Generation of Structured Process Models Using Genetic Programming

Hartmut Pohlheim

Daimler Benz AG
Systems Technology Research
Alt-Moabit 96a
D-10559 Berlin
pohlheim@dbresearch-berlin.de

Peter Marenbach

Darmstadt University of Technology
Institute of Control Engineering
Department of Systems Theory & Robotics
Landgraf-Georg-Straße 4
D-64283 Darmstadt
mali@rt.e-technik.th-darmstadt.de

The design of structured mathematical models of processes in a certain level of abstraction defined by the given task appears to be difficult and time consuming even for experienced experts.

This paper reports on a new method for the design of structured process models based on the metaphor of Genetic Programming. This new methodology allows the automatic generation of non-linear process models in a self-organizing way.

Keywords:
genetic programming, modelling, system identification, process models, structured models, structure optimization, parameter optimization, control systems, learning control, industrial application, biotechnology, SMOG, MATLAB, SIMULINK

1 Introduction

The generation of dynamic process models is one of the fundamental tasks for the control engineer. The model implements knowledge about structure and dynamics of the system/the process and is the starting point for the design and configuration of controllers.

Two different classes of process models can be distinguished:

- structured mathematical models (transfer functions in the frequency domain, linearized and nonlinear differential equations),
- unstructured/metaphorical input/output models (neural nets, function approximation).

Structured process models are usually derived analytically using different identification algorithms. This requires special knowledge from the control engineer. For complex systems the analytical methods can be very complicated and time consuming or even impossible. However, structured models are well known and there exists a comprehensive theory for these models and the synthesis of controllers.

Unstructured input/output models are derived using learning/searching algorithms. They require the definition of relevant input and output variables only, thus work data driven. The more data are available, the more the model can be enhanced using the learning algorithm. On the other side there is no coherence between the internal structure of the derived model and the structure of the real systems. Thus, the transparency of the derived model is low or missing. This usually leads to problems concerning the acceptance of these approaches in industrial applications. Additionally, there exists no extensive theory for the analysis of the models and the design of controllers. However, often there exist only measured input/output data from the process and nearly no knowledge about the real system. Then there is no alternative to a learning approach.

In this paper we report on a system for the automatic generation of structured dynamic process models, the Structured Model Generator SMOG. The system uses data-driven learning, as known from unstructured input/output models and combines this with the transparency and well-known handling of block diagrams, the classical representation in control theory. Furthermore available knowledge about the process can be incorporated into the modelling.

Section 2 describes the fundamental idea and main aspects of the approach. Section 3 shows an example application in detail and reports on other industrial applications the system was used for. Section 4 summarizes the results and points out directions of future research.

2 Generation of Structured Models

Figure 1 shows the scheme of the system for the model generation. It mimics the way models are developed by an engineer. Starting with a collection of elementary elements (the construction set) a number of models is generated. In an iterative evolutionary process each model is adapted to the measured process data by optimizing it's internal parameters and evaluating the fitness by assessing accuracy and complexity. Directed by this fitness value new models are created by modifying and extending the actual's models structure.

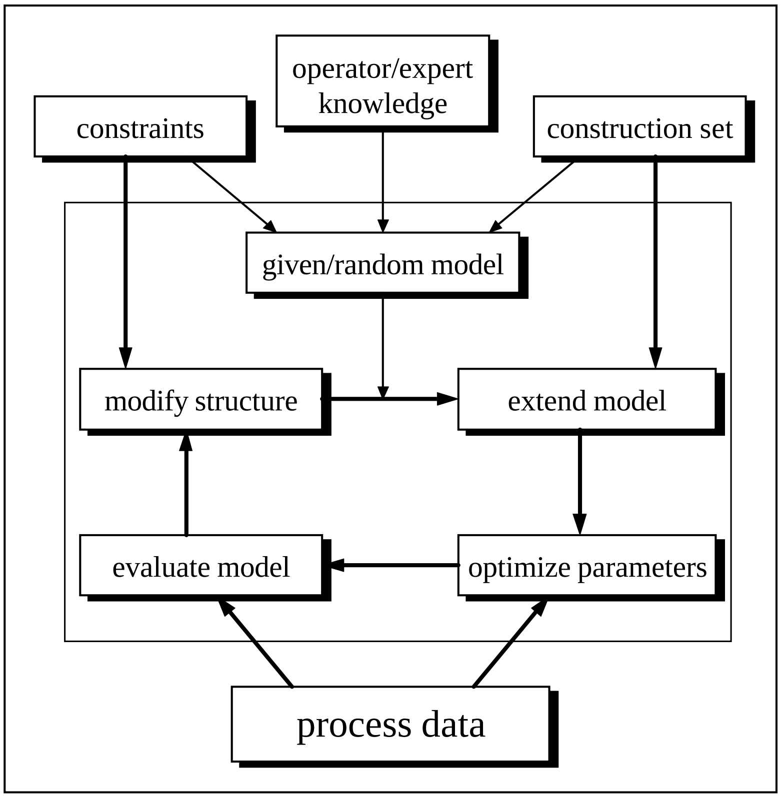

Fig. 1. Design of structured process models

This scheme allows the incorporation of operator knowledge and constraints as well as process data into the modelling process. The operator has different possibilities to input special knowledge and constraints into the process:

- constraining the elements in the construction set and their selection frequency,
- definition of known partial models („super blocks"),

- definition of constraints for the inner parameters of used blocks.

The used scheme of the system distinguishes between two tasks:

- symbolic generation of an appropriate model structure and
- optimization or identification of the parameters of the blocks of each model.

The parameter optimization is done by well known conventional optimization methods (Hooke-Jeeves [4], Rosenbrock [8]). First experiments were done with a genetic algorithm and an evolutionary strategy. Both produced superior results, however, the computational time was much higher than for one of the conventional methods.

The symbolic structure generation is done by means of genetic programming [5]. The distinction between parameter and structure optimization can be seen clearly in the structure of the implementation in figure 2.

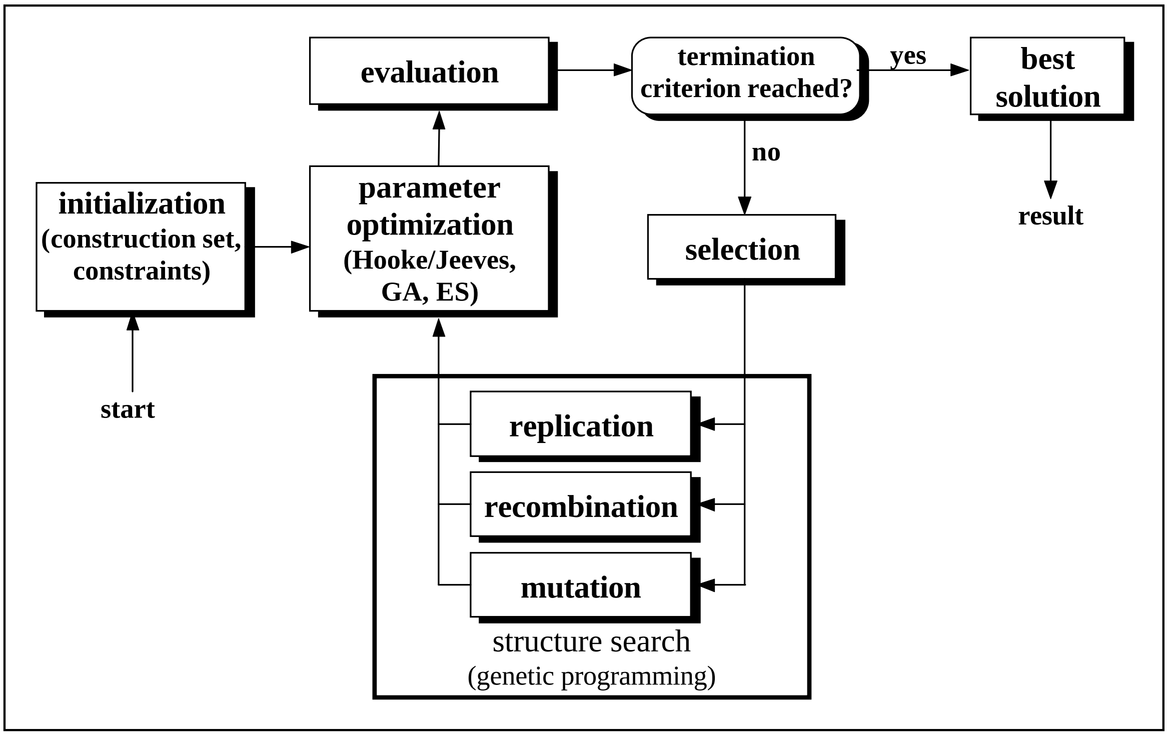

Fig. 2. Structure of the implemented program

Instead of using differential equations as commonly done in approximation of functional relations we decided to use block diagrams as they are commonly used in control theory to describe a system and it’s internal structure.

Thus, the construction set consists of:

- basic arithmetic functions,
- transfer function blocks of different order or dead time,
- non-linear elements like switches or limiters
- feedforward and feedback loops,
- domain-specific elements like Monod kinetics and sigmoid and gauss functions.

Figure 3 shows an example of a model depicted as a block diagram and as the equivalent genetic representation in a tree structure.

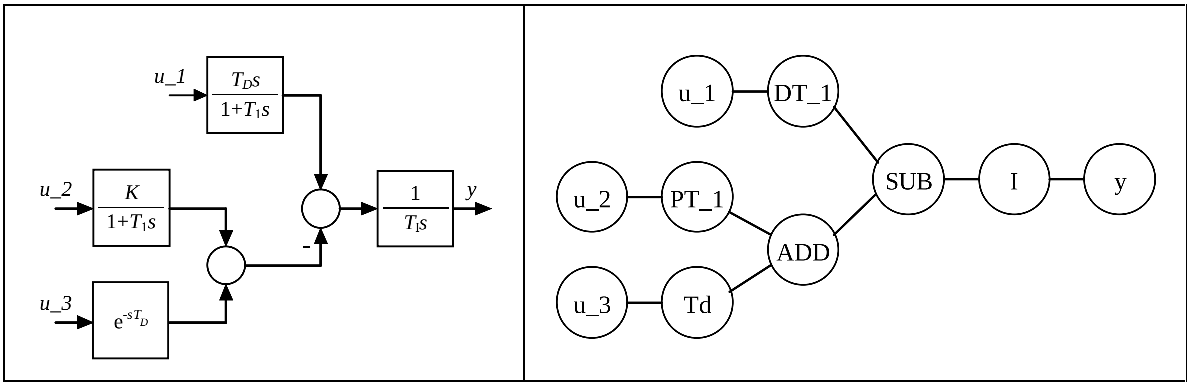

Fig. 3. Structured process model: block diagram and tree representation

During the optimization two concurring criteria have to be combined: high accuracy and low complexity of the derived model. That means the objective for the optimization process is not only to find a model that approximates the given data, but also to derive models of moderate and compact size. Therefore fitness of a model *i* is calculated by:

$$f_i = \frac{1}{1 + E_i \cdot C_i}$$

where E_i is the root-mean-square error and C_i is a heuristic complexity value. The complexity value is calculated as the sum of complexity values of all blocks of the model.

The simulation of the dynamic behaviour of each model appears to be the major time consuming factor during the optimization process. Due to the block oriented representation of the models the dynamic elements can be simulated using their Z-transform. Compared to the use of numerical integration methods this drastically reduces the required computation time and makes it possible to optimize real world problems.

The concept of self-organizing generation of process models SMOG was implemented in C++ for UNIX workstations. The evolutionary structure search and the parameter optimization were realized in separate programs implementing a scaleable functional parallelism with master-slave topology. The parameter optimization can be run in parallel on a number of workstations and provides a simple but powerful and flexible way to parallel computation.

3 Applications

To demonstrate the capabilities of the current version of the Structured Modelling Generator SMOG an example will be shown. In earlier work [1] the error between the models output and measured process data as well as the models plausibility was studied. This time a test system was designed which was directly compared with the structures of the models found by SMOG.

3.1 Artificial Non-linear Example

Using Matlab and Simulink, a non-linear system (original system) was designed. Figure 4 shows the block diagram of the test system used. It contains two inputs, one output, common used transfer function blocks, a feedbackward loop and a multiplication.

Using different input signals (sinus, square, sawtooth and noise in a frequency range of 0.03 rad - 0.60 rad) 20 data sets 250 data points each were generated. To produce more real world like data a random sequence was added to the output of the system (see figure 4). The amount of noise was in the range of 20% of the output signal.

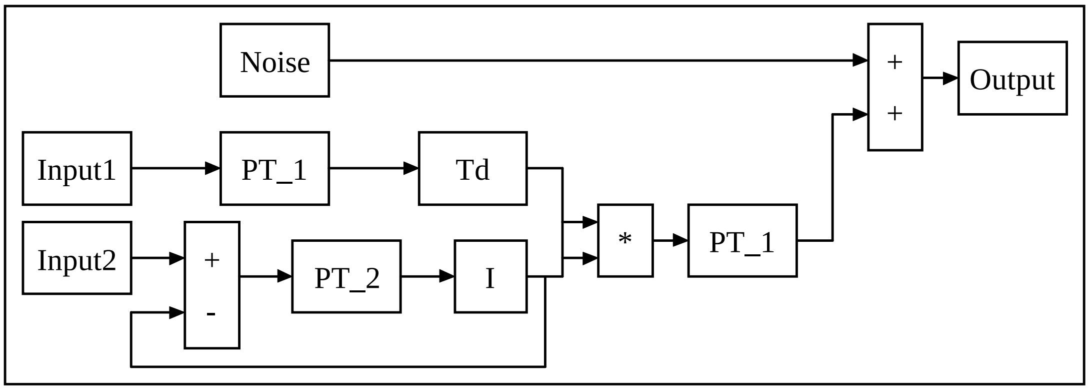

Fig. 4. Example system (Simulink)

5 data sets were used for the modelling. The optimization was run several times over 100 generations with 200 individuals using 200 iterations for parameter optimization each. Tournament selection with a tournament size of 2 combined with elitism and the intelligent mutation operator was used. The construction set contained: addition, subtraction, multiplication, division, P, I, PT1, DT1, PT2, PDT2, Td, feedforward and feedbackward.

The generated models were tested using the remaining 15 input data sets, however, here the output signal was not randomized, thus comparing the generated model with the original model.

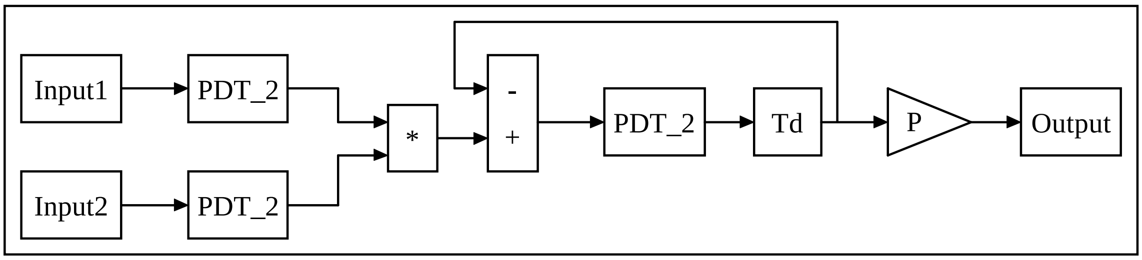

Fig. 5. Generated system (SMOG) for the example system in figure 4

Figure 5 and figure 6 show block diagrams of generated models producing a good match between the output signal of the original system (figure 4) and the generated model. These models - and all of the other generated models - are not completely identical to the original model. However, the structure of the derived model is similar and contained most of the blocks used in the original system.

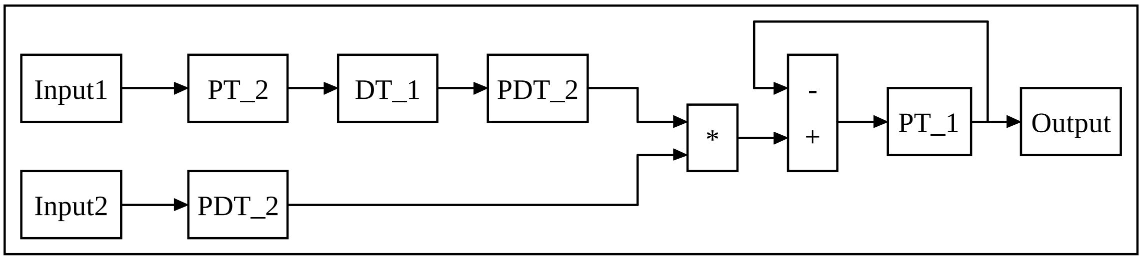

Fig. 6. Another generated system (SMOG) for the example system in figure 4

The quantitative comparison between original and generated model should be done between the output data of both systems. This is the only clue in the optimization process. If both signals match each other, the optimization was successful (even if the structure of the original system is different, since it is possible, that for the given input signals more than one system produce the same set of output data). Figure 7 shows 12 diagrams comparing the output signals of the original and the derived model (figure 5). The output signal of the generated model is plotted on top of the original model. Thus, the difference between both signals can be judged in how much of the original output graph is visible.

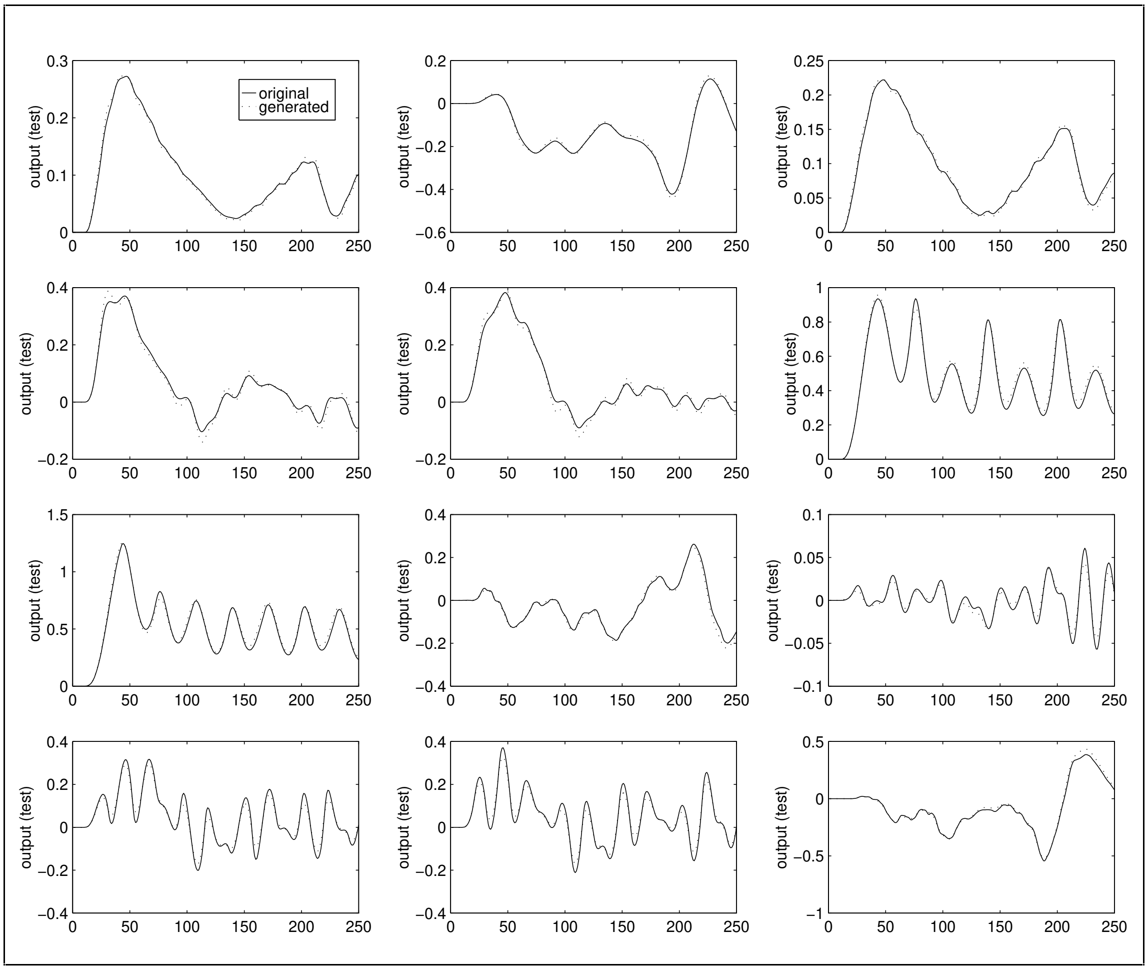

Fig. 7. Output of original and generated model

This example shows, that SMOG is able to find a structure for the data sets.

We used this comprehensive example system to give an impression of the power of SMOG and the possible results. As can be seen in figure 5 and figure 6 the generated systems didn't grow to an unmanageable size in number of used blocks. (This can be enforced by carefully setting the weights for error and complexity in the fitness function.) Even in the small size of the example system it is non-linear and thus difficult for many other identification methods.

3.2 Other Applications

The modelling tool SMOG was already used successfully in a number of industrial applications.

Daimler Benz AG is still in the process of evaluating the power of the generation of structured process models. Beside testing on different example problems SMOG was used for the modelling of an industrial steel rolling process. Because of the complexity of the process and the small number of available data this was a very hard problem. However, during the first experiments SMOG produced promising results of good quality and, very important, models of moderate size. This new approach opened a new way for modelling and understanding this problem. Different generated models can be evaluated by the control engineer gaining new insight. A second step can be a combination of the generated models with new knowledge to derive a superior model of the process, thus providing the basis for a much better control of the process.

At the TH Darmstadt the modelling tool was used for the modelling of a river barrage system [6]. The generated models were similar to the one known from experts. Further modelling couldn't be done because of lack of sufficient data. A second area in which SMOG was applied successfully is biotechnology, where an analytical development of structured models is extremely difficult due to incomplete knowledge on the underlying biological pathways. Within an industrial cooperation with BASF AG Germany SMOG was used for the modelling of a fed-batch fermentation process [1]. The derived models inspired the chemists and biologists during their design phase as well as they increased the transparency and understanding of the process behaviour. This is an essential feature of this approach that cannot be achieved by the use of, for instance, Neural Networks.

4 Concluding Remarks

We presented an application of the genetic programming paradigm for the design of structured dynamic process models. This approach combines genetic programming with well known and proven techniques of control and systems theory (block diagrams, Z-transform). The synthesis of these approaches is a powerful tool for data-driven modelling of complex processes.

The so derived Structured Model Generator offers many possibilities to integrate existing knowledge (expected elements, submodels, known elements).

The distinction between symbolic structure search (genetic programming) and the numeric parameter search (parameter optimization) combined with the scaleable parallel implementation within a workstation cluster increases the efficiency of the realized tool.

The derived process models provide a transparent insight into the structure of the process. This is an essential feature that can not be achieved by other unstructured modelling method. That means that this approach may overcome the difficulties that are bound to the use of adaptive or learning (e.g. neural networks) methods.

However, the generation of models by genetic programming cannot make the engineer unnecessary. Instead he is given a powerful tool to allow him to concentrate on creative considerations. By introducing concrete ideas and modifying experimental constraints he can guide the evolutionary process. The final decision which model appears to be most plausible for the desired purpose stays in the hand of the engineer.

The current implementation of the generation of structured process models is powerful and can be used for real world problems. However, a number of enhancements are possible and will be implemented and tested in the future:

- modelling of systems with multiple outputs,
- dividing the population in subpopulations as known from genetic algorithms ([7]),
- new parameter optimization methods (evolutionary strategies),
- multiobjective optimization ([2]) and
- intelligent genetic operators.

5 References

[1] *Bettenhausen, K.D., Marenbach, P., Freyer, S., Rettenmaier, H. and Nieken, U.*: Self-organizing structured modelling of a biotechnological fed-batch fermentation by means of genetic programming. First IEE/IEEE International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications, 12-14 September 1995, Conference Publication No. 414, pp. 481-486, 1995.

[2] *Fonseca, C. M. and Fleming P. J.*: Genetic Algorithms for Multiple Objective Optimization: Formulation, Discussion and Generalization. Proceedings of the Fifth International Conference on Genetic Algorithms and their Application, pp. 416-423, San Mateo, California, USA: Morgan Kaufmann Publishers, 1993.

[3] *Goldberg, D. E.*: Genetic Algorithms in Search, Optimization, and Machine Learning. Reading, Mass.: Addison-Wesley, 1989.

[4] *Hooke, R. and Jeeves, T.A.*: Direct search: Solution of numerical and statistical problems. Journal of the Association of Computing Machinery, pp. 212-224, 1961.

[5] *Koza, J.R.*: Genetic Programming: On the Programming of Computers by Means of Natural Selection. The MIT Press: Cambridge, Massachusetts, 1992.

[6] *Marenbach, P., Bettenhausen, K.D. and Cuno, B.*: Selbstorganisierende Generierung strukturierter Prozeßmodelle. at-Automatisierungstechnik 6 (1995), pp. 277-288, Berlin, 1995.

[7] *Pohlheim, H.*: Ein genetischer Algorithmus mit Mehrfachpopulationen zur Numerischen Optimierung, at-Automatisierungstechnik 3 (1995), pp. 127-135, Berlin, 1995.

[8] *Schwefel, H.-P.*: Numerical optimization of computer models. Chichester: Wiley & Sons, 1981.