

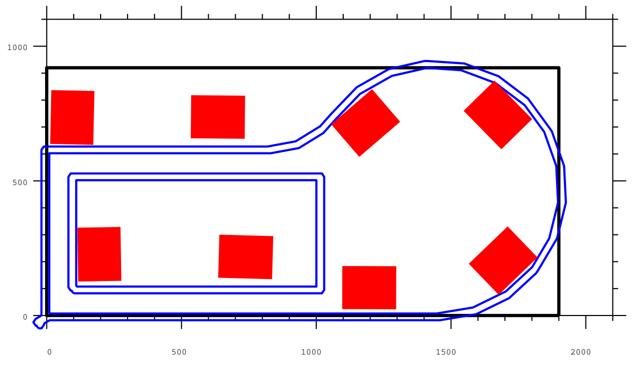

Figure 1: Result of an Evolution Strategy: the figure shows the work table with a raw woodpiece (solid rectangle), robot operation traces (thin polygons) and eight clamps (filled rectangles), which have to fix the wooden plate.

It is difficult to find a mathematical fitness function which meets all these objectives and constraints. In order to be more flexible, we parameterize this fitness function by weighting factors for the clamp distribution, closeness to the trace, different kinds of clamps, number of clamps etc. In addition there are the parameters of the ES like population sizes.

A further task is to find optimal parameters for special input classes. Most customers of our industrial partner produce special kinds of wooden pieces. The question here is, which are the best parameters for a customer, who produces only special, but similar pieces (e.g. kitchens).

3 META ALGORITHM

Figure 2 shows the two-tier, master/slave Meta-EA structure. The primary ES deals with basic optimization problems. The object variables (individuals) of the secondary ES are the parameters of the primary ES and our application parameters.

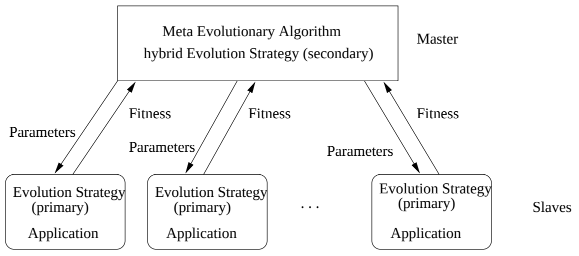
Meta Evolutionary Algorithm
hybrid Evolution Strategy (secondary)
Master
Parameters
Fitness
Parameters
Fitness
Parameters
Fitness
Evolution Strategy (primary)
Application
Evolution Strategy (primary)
Application
...
Evolution Strategy (primary)
Application
Slaves

Figure 2: Workflow of the Meta-EA: the Master sends object variables of its individuals of a population as parameters for an ES to the slaves, which each run one ES for the given application and return their results as fitness values of the best individual back to the master.

Because our primary ES has variables over alphabets of high cardinality (ES are better than GA here) and on the other over alphabets of low cardinality (GA are better than ES here) we use a hybrid of mainly an ES and some parts of a GA for the secondary EA. An example is the recombination operator for the object variables; the set of potential values is $R = \{\texttt{none}, \texttt{intermediate}, \texttt{dominant}\}$. In order to adapt this variable, one value x is taken randomly from the individuals of the recombination population. The mutation operator now determines an $[0, 1]$ equal distributed random number χ, tests it against the given mutation probabilty $p_m \in [0, 1]$: if $\chi > p_m$ then $x_{neu} = x$ else $x_{neu} \in R$ randomly determined.

The parameters of the primary ES to be optimized by the secondary ES are: population sizes μ, λ, ρ, initial self-adaption size δ, modification factor of self-adaption α, self-adaption on/off, type of self-adaption, recombination operator, selection operator; and for the clamp application the weighting factors of the fitness function, number and type of clamps (different polygon shapes).

Meta-EAs are very interesting because they give another kind of adaption for the searched parameters. In this case we have self adaption of the primary ES and the adaption of the parameters by the Meta-EA.

4 IMPLEMENTATION

The toolkit EvA which contains the ES, Meta-EA and applications is written in C/C++. The parallel part is implemented with MPI [For95]. Master and slaves are separate processes on different processors and communicate via MPI.

Meta-EA runs and has been used parallel on a cluster of linux workstations, two Hewlett-Packard V2 200 (4 and 16 nodes) machines and also on a two processor Windows NT 4.0 PC.

5 RESULTS

All parameters have been evaluated all alone, together with others and all together. In this manner we found the optimum for each parameter, their dependencies among each other and also obsolete ones (no influence on the fitness at all). In comparison to the procedure of trial and error in order to get better parameters we found for the most parameter better ones and got an better insight into dependencies of parameters.

For building classes of parameter sets for given similar input polygons (e.g. different table boards) we always obtained the same optimal parameters. This implies we have a rather robust fitness function, because all of the results have been improved. In addtition there is no need to distinguish between different input polygons.

6 RELATED WORK

The self adaption of parameters is the speciality of Evolution Stategies. But there is no research being done about finding optimal initial strategy parameters of an ES.

Genetic Algorithms do not have such a feature and have problems to adapt to the solution space. There are a few approaches in this field: Bäck and Schütz [BS96] adapt the mutation rate of the canonical GA, but not the main operator. Jeong and Lee [LL98] change probabilities for crossover and mutation operators while running the GA; the higher the fitness of an individuum the lower the probailities. Smith and Fogarty [SF97] and Tuson and Ross [TR98] look at the foundations of self adaption in general. Smith and Fogarty give an overview of self adaption for GAs while Tuson and Ross scrutinize the sense of self adaption.

On the field of Meta-EAs there is less research.

Bäck [Bäc96], Freisleben and Härtfelder [FH93] used GAs for primary (Meta-EA) and secondary EAs; they used in addition indicator functions to get the parameters with highest impact.

7 CONCLUSION

Our Meta-EA has been proven to find optimal parameters for an Evolution Strategy for real world optimization problems. We adapted these parameters with a parallel hybrid Meta Evolution Strategy.

This Meta algorithm may be used in order to find better parameters of complex, multi-objective problems in an acceptable time period. Additionally the constructed fitness function may be evaluated automatically.

Acknowledgments

Part of this work has been supported by Homag Maschinenbau AG, Schopfloch, Germany.

References

[Bäc96] Thomas Bäck. *Evolutionary Algorithms in Theory and Practice.* Oxford University Press, 1996.

[BS96] Thomas Bäck and Martin Schütz. Intelligent mutation rate control in canonical genetic algoithms. In Zbiginiew W. Ras and Maciek Michalewic, editors, *proceedings of the ninth international symposium on foundations of intelligent systems*, pages 158–167, Berlin, 1996. Springer.

[FH93] Bernd Freisleben and Michael Härtfelde. Optimization of genetic algorithms by genetic algorithm. In R.F. Albrecht, C.R. Reeves, and N.C.Steele, editors, *artificial neural nets and genetic algorithms*, pages 392–399, Wien, 1993. Springer.

[For95] Message Passing Interface Forum. *MPI: A Message Pasing Interface Standard*, 1995.

[KMSZ99] Thomas E. Koch, Ralph Müller, Franz Schnei-
der, and Andreas Zell. Positionierung von Spannmitteln zur Werkstückfixierung mittels Evolutionärer Agorithmen. *Automatisierungstechnische Praxis atp*, pages 32–39, 9 1999.

[LL98] Il-Kwon Leong and Ju-Jang Lee. a self-organizing genetic algorithm for multimodal function optimization. *artificial life and robotics*, 2:48–52, 1998.

[SF97] Jim E. Smith and Terence C. Fogarty. Operator and parameter adaption in genetic algorithms. *soft computing*, 2(1):81–87, 1997.

[TR98] Andrew Tuson and Peter Ross. adapting operator settings in genetic algorithms. *evolutionary Computation*, 2(6):161–184, 1998.

[WZ97] Jürgen Wakunda and Andreas Zell. EvA - a tool for optimization with evolutionary algorithms. In *Proceedings of the 23rd EUROMICRO Conference*, Budapest, Hungary, September 1-4 1997.