Meta-Model - based Estimation of Optimal System Variants with MATLAB/Simulink

Olaf Hagendorf[1] *, Thorsten Pawletta[1], Roland Larek[2]

[1]Wismar University of Applied Sciences, Res. Group Computational Engineering and Automation (CEA), PB 1210, 23952 Wismar, Germany, **olaf.hagendorf@hs-wismar.de*

[2]Bremen University of Applied Sciences, Foundation Institute of Materials Science, Bremen, Germany

Abstract. The idea that the efficient usage of all necessary production resources has not only an ecological importance but is also an overwhelming economical competitive factor becomes accepted since the 1980s. Modeling and simulation with integrated parameter optimization is used routinely to improve process performance. In engineering a well known environment for this task is the MATLAB/Simulink programming system. Using this or similar, established techniques only model parameter of a single model structure is optimized. Model structure is considered to be fixed as the relationships between model elements are defined during model development. Until now no tools and methods are known which can optimize product design and production processes utilizing all existing degrees of freedom. As process performance is optimized it may be necessary to redesign the model structure. The redesign is normally carried out manually by an analyst but not automatically by the optimization method. This suboptimal combination of automatic parameter optimization and manual structure changes leads to a time consuming and error-prone optimization task.

The system theoretical approach of the System Entity Structure/Model Base framework (SES/MB) is able to define alternative model structures and parameter sets in a single meta-model, called System Entity Structure (SES). Moreover, atomic models are stored in a model base (MB). Using both, SES and MB, it is possible to generate modular, hierarchical models with different structures and parameters.

Evolutionary Algorithms are a subtopic of Artificial Intelligence that are involved in combinatorial optimization problems. These algorithms are based on ideas inspired by biological evolution. They often perform well for many problem types because they do not make assumptions about the problem specific search space.

The research reported in this paper details an approach providing optimization through automatic reconfiguration of both: model structure and model parameters. An evolutionary algorithm based optimization method is assisted by an SES/MB based model management. It searches for an optimal solution with repeated, combined model parameter and model structure changes resulting in a combined parameter and structure optimized model

Introduction

Modeling and simulation with integrated optimization is a well-established technique in engineering applications. Such techniques are used for system design, real time planning and to control production systems. The MATLAB / Simulink environment is one of the well known and widely used modeling and simulation tools in this application field [3 7 8].

With increasing complexity and flexibility of systems the requirements for modeling and simulation tools are growing. Existing applications using simulation based optimization are restricted to parameter optimizations. The user has to change model structure manually and repeat optimizations until a suitable solution is found. With increasing production system flexibility the number of possible structure variants increases and the potential benefit of automatic model structure optimization becomes significant.

The focus of this paper is the description of a methodology for a combined parameter and structure optimization for modular, hierarchical systems, which is implemented and tested in the MATLAB/Simulink environment but not restricted to it. In contrast to the common usage the model structure and model parameter values are controlled together by a super ordinate optimization module. To support the optimization method appropriate model management and generation methods are necessary.

SNE Simulation Notes Europe – Print ISSN 2305-9974 | Online ISSN 2306-0271
SNE 22(1), 2012, 7-14 | doi: 10.11128/sne.22.tn.10105

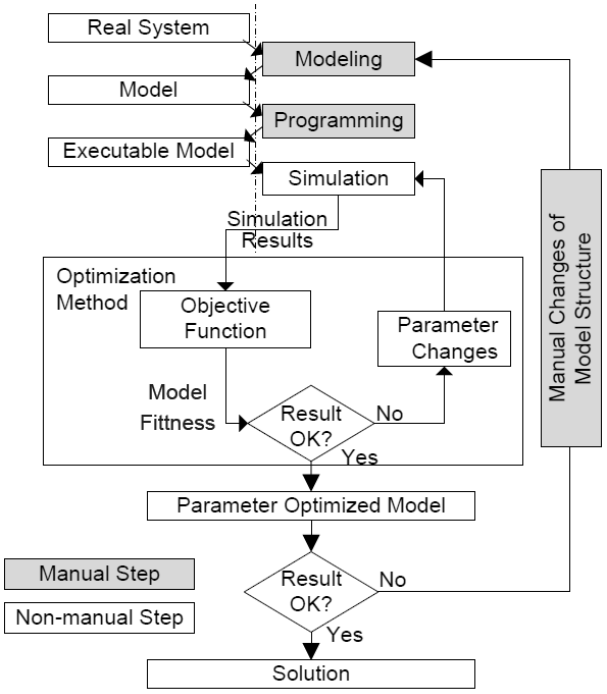

Figure 1. Simulation based parameter optimization.

As a basis for model management and generation the System Entity Structure/Model Base (SES/MB) formalism, introduced by Rozenblit, Zeigler et al [4 5 6], is employed. The SES/MB formalism is a generic, knowledge based framework consisting of a tree like entity structure and a model base. With its features the framework is able to define a set of modular, hierarchical models and to generate specific parameterized model structures.

Section 1 provides a short summary to the well-known simulation based parameter optimization approach. In the following the basic ideas of the advanced simulation based optimization approach are introduced. The synthesis of three fundamental elements: (i) optimization algorithm, (ii) model management and model generation and (iii) simulation execution to a framework for a combined structure and parameter optimization of complex modular, hierarchical systems is presented.

Finally, the new ideas are illustrated by a small logistical problem in Section 2. The implementation in MATLAB / Simulink is sketched. The conclusion refers to real world problems solved by the introduced approach.

1 A Framework for Parameter and Structure Optimization

1.1 Simulation Based Optimization

Simulation experiments can be of different complexity. The least complex ones are ordinary simulation runs. The modeling and simulation method is highly dependent on the type of the real system, the analysis objectives and the used software environment. In MATLAB / Simulink the programming step from the user's point of view consists of a graphical based model specification and often it is considered as part of the modeling. However, after examining simulation results the analyst manually changes model parameter values and/or model structure and starts the simulation again. These steps are repeated until a suitable solution is found. A more complex approach is simulation based parameter optimization, depicted in Figure 1.

The optimization method alters model parameter values to improve the result of an objective function until a stop criterion is fulfilled. The result is a parameter optimized model. Structure changes are carried out manually by a user followed by a repetition of the automated parameter optimization.

It is important to note that automatic structure changes induced by an optimization method are not possible with this approach. Instead, structure changes are carried out manually by a user. Each structure change requires a repetition of the parameter optimization. This principle is typical when using MATLAB / Simulink in a simulation based optimization. Therefore a Simulink model with a fixed structure is embedded into a super-ordinate optimization method which can alter specific Simulink block parameters to find the optimal values. Structure changes are done manually.

1.2 Principle of the Framework with combined Parameter and Structure Optimization

The idea behind this research is the extension of the simulation based parameter optimization method with the ability to additionally change the model structure thus improving the objective function result. The effect of this extension is a simulation based structure and parameter optimization.

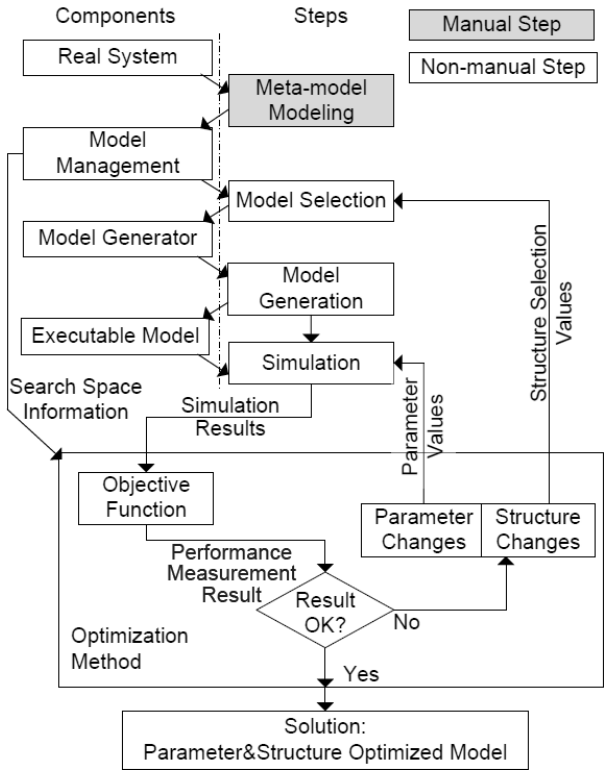

Figure 2. Principle of a simulation based parameter structure optimization.

Figure 2 presents the approach in principle with its combination of three methods: (i) a meta-model framework for model management, (ii) a modeling and simulation environment and (iii) an optimization method. In contrast to the established approaches:

1. The optimization method controls both: the model parameter values and the model structure, changing both until a stop criterion is fulfilled. The result of this process is a combined parameter and structure optimized model.
2. The user has to organize a set of models. One possibility is to define a model which describes a set of model variants instead of one single model of the real system. Such models that define the creation and interpretation of a set of models are named meta-models. Through this inclusion of the meta-model based automatic model generating element the optimizer can additionally control model structure changes to find an optimized solution.

Crucial parts and algorithms of this approach are described in the next sections.

1.3 Specification of Model Sets with SES/MB

As an appropriate meta-modeling framework the System Entity Structure/Model Base (SES/MB) formalism was determined. This formalism is a general, knowledge based framework. With its key feature to depict the three relationships (i) decomposition, (ii) taxonomy and (iii) coupling it is capable of defining a set of modular, hierarchical models [4 5 6].

Decomposition means that the formalism is able to decompose an object into sub-objects. Taxonomy means the ability to represent several, possible variants of an entity. Composition of an entity from sub-entities is done by coupling.

A SES/MB meta-model consists of two major parts: (i) a system entity structure (SES) and (ii) a model base (MB):

- The SES is a tree like structure which contains invariable and variable branches. The variable branches start at decision nodes. To create one structure variant the entity structure is pruned. At each decision node a specific branch is chosen. The result is a pruned entity structure (PES) which is the basis to create a hierarchical model that can be simulated.
- A SES can contain different node types and additionally, nodes can have optional and/or obligatory properties. One node type, the atomic entities have their counterparts in the MB.
- The MB contains a corresponding basic model for each atomic entity. The types of the basic models can vary. It depends only on the underlying simulation method e.g. in this research the model base matches the Simulink library.

A detailed description of all elements can be found in reference [1].

1.4 Optimization Process

Before an optimization can be carried out, information about the search space, in particular its dimensions and domains, is necessary. In this approach the search space is defined by the set of model structure variants established by analyzing the SES and the set of model parameters, defined by each model structure.

During the optimization process several points in the search space are examined. Each point defines one single model structure to be generated through the model generator with one set of parameter values.

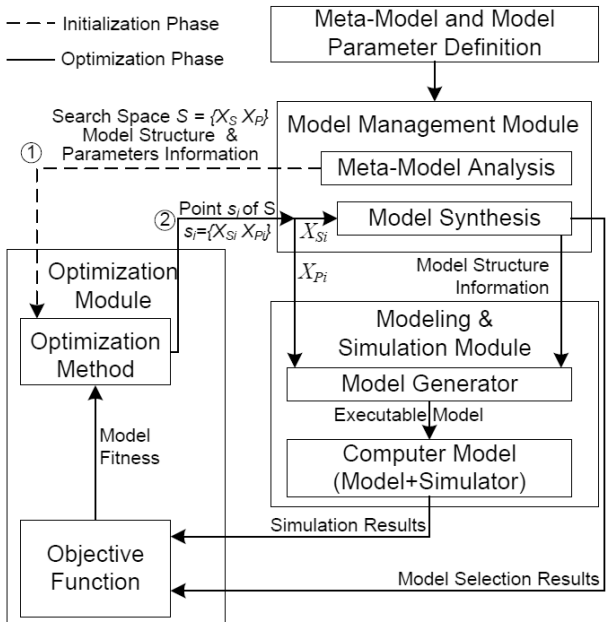

Figure 3. Structure and operation of the framework.

On closer examination of the framework, as depicted in figure 3, it is crucial to divide an optimization experiment into two phases:

1. *Initialization phase*: The model management reads and analyzes a meta-model. Results of the analysis are information about the multidimensional search space $S = (X_S, X_P)$. The optimization module is initialized with this information.
2. *Optimization phase*: During the optimization phase the search space is explored within a loop. Each examined search space point, i.e. an ordered set of values $(X_{Si}\ X_{Pi})$ is delivered to the model management module. This module starts up the processes: structure synthesis, model generation, simulation and perfor-mance estimation. The optimization loop ends when a predefined stop criterion is fulfilled.

1.5 Interfaces: Optimization Module – Model Management Module

Crucial parts of this framework are the interfaces (1) SES tree analysis during initialization phase and (2) transformation (search point + SES) → PES during optimization phase both depicted in Figure 3.

Within interface (1) the Model Management Module has to analyze the SES tree to transform formal meta-model structure information into numerical data useable by the Optimization Module.

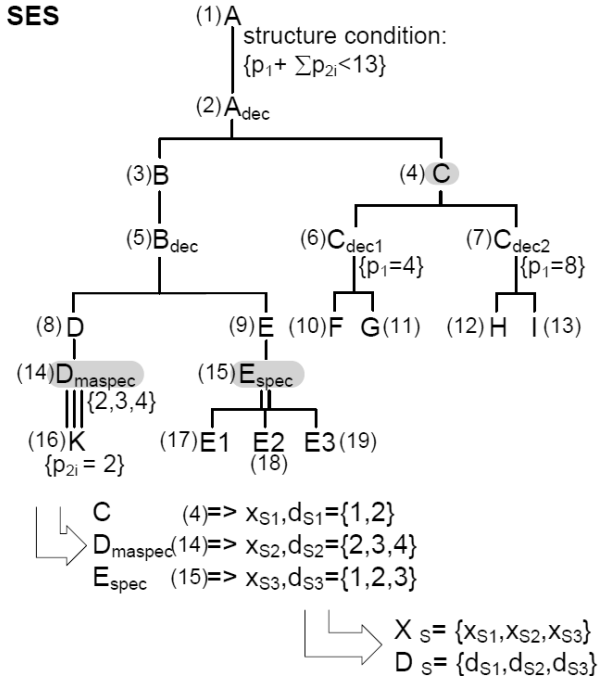

Figure 4. Transformation SES → sets X_S + D_S

This is done by a tree analysis starting at the root node, traversing the tree in a defined direction, namely depth-first and breadth-first analysis, and considering every node. If a node contains variable branches a structure parameter x_{Si} is added to the structure parameter set X_S with a corresponding domain d_{Si} added to the domain set D_S.

Figure 4 illustrates the algorithm for creating these sets using a breadth-first analysis. The steps of build-up order are marked with small sequence numbers. Starting at node *A*, the composite entity node *C* is the first decision node. A first parameter x_{S1} is added to the search space X_S with the domain $d_{S1} = \{1,2\}$ because it has two alternative successors.

The next decision node is D_{maspec} with a number range property $\{2,3,4\}$. A second parameter x_{S2} is added to X_S with the domain $d_{S2} = \{2,3,4\}$. The next node, the specialization node E_{spec}, is again a decision node. It has three alternative successors. A third parameter x_{S3} is added to X_S with domain $d_{S3} = \{1,2,3\}$.

The remaining nodes are non-decision nodes. Thus, the resulting structure parameter set is $X_S = \{x_{S1}, x_{S2}, x_{S3}\}$ with the corresponding domain set $D_S = \{d_{S1}, d_{S2}, d_{S3}\}$. These sets together with the model parameter and domain sets X_P, D_P define the search space S.

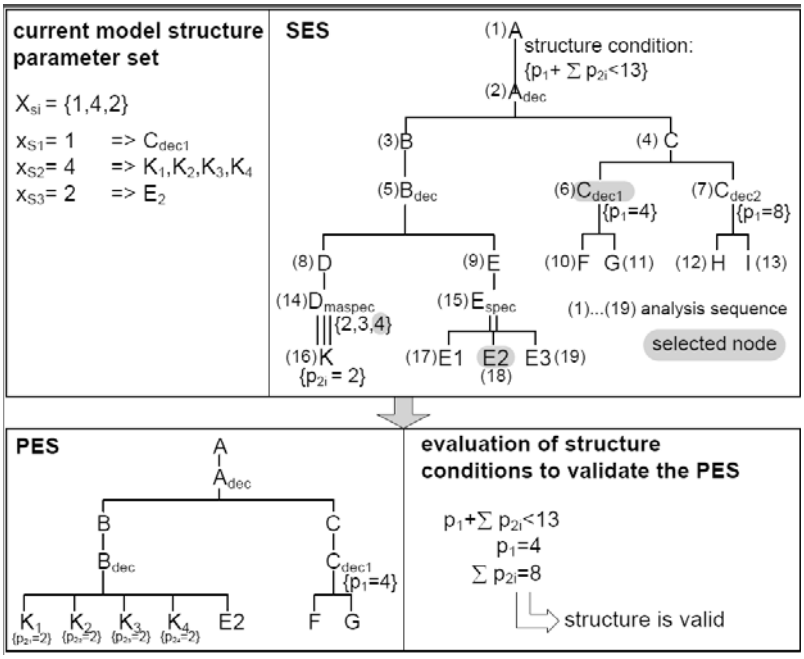

Figure 5. Transformation X_{Si} + SES → PES.

The second transformation, the line (2) in Figure 3, is the reverse of the first. The Model Management Module gets a point si in the search space from the Optimization Module i.e. the numerical data set $X_i^* = X_{Pi} \cup X_{Si}$, where set X_{Si} codes a specific model structure and set X_{Pi} codes its model parameters. It has to synthesize the corresponding model structure and to infer the model parameters. The transformation traverses the tree using the same direction as during the first transformation.

Figure 5 illustrates the principle of the second transformation. The analysis and pruning order is marked with sequence numbers. The breadth-first analysis starts at the root node *A*. The first decision node is node *C*. The first element of X_{Si} is $x_{S1} = 1$, i.e. the first aspect node C_{dec1} is chosen for the PES. The next decision node is node D_{maspec} and the corresponding set element is $x_{S2} = 4$, i.e. the PES contains four nodes *K*.

The last decision node is node E_{spec} and the corresponding set element is $x_{S3} = 2$, i.e. the PES contains the second specialization *E2*. After pruning, the PES is verified by evaluating structure conditions, indicated at node *A* in Figure 4 and Figure 5. Afterwards the model generator uses the PES to generate an executable simulation model utilizing basic models stored in the MB.

As mentioned, the result of the model synthesis and model generation is an executable simulation model with a specific model structure and specific model parameter values. Simulating this model delivers the simulation results of a single simulation run. In case of stochastic parameters simulation runs have to be repeated. The simulation results or the mean values are the input for the objective function.

2 Application: Process Planning

The chosen example is a typical optimization problem of a job scheduling system with two variable properties: the job arrival time and the number of facilities. The complexity of the system is defined consciously simple to demonstrate the introduced approach.

The introduced concept of a simulation based parameter and structure optimization was implemented in the MATLAB/Simulink environment. The necessary modeling and simulation method is provided by the MATLAB/Simulink system and the additional MATLAB toolbox SimEvents itself.

In [1] a MatlabSES toolbox was described. This toolbox handles SES meta-models described by XML files and implements the algorithms explained above. During this research a new model generator has been implemented. It uses Simulink blocks i.e. original blocks from the Simulink libraries or self-implemented Simulink blocks as basic models which are referenced by the SES meta-model.

MATLAB incorporates an Optimization toolbox [9] containing an EA implementation. The MatlabSES model synthesis and the Simulink model simulation are called through function pointer by the EA toolbox.

Figure 6 depicts one variant of the process implemented as a Simulink model. The variable model parts are marked with boxes.

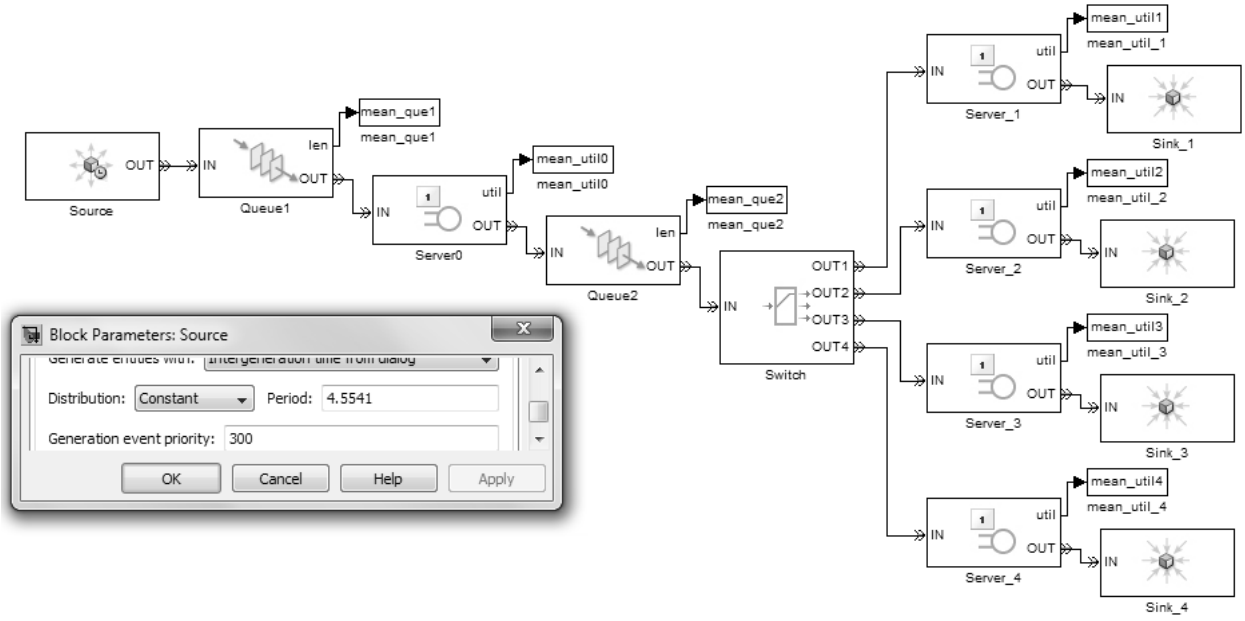

Figure 6. One variant of the example application.

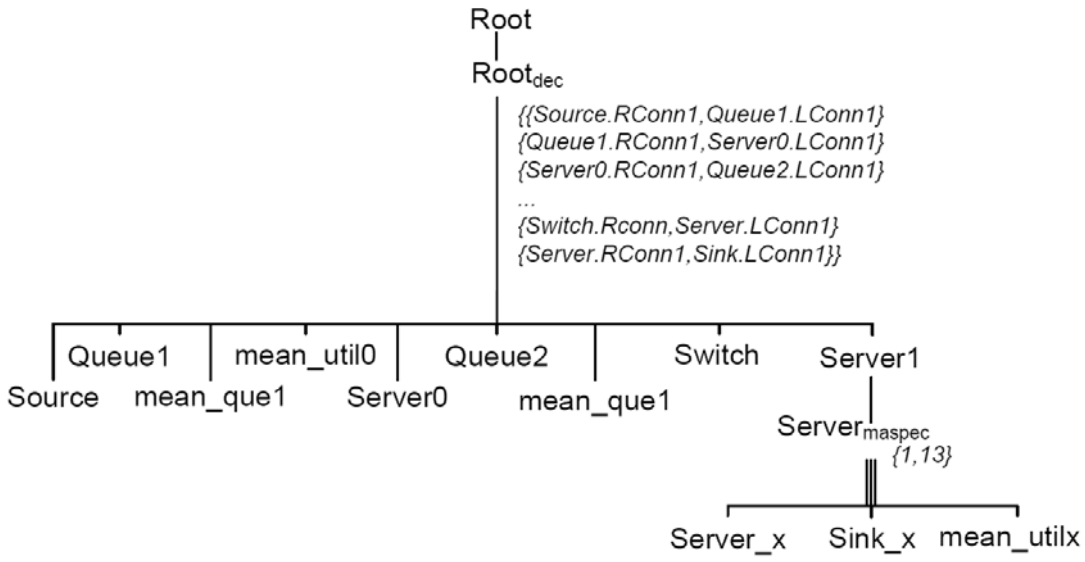

Figure 7. SES of the example application.

1. The Simulink block *Source* generates jobs with a variable arrival time within the uniform range 4..20s. The value is defined by the Simulink block parameter *Period*.
2. The *Switch* block distributes the jobs to a variable number of servers. The example in figure 6 shows 4 servers. The number is defined within the range 1..13 according with the SES in Figure 7.

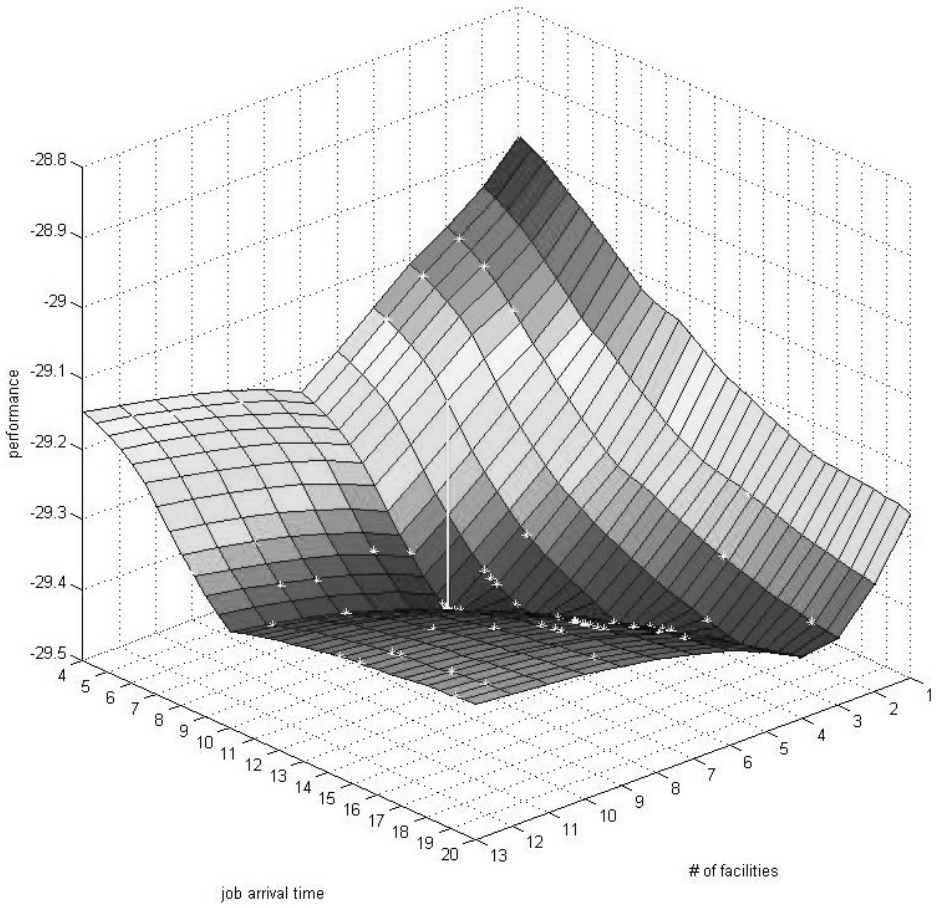

Figure 8. SES of the example application.

The *Server0* has a job handling time of 10s, all other servers have a handling time of 70s. The optimum of the system is defined as the maximal number of handled jobs in a given time.

Through the straightforwardness and the deterministic parameters of the example the optimal parameter and model structure can be taken out directly:

- job arrival time = 10s
  faster arrivals fill only *Queue1*
  slower arrivals cause *Server0* wait states
- number of server = 7
  fewer servers fill only *Queue2*
  more servers cause wait states of one or more servers

With the common procedure (see Figure 1) an analyst would repeat the cycle:

- automatic searching of the optimal model parameter for the job arrival time for a single model structure
- evaluating the model performance
- when the performance is not satisfying: changing the structure of the Simulink model and starting again
- … until finding a satisfying solution.

This method is time-intensive and don't guaranty the finding of an optimal solution. A further disadvantage is necessary profound knowledge of the analyst about the system to guess promising structure changes for the next step.

In contrast to this common principle, using the introduced approach an analyst doesn't implement the Simulink model but determines/creates basic blocks from the Simulink Library and specifies the SES which defines all model variants within a single meta-model. Figure 7 depicts the SES of the example.

As section 1.4 describes, the next step of the algorithm is analyzing the SES tree and generating information about the search space. The analysis finds one structure parameter at the decision node $Server_{maspec}$. Together with the continuous model parameter *job arrival time* the optimization method has to search in a two-dimensional search space. Figure 8 depicts the fitness surface within the search space. Each point on this surface corresponds to one specific Simulink model structure and Simulink Source block *Period* parameter value combination.

During the optimization phase the EA estimates the performance of randomly selected points on the surface to find the optimal solution. In Figure 8 dots on the surface illustrate analyzed points in one experiment. The figure also shows the found optimal model parameter and model structure combination with a line at the expected location: job arrival time = 10 and number of servers = 7. The experiment was repeated and the average number of simulated search space points is 240.

3 Conclusions and Further Work

This paper has introduced a simulation based combined structure and parameter optimization method. To achieve this aim the approach combines three established methods and extends established optimization techniques to the fundamental model structure to enable combined structure and parameter optimization.

Three main elements have been determined: (i) a model generating meta modeling technique based on SES/MB formalism, (ii) a MATLAB/Simulink based modeling and simulation environment, (iii) an EA optimization method. The introduced approach was implemented and tested in the Scientific and technical Computing Environment MATLAB/Simulink using the additional MATLAB toolboxes SimEvents and Global Optimization.

The results of the presented application show that the approach is able to find an optimal model structure with appropriate model parameters.

The complexity of the shown example is defined consciously small to demonstrate the introduced approach. In [1 2] examples with a higher complexity but another modeling and simulation method are described. The presented framework and its application are an intermediate result in the scope of the research project supported by the German Research Foundation.

References

[1] O.Hagendorf. *Simulation Based Parameter and Structure Optimisation of Discrete Event Systems.* PhD thesis, Liverpool John Moores University, 2009.

[2] O. Hagendorf, T. Pawletta, C. Deatcu, R. Larek: *An approach for combined simulation based parameter and structure optimization using evolutionary algorithms.* In: Proc. of The 7th EUROSIM 2010 Congress, Vol.2, Prag, Czech Republic, 2010

[3] T. Pawletta, C. Deatcu, O. Hagendorf, S. Pawletta, G. Colquhoun. *DEVS-Based Modeling and Simulation in Scientific and Technical Computing Environments.* In: Proc. of DEVS Integrative M&S Symposium (DEVS'06) - Part of the 2006 Spring Simulation Multiconference, Huntsville/AL, USA, 2006.

[4] J.W. Rozenblit, B.P. Zeigler. *Concepts for Knowledge-Based System Design Environments.* In: Proc. of the 1985 Winter Simulation Conference, San Francisco/CA, USA, 1985.

[5] B.P. Zeigler, H. Praehofer, T. G. Kim. *Theory of Modeling and Simulation.* Academic Press, 2000.

[6] G. Zhang, B.P. Zeigler. *The system Entity Structure: Knowledge Representation for Simulation Modeling and Design.* Artificial Intelligence, Simulation, and Modeling, Widman L.E., Loparo K.A., Nielsen N.R. (Ed.), John Wiley & Sons Inc, 1989.

[7] The Mathworks® (2011) *MATLAB®*. http://www.mathworks.com/products/matlab/.

[8] The Mathworks® (2011) *SIMULINK®*. http://www.mathworks.com/products/simulink/.

[9] The Mathworks® (2011) Global Optimization Toolbox. http://www.mathworks.com/products/global-optimization/.

Submitted: September 2011
Revised: February 16, 2012
Accepted: March 1, 2012