Heuristic Optimization Methods for the Tuning of Input Parameters of Simulation Models

Michael Affenzeller[1], Gabriel Kronberger[1], Stephan Winkler[1], Mihaela Ionescu[2] and Stefan Wagner[1]

[1]Upper Austrian University of Applied Sciences, Campus Hagenberg
Heuristic and Evolutionary Algorithms Laboratory
Softwarepark 11, A-4232 Hagenberg, Austria

[2]Johannes Kepler University Linz
Institute of Bioinformatics
Altenbergerstr. 69, A-4040 Linz, Austria

E-Mail: {michael,gabriel,stephan,mihaela,stefan}@heuristiclab.com

ABSTRACT

This paper proposes an approach combining Evolutionary Computation and Simulation: An extended problem representation as well as solution manipulation concepts for Evolution Strategies are proposed with the aim to fulfill the needs of input parameter optimization of simulation models; moreover, this approach is not restricted to real-valued parameters. The usage of a parallel infrastructure is also proposed and planned to be realized based on the HeuristicLab framework for heuristic optimization.

1. INTRODUCTION

Due to the continuously increasing availability of computing power, combinations of simulation and heuristic optimization become more and more attractive and also more and more practicable in various areas of application. An approach that is frequently used in this area is to use simulation models for heuristic optimization in order to obtain a quality measure for potential solution candidates in cases where the quality of a solution cannot be obtained analytically. In other words, the fitness evaluation is done by simulation instead of computation which is usually much more time consuming but indispensable in some cases.

In that sense (Brady and McGarvey 1998) for example use a computer simulation model to generate output responses for Simulated Annealing, Tabu Search, Genetic Programming, and a novel frequency based heuristic approach. (Stoecher et al. 2007) utilize a simulation model for estimating the quality of complex priority rules for production planning optimization whereas the optimization of the priority rules is done by means of Genetic Programming.

The field of Simulation Optimization on the other hand deals with the optimization of simulation models, i.e. to find out which of possibly many sets of model specifications given in terms of input parameters and/or structural assumptions lead to optimal performance. A comprehensive overview of Simulation Optimization including several typical applications is given in (April et al 2004), (Azadivar 1999), (Brady and McGarvey 1998), (Fu and Glover 2005) or (Bowden and Hall 1998).

A very essential aspect in all combinations of simulation and heuristic optimization is efficiency. On the one hand, modern and capable meta-heuristics rely on the evaluation of a huge amount (usually millions) of potential solution candidates; the evaluation of a simulation model on the other hand requires much more computational effort than the evaluation of a mathematical function. Therefore, the efficiency of the simulation model is a bottleneck in all of these approaches. The only solution to this problem is often the application of parallel and grid computing (e.g. Stoecher et al 2007). Parallelization is especially efficient and easy to implement in combination with population-based meta-heuristics; a Java framework for distributed simulation optimization is given in (Gehlsen 2001), e.g.

One of the central aims of the approach proposed in this paper is to model, adapt and combine various solution manipulation concepts from the theory of heuristic optimization so that these can be utilized in the optimization of input parameters of simulation models. This is to be done in such a generic way that more or less arbitrary simulation models can define their respective interfaces for the input parameters which are to be optimized. Furthermore, we are planning to use HeuristicLab (Wagner and Affenzeller 2005), a paradigm-independent and extensible environment for heuristic optimization, as optimization framework. The input parameters of the simulation model may be binary, integer, real-valued, or even a decision situation in that sense that from i to j elements are to be chosen out of a set of n possible values where $0 \leq i \leq j \leq n$.

As concrete heuristic optimization algorithm we aim to use Evolution Strategies (ES) with self adaptive mutation step-width as ES are considered one of the most capable heuristic optimization techniques for parameter optimization purposes (as is described in Section 3). The adaptation of

mutation step-width is additionally carried out independently for each dimension of the parameter vector which is another strong argument for our purposes.

The rest of the paper is organized as follows: Section 2 gives a brief overview about Evolutionary Computation in general whereas Section 3 addresses Evolution Strategies (ES) which is the heuristic optimization method on which the simulation parameter optimization discussed in the present paper is based upon. Sections 4 and 5 focus on the limitations of conventional ES-problem representations and the according solution manipulation operators when it comes to practical parameter optimization problems including mixed-discrete constrained parameter optimization as it is the case when tuning the input parameters of simulation models. Finally, Section 6 and Section 7 describe the algorithmic framework intended to be used for the described approach and end up with some concluding remarks, respectively.

2. EVOLUTIONARY COMPUTATION

Evolutionary Algorithms use the principles of natural evolution to solve complex optimization problems. In successive iterations of the simulated evolution, good individuals of a population of solution candidates are selected for interbreeding and the generation of new solution candidates with the goal to generate better solutions candidates. The first Evolutionary Algorithms described were Genetic Algorithms (Holland 1975) and Evolution Strategies (Rechenberg 1971). Traditionally, Genetic Algorithms used bitstrings to encode solution candidates and concentrated on recombination as a means to create better solution candidates while Evolution Strategies used a vector of real-valued parameters to encode a solution candidate and concentrated on mutation to tune the parameter vector. However nowadays Genetic Algorithms also use more sophisticated solution representations and Evolution Strategies also include recombination operators.

Genetic Programming (Koza 1992) is mainly based on Genetic Algorithms; the main difference is that complete computer programs or mathematical functions are evolved instead of parameter vectors. Typically, the individuals are represented by a tree structure known from functional programming languages like LISP to allow simple definitions of recombination operators (e.g. the exchange of subtrees).

Since ES was originally developed for continuous parameter optimization, we are convinced that is also well suited for the optimization of simulation models. Various extensions for ES have also been developed that are especially tuned for parameter optimization; relevant extensions are described in more detail in the next section.

3. EVOLUTION STRATEGIES (ES)

The first Evolution Strategy described used a population of one parent and one offspring typically denoted as (1+1)-ES. Starting from a randomly initialized parent, mutation adds Gaussian distributed noise to each value of the parameter vector. The mutated parameter-vector is evaluated and replaces the old parent if it is better than the parent; if not, the new solution is discarded.

This algorithm was later extended to the $(\mu + 1)$-ES that selects uniformly from μ parents to generate and mutate a new individual and then keeps the μ best individuals from the parents and the new child.

Schwefel (Schwefel 1987) soon came up with the general $(\mu + \lambda)$-ES where λ new individuals are generated for the next replacement phase. This $(\mu + \lambda)$-ES is effectively an elitist algorithm because it selects the parents for the next mutation from the old parents and its offspring. This means that a disproportionately good individual will stay in the population for a long time which could lead to the negative effect of premature convergence. An alternative scheme called 'comma-strategy' tries to lessen this effect by using the μ best individuals only from the offspring not considering the old parents. This is denoted by (μ, λ)-ES.

The random noise added to each element of the parameter vector is normally distributed with parameters $(0, \sigma)$. Rechenberg observed for the (1+1)-ES that adapting σ in the course of the evolution leads to better results; based on this observation he formulated the "1/5 success rule" that adapts σ based on the average success of the last n mutations with the goal of generating on average 1/5 successful mutations. When the ratio of successful mutations is greater than 1/5, the standard deviation should be increased while it should be decreased when the ratio of successful mutations is less than 1/5.

Schwefel and Bäck further improved ES by the introduction of self adaptive mutation (Back and Schwefel 1993). For self adaptive mutation the length of the real-valued vector that represents an solution candidate is doubled where the second part holds the variance σ_i for each individual element x_i of the original parameter vector. The self adaptive mutation uses the corresponding variance σ_i to mutate each parameter x_i individually while the variance values at the end of the vector are subject to traditional mutation with constant or adaptive variance. The effect of this scheme is that the ES optimizes the step size for each di-

mension individually leading to a well balanced search in each dimension.

The idea of applying Evolution Strategies for the search for optimal parameters is not new and dates back to the late seventies (Schwefel 1979). However, in this contribution Schwefel considered just real-valued input parameters and also adaptive steering of mutation step width was not available at those days. A more recent article dealing with use of ES integrated with a simulation model is given by (Hall and Bowden 1996) which describes the use of ES to solve the kanban sizing problem.

Conventional ES are indeed restricted to real-valued vectors. However, in some cases solution candidates for an optimization problem cannot be represented in this way. Typical examples for such values are boolean switches or discrete values which are also common parameters in simulation models. The next section describes extensions to the representation of solution candidates that make it possible to optimize such parameters with ES as well.

4. EXTENDED ES-PROBLEM-REPRESENTATION FOR MIXED-DISCRETE PARAMETER OPTIMIZATION

In the previous section we have stated that the representation of solution candidates in standard ES is not suited for the optimization of simulation models because parameters of simulation models are typically not limited to real values.

ES with self adaptive mutation length uses real-valued vectors $\vec{a} = (\vec{x}, \vec{d}, \vec{\sigma}, \vec{p})$ to represent solution candidates where $\vec{x} \in R^n$ is the object variable for optimization and $\vec{\sigma}$ holds the self adaptive standard deviations. The fitness of an individual depends only on the object variable $\vec{x}$. The mutation operator first mutates each element of $\vec{\sigma}$ independently and then uses the new standard deviations to mutate each element of $\vec{x}$:

$$\sigma_i'' = \sigma_i' \cdot exp(\tau' \cdot N(0,1) + \tau \cdot N_i(0,1)),$$
$$\vec{x}'' = \vec{x}' + \vec{N}(0, \vec{\sigma}'')$$

where τ is an exogenous parameter that controls the step size for mutating the standard deviations.

For the optimization of simulation models we have to extend the standard representation to following parameter types:

Integer parameters and continuous parameters with restricted search spaces are realized by means of constraint functions of the form $c_i : R^n \to R$. We distinguish between three kinds of constraints: Equality constraints, inequality constraints and integer constraints with respective index sets I_1, I_2 and I_3:

$$R = \{ x \in R^n \mid c_i(x) = 0, i \in I_1 \} \cap$$
$$\{ x \in R^n \mid c_i(x) \le 0, i \in I_2 \} \cap$$
$$\{ x \in R^n \mid x_i \in Z, i \in I_3 \}$$

Discrete parameters for which no distinct order relation can be formulated are also to be considered. An example in the context of manufacturing simulation models is the choice of a dispatching strategy for orders or the choice of lot sizing policies. Since such parameters cannot be described properly by means of constraint functions we extend the definition of the search space to $\Omega = R \times M$ where $M = \otimes_i^m M_i$ and M_i denotes an arbitrary disordered set with a finite number of elements that represent the possible choices for this parameter.

The representation is now extended to $\vec{a} = (\vec{x}, \vec{d}, \vec{\sigma}, \vec{p})$ where $\vec{x} \in R^n$ denotes the vector of continuous objective values, $\vec{d} \in M$ the discrete objective values, $\vec{\sigma} \in R^n$ the standard deviations for mutation and $\vec{p} \in [0,1]^{nd}$ the mutation probabilities for the discrete values. The mutation for the real-valued part of the objective values remains unchanged. The discrete part of the objective values and the mutation probabilities for the discrete variables are mutated as proposed in (Back and Schutz 1995):

$$p_j'' = \left(1 + \frac{1 - p_j'}{p_j'} \cdot exp(-\gamma \cdot N_j(0,1)) \right)^{-1},$$
$$d_j'' = \begin{cases} d_j' & if\ u > p_j'' \\ D & if\ u \le p_j'' \end{cases}$$

where the uniformly distributed variables $u \subset U(0,1)$ and $D \subset U(M)$ are sampled for each dimension independently and γ is an exogenous parameter which is optionally automatically adapted by the ES. In other words, the mutation operator changes each discrete parameter with a given probability where a change means that it selects one element of the set of possible values for this parameter based on a uniform distribution.

5. USING EXTENDED ES-CONCEPTS FOR THE OPTIMIZATION OF INPUT PARAMETERS OF SIMULATION MODELS

Simulation can be used to analyze the behavior of complex systems by means of a computer model. Simulation models can be formulated in various ways of which the most common are system dynamics, discrete event driven or agent based systems. However, for the approach discussed in this paper the language used to define the model is irrelevant; all simulation models simply have to have in common that there are a number of input parameters which influence the behavior of the system and a number of variables that describe the effects or outputs of the system. Typically, the values of the variables are non-deterministic and it is impossible to predict the values of the output variables based on the parameter settings.

One main intention of simulation is to find good or even optimal parameter settings for the system that lead to the best output. When the number of parameters and the search space is relatively small one can try to find the best settings manually by observing the effects of successive parameter changes on the output variables. However the limit of this approach is reached very easily and it is hopeless to find optimal settings for models that are reasonably interesting. Therefore heuristic optimization methods are necessary for the optimization of simulation models. We suggest that ES could be easily and effectively used for this kind of optimization problem.

Simulation models can have different kinds of parameters:

- Real-valued parameters,
- discrete parameters,
- boolean parameters: single choice, and
- set parameters: multiple choice

In the extended ES model described in the previous section each of these parameter types can be represented in solution candidates.

A further aspect that may come along with the tuning of input parameters of simulation models is the fact that there is often more than one objective that should be optimized by the simulation model which leads to the theory of multi-objective optimization. For this purpose we plan to optimize the weighted sum of functions of the multi objective problem as

$$\sum_{i=1}^{n} \alpha_i f_i(x)$$

where it is up to the user to choose appropriate weights. Also it is quite easy to show that the minimizer of this combined function is Pareto optimal.

One further problem of simulation optimization is that a single simulation might take several seconds or even minutes for very complex problems. To be able to optimize the model it is necessary to run multiple simulations this leads to very long runtimes. However since ES are inherently parallel it is possible to run many simulations concurrently on a relatively cheap and simple cluster. This makes it possible to find good solutions to the problem in reasonable runtime.

6. THE BASIC ARCHITECTURE OF A GENERIC FRAMEWORK

Careful consideration is necessary in the design phase of a generic framework that implements the optimization method described in this paper. In order to achieve maximal effectiveness the implementation has to be compatible to various simulation environments as many commercial simulation environments provide a convenient interface to database management systems. Therefore the proposed implementation will use a common database (DB) to exchange data between the simulation model and the ES. This design makes it possible to combine optimization and simulation with only minimal changes in the simulation models which can be implemented in any possible programming language or simulation environment that supports DB connections.

The proposed architecture has three distinct components: the optimization component that executes the ES, a central DB that stores solution candidates and simulation results, and a simulation client that polls the DB for new solution candidates, simulates the model with the supplied parameters and writes the results of the simulation back into the DB.

Since each simulation model has different parameters and results it is necessary to define a custom ES representation for each simulation model that holds the names and possible values for each possible parameter and that can evaluate the quality of a solution based on the result values from the simulation. The fields of this representation can be directly mapped to database tables for the parameters and results of the simulation.

Another benefit of this architecture is that it is possible to start multiple simulation clients in a global parallelization scheme for the evaluation of solution candidates. This is especially profitable when the simulation time is longer than just a few seconds.

Of course a large overhead is generated by the communication with the DB; however, we are confident that the flexibility of this approach outweighs the costs by far.

7.CONCLUSION AND FUTURE PERSPECTIVES

In this article we have described a conceptual framework for the optimization of input parameters for simulation models. The next goal is the implementation of these concepts in a generic environment in order to allow further experiments and analyses of the optimization methods. The requirements for this implementation include the support of various kinds of simulation models from diverse problem domains as well as a mechanism for a comfortable exchange of these models. Therefore, a generic interface for control and communication with simulation models is necessary that allows to set parameters of the model, execute simulation runs and retrieve values of the target variables for the optimization process. A pragmatic way to implement this interface is to use a database management system to store the model parameters and simulation results. The use of a DBMS brings along the advantageous consequences that simulation and optimization are completely independent from each other and that it is easy to run multiple simulations with different parameter settings concurrently.

We plan to integrate the implementation of this generic optimization environment for simulation models into HeuristicLab (http://www.heuristiclab.com), a paradigm-independent environment for heuristic optimization which already provides implementations of various heuristic optimization methods.

REFERENCES

J. April et al., 2003. Practical Introduction to Simulation Optimization, Proceedings of the 2003 Winter Simulation Conference, pp 71--78.

F. Azadivar, 1999. Simulation Optimization Methodologies. Proceedings of the 1999 Winter Simulation Conference, pp 93--100.

T. Back, H.-P. Schwefel, 1993. An overview of evolutionary algorithms and parameter optimization. Evolutionary Computation 1: 1--23.

R. Bowden, J. Hall, 1998. Simulation optimization research and development, Proceedings of the 1998 Winter Simulation Conference, pp. 1693 -- 1698.

T Brady, B. McGarvey. 1998. Heuristic Optimization Using Computer Simulation: A Study of Staffing Levels in a Pharmaceutical Manufacturing Laboratory, Proceedings of the 1998 Winter Simulation Conference, pp 1423--1428.

M.C. Fu, F.W. Glover, 2005. Simulation Optimization: A Review, new Developments, and Applications, Proceedings of the 2005 Winter Simulation Conference, pp 83--95.

B. Gehlsen., 2001. A Framework for Distributed Simulation Optimization, Proceedings of the 2001 Winter Simulation Conference, pp. 508--514.

J. Hall, J.R. Bowden and J. Usher, 1996. Using evolution strategies and simulation to optimize a pull production system. Journal of Materials Processing Technology 61, pp. 47—52.

J. Holland, 1975. Adaption in Natural and Artificial Systems, The University of Michigan Press, Ann Arbor.

J. Koza, 1992. Genetic Programming: On the Programming of Computers by Means of Natural Selection, MIT Press.

I. Rechenberg, 1971. Evolutionsstrategie – Optimierung technischer Systeme nach Prinzipien der biologischen Evolution, PhD thesis, Reprinted by Fromman-Holzboog, Stuttgart, 1973.

H.-P. Schwefel 1979. Direct search for optimal parameters within simulation models, Proceedings of the 12th symposium on Simulation, Tampa, Florida, USA, pp. 91 -- 102.

H.-P. Schwefel 1987. Collective Phaenomena in Evolutionary Systems. 31st Annual Meeting International Society for General System Research, Budapest, 1025-1033.

W. Stoecher, B. Kabelka and R. Preissl, 2007. Automatically Generating Priority Rules For The Flexible Job Shop Problem with Genetic Programming, Accepted to be published in Proceedings of the EuroCAST 2007, LNCS.

S. Wagner and M. Affenzeller, 2005. HeuristicLab: A Generic and Extensible Optimization Environment. Adaptive and Natural Computing Algorithms, Springer Computer Science, pp. 538--541.

BIOGRAPHIES

MICHAEL AFFENZELLER has published several papers and journal articles dealing with theoretical aspects of Genetic Algorithms and Evolutionary Computation in general. In 1997 he received his MSc in Industrial Mathematics and in 2001 his PhD in Computer Science, both from the Johannes Kepler University Linz, Austria. He is professor at the Upper Austrian University of Applied Sciences Hagenberg, Austria and associate professor at the Institute of Formal Models and Verification at Johannes Kepler University Linz, Austria since his habilitation in 2004.

GABRIEL KRONBERGER is a research associate at the Research Center of the Upper Austria University of Applied Sciences in Hagenberg. He received his MSc degree in computer science from Johannes Kepler University Linz in 2005. His research interests include manufacturing simulation, production planning optimization, evolutionary algorithms especially genetic algorithms and machine learning.

STEPHAN M. WINKLER received his MSc in Computer Science from Johannes Kepler University Linz, Austria in 2004. His research interests include Genetic Programming, Nonlinear Model Identification and Machine Learning. Currently he is a research associate within the Translational Research Program L284 "GP-Based Techniques for the Design of Virtual Sensors", a research project funded by the Austrian Science Fund (FWF).

MIHAELA IONESCU is an assistant at the Institute of Bioinformatics at the Johannes Kepler University Linz. She received her MSc degree in computer science from Johannes Kepler University Linz in 2005. Her research interests include alternative splicing, machine learning and evolutionary algorithms especially evolution strategies

STEFAN WAGNER also received his MSc in Computer Science from Johannes Kepler University Linz, Austria in 2004. He now holds the position of an associate professor at the Upper Austrian University of Applied Sciences Hagenberg, Austria. His research interests include Evolutionary Computation and Heuristic Optimization, Theory and Application of Genetic Algorithms, Machine Learning and Software Development.

The Web-pages of the authors as well as further information about HeuristicLab and related scientific work can be found at `http://www.heuristiclab.com`.