Bargiela, A.; Kerckhoffs, E. (eds.): Simulation Technology: Science and Art – 10th European Simulation Symposium 1998. Society for Computer Simulation International, 1998, pp. 124–128

A MULTIAGENT SIMULATION OPTIMIZATION SYSTEM

Sven Hader
Department of Computer Science
Chemnitz University of Technology
D-09107 Chemnitz, Germany
E-Mail: sha@informatik.tu-chemnitz.de

KEYWORDS

simulation optimization, hybrid optimization, multiagent systems

ABSTRACT

Due to their increasing complexity, most modern technical systems cannot be described by mathematical means but only in form of simulation models. Optimization of simulation models is an intricate process whose success is mainly based on the selection of appropriate optimization techniques. In this paper, the multiagent system MASCOT for simulation optimization is introduced. It combines a set of optimization techniques into one self-adapting hybrid technique. Thus, it reaches a high degree of efficiency and robustness for a wide range of parameter optimization problems. To illustrate this, some results from the optimization of a job shop model are presented.

INTRODUCTION

There has been a growing trend towards increasingly complex technical systems in industry within the last years. The design of such complex systems makes great demands on the abilities of the engineer, since the systems usually have to be optimal according to some objective (costs, performance, etc.).

Many of the optimization problems that have to be solved in practice possess a number of special characteristics, like

- objective function and/or constraints cannot be formulated by mathematical means but only in form of simulation models
- there is a large number of design variables and of linear/nonlinear constraints,
- there is a variety of local optima,

that make the search for optimal solutions more difficult. To make things worse, there is only very limited time available for optimization so that only a small portion of the search space can be examined. According to those reasons, the selection of a suitable optimization technique is a crucial initial task in simulation optimization. Strictly speaking, the engineer has to solve a *meta-optimization problem* of the form

> *"find an optimization technique (as well as favorable strategy parameters) whose application to the given optimization problem provides the best (*weaker: *an acceptable) solution, taking into account the limited time and resources"*

prior to the original problem. Due to the variety of existing optimization techniques and the insufficient knowledge about their applicability, most engineers are incapable of solving this meta problem properly.

In this paper, the multiagent system MASCOT is introduced that combines a set of optimization techniques into one *hybrid* technique. Through the principles of cooperation and competition among agents this technique adapts *dynamically* to concrete problems during the optimization run. MASCOT is intended to support the engineer by solving the meta-optimization problem stated above as well as by automatically performing optimization runs. The paper is organized as follows. First of all, the use of hybrid optimization is motivated and some fundamentals of multiagent theory are introduced. Then, structure and functionality of the multiagent system MASCOT are described. Finally, the application of MASCOT is illustrated using an example from the domain of job shop scheduling.

HYBRID OPTIMIZATION

Often, it is advantageous to divide an optimization process into an "identification" phase for finding promising search space regions and a "local convergence" phase for converging on local extrema, cf. (Hart and Eldred 1996). Usually, there is no single (also called *atomic*[1]) optimization technique suited for both phases. Therefore, a *hybrid*

[1] Atomic techniques work homogeneously based on only one optimization principle (Syrjakow and Szczerbicka 1995).

optimization technique has to be used that combines different atomic techniques (generally local as well as global ones) during an optimization run. This hybridization exploits the strengths of the techniques used while avoiding their weaknesses. Thereby, a hybrid technique often reaches a higher degree of efficiency and robustness compared to the atomic techniques it consists of. Nevertheless, these advantages arise in many cases only from carefully designing the hybrid technique with regard to the optimization problem (or problem domain) given. To design a hybrid technique, two main questions have to be answered:

- Which atomic techniques should be used and what are their favorable strategy parameters (step sizes, termination conditions, etc.) ?
- How should the techniques interact (order of application, switch-over conditions, etc.) ?

Currently, there is a number of tools that provide the capability to apply hybrid techniques to optimization problems of practical interest, e.g., EnGENEous (Powell et al. 1991) and REMO (Syrjakow and Szczerbicka 1995). In general, the user of these tools has to create some kind of "optimization plan" (answering the questions mentioned above) before the automatic optimization run can be started.

MULTIAGENT SYSTEMS

Multiagent systems are a fundamental research discipline of Distributed Artificial Intelligence, see (Wooldridge and Jennings 1994; Müller 1996) for an introduction. A multiagent system consists of a collection of autonomous hardware or software systems, so-called *agents*. Agents respond to changes occurring in their environment (the "world" outside the agent) in which they may take into account their local state, knowledge, skills, plans, and goals.

A common problem in multiagent systems is to find the agent most suitable for a given task (*task allocation problem*). This will be trivial if the task can be solved by only one agent, but might become complicated if several agents can solve it. For example, there might be several agents capable of solving a system of equations, each using a different method. The *contract net protocol*, developed by Smith (Smith 1980), is one approach to overcome this problem. It applies a mutual negotiation process involving a *manager* (the agent that needs the solution) and a set of potential *contractors* (agents that can generate the solution). The process consists of the four main steps:

1. The manager sends a task announcement to all agents it assumes to be qualified.
2. Each of these agents evaluates, based on local knowledge of static and dynamic nature, whether and how well it can solve the task. If suitable, it sends a bid message to the manager.
3. The manager evaluates the bids received, chooses the best one, and establishes a contract with the corresponding agent (that thus becomes the contractor).
4. The contractor solves the task, e.g., by splitting it up into sub-tasks and announcing them to other agents, and finally sends the solution to the manager.

It is obvious that the solution of problems in this kind of multiagent system is not centrally controlled but "emerges" from the cooperation and competition of agents.

MASCOT: A MULTIAGENT SIMULATION OPTIMIZATION SYSTEM

MASCOT (**M**ulti-**A**gent **S**ystem for the **C**ombination of **O**ptimization **T**echniques) is a multiagent system for simulation optimization, see (Hader 1997, 1998). It is based on the new idea of realizing a self-adapting hybrid optimization technique through a negotiation-based multiagent system. MASCOT is intended to support the user by automatically solving a wide range of parameter optimization problems.

In MASCOT, the main software components are modeled as *software agents* (see Figure 1). They are represented by external programs that run in parallel and possibly distributed on a set of computers. Communication is performed via TCP/IP connections using the PVM system (PVM 1994). The advantages of this architecture are twofold. First, each software component can be executed on the hardware platform best suited, e.g., the user interface runs on the user's PC while the optimization is performed on a high-speed workstation. Second, software components can be replaced very easily, that is, the system is open to improvements and adaptations.

Each of the agents has a set of special skills (*talk to the user*, *execute simulation program*, *retrieve data*, *perform optimization*, etc.) and consists of a communication processor, a dialog processor, a task processor, and a local database (see Figure 2). The communication processor maintains the low-level communication with other agents, providing acknowledgement and time-out mechanisms. The dialog processor handles the existing dialogs

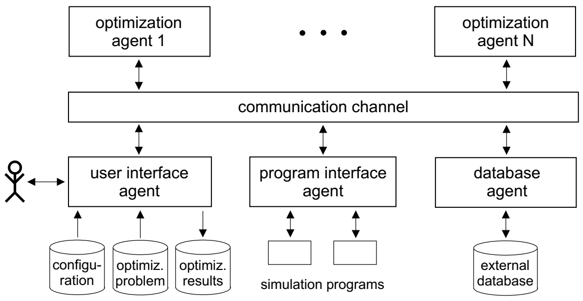

Figure 1: Structure of the MASCOT System

with other agents. Especially, it controls the negotiation process, that is, it evaluates announcements received, generates bids and own announcements, and establishes contracts. The task processor manages the tasks currently performed by the agent and provides a task scheduler. The local database stores the local knowledge of the agent, including information about its work load, former successes/failures on tasks awarded, and abilities and strategies for the generation and evaluation of announcements and bids.

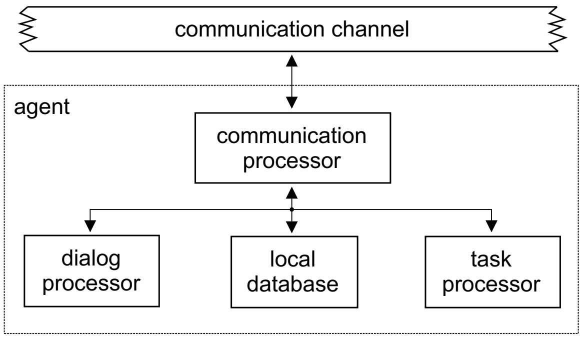

Figure 2: Structure of a MASCOT agent

MASCOT agents can be classified into the following essential agent groups:

- user interface agents
- program interface agents
- database agents
- optimization agents

User interface agents provide a man-machine interface between the user and MASCOT.

Program interface agents connect MASCOT to external (simulation) programs. They provide methods for data exchange and process synchronization. Currently, data exchange is performed via text files.

Database agents deal with storage and retrieval of data of global interest, like results of former simulation runs. They can be connected to external database management systems.

Optimization agents perform atomic optimization techniques of the types

- global optimization
  (e.g. Global Random Search, Genetic Algorithms)
- local optimization
  (e.g. Local Random Search, Steepest Descent Method)
- knowledge-based optimization
  (for the incorporation of problem-specific heuristics)

Before starting MASCOT, the user has to generate a text file containing a formal description of the current optimization problem. For this purpose, a special block-oriented description language exists that provides constructs for design variables, constraints, objective functions, program interfaces, and the like. For an example, see below.

```
data_objects
{ int x1, x2;
  float y;
}
parameters
{ x1 = 1 .. 1000;
  x2 = 0 .. 1;
}
objective_function
{ formula   = y;
  objective = minimize;
}
program
{ identifier = sim1;
  name       = "/home/sha/simulator";
  in_data    = { x1, x2 };
  out_data   = { y };
  interface  = i_face;
  in_file    = "/home/sha/simulator.in";
  out_file   = "/home/sha/simulator.out";
}
```

One of MASCOT's most important parameters is the maximum time the optimization run is allowed to take. Based on runtime statistics for the simulation program used, an estimation of the maximum number of simulation runs is calculated that is used in the selection of appropriate optimization techniques (see below).

The self-adapting hybrid optimization technique used in MASCOT is based on a negotiation process among the existing optimization agents (using the contract net protocol). Tasks of the form *"improve the best solution found so far"* are successively announced among the agents. Each agent evaluates whether and how well it can solve the task. The evaluation is based on a local strategy that takes into consideration the following essential information:

- classification of the optimization problem (number and type of design variables, nature of objective function and constraints, etc.)
- information gathered during the optimization (best solution found so far, former successes/ failures of the agent, etc.)
- maximum number of simulation runs for task solution
- current (and future) work load of the agent.

Based on this evaluation the agent may or may not generate a bid to the task announced. The best bid is chosen and the corresponding agent continues the optimization run until it finds a better solution or the maximum-number-of-runs limit is reached. Then, the next negotiation cycle is started.

MASCOT is currently in the stage of development and testing. To enter a stage where commercial use is possible, efficiency and robustness of the system have to be improved. Also, the user interface should be revised to meet the state-of-the-art in man-machine interaction.

AN EXAMPLE: JOB SHOP

MASCOT has been applied to several optimization problems based on deterministic and stochastic simulation models, mostly from the fields of warehousing and manufacturing. For a comparison of MASCOT's hybrid optimization technique with several atomic optimization techniques the following job shop like problem was chosen:

Consider a manufacturing system of m machines M_1, .., M_m. The production program consists of a set of n jobs J_1, .., J_n with predefined due times d_i. A technological order $O_i = (o_1, .., o_{l_i})$ of operations is assigned to each job J_i. Each operation can be processed on a subset of the machines, with setup and processing times being deterministic and machine-independent. Passing times of jobs between machines are neglected. Once released, the jobs are scheduled using a FIFO priority rule. Let c_i be the completion time of job J_i. The problem is to find a release time r_i^* from a predefined time interval $[\, r_{min_i}\, ,\, r_{max_i}\,]$ for each job J_i that minimizes the tardiness and earliness objective function

$$f(\mathbf{r}) = \sum_{i=1}^{n} (\alpha T_i + \beta E_i)$$

with

$$T_i = \begin{cases} c_i - d_i & : \quad c_i > d_i \\ 0 & : \quad c_i \leq d_i \end{cases}$$

$$E_i = \begin{cases} d_i - c_i & : \quad d_i > c_i \\ 0 & : \quad d_i \leq c_i \end{cases}$$

and α, β tardiness and earliness weights, respectively.

The example data were taken from the 1996 production program of a small engineering company in Saxony. A set of 60 jobs had to be processed on a collection of 66 machines operating on an one, two, or three shift basis (Monday to Friday). The number of operations per job lay between 4 and 19. In practice, tardiness is often more serious than earliness so that the weights $\alpha = 1$ and $\beta = 1/4$ were chosen. Objective function values were computed using a deterministic simulation model. Each simulation run took about 5 seconds on a Pentium-133.

The following table shows the results of the application of several optimization techniques to the problem described above (using a maximum of 10,000 simulation runs). Except for the heuristic search and MASCOT's knowledge-based optimization, the techniques were used in their "standard" form, that is, without special adaptation to the job shop domain. The case

$$r_i = r_{min_i} \quad \forall i = 1, .., n$$
$$f(\mathbf{r}_{min}) = 324,078$$

(each job is released as early as possible) was used as the starting point for optimization.

optimization technique	number of runs	best $f(.)$ found
Genetic Algorithm	10,000	152,975
Steepest Descend	2,045	245,424
Local Random Search	10,000	239,110
Heuristic Search	3,780	180,956
MASCOT	1,287	131,971

Although it is not possible to compare optimization techniques by the results of just one test problem, it can be seen that MASCOT found the best solution[2] of all techniques within the fewest simulation runs.

[2]Unfortunately, the optimal solution is not known due to problem complexity.

CONCLUSIONS

In this paper, the use of hybrid techniques in simulation optimization was motivated. It was stated that hybrid optimization techniques may reach a higher degree of efficiency and robustness compared to atomic techniques if designed properly. The two main questions in designing hybrid optimization techniques were named (Which atomic techniques should be used ? How should the techniques interact ?).
The multiagent system MASCOT was presented that combines a set of atomic optimization techniques into one self-adapting hybrid technique. Each optimization technique is represented by an agent that uses a local strategy for assessing its own suitability for a given optimization problem. This strategy is based on a classification of the optimization problem, the maximum available time for solution, and other information. An optimization run consists of a sequence of negotiations for "find a better solution" tasks, each followed by the application of the currently best optimization technique. MASCOT is intended to support engineers in solving parameter optimization problems that are based on simulation models. To illustrate the efficiency of MASCOT's hybrid technique, MASCOT and selected atomic techniques were applied to an example problem from the domain of job shop scheduling.

Despite the encouraging results obtained so far, there is still much work to be done. Further research should concentrate on improving efficiency and robustness of the optimization process. Since there is only very limited time available for optimization, information provided by the problem description (objective function, constraints, etc.) as well as the simulation runs has to be exploited more intensive. Apart from theoretical considerations, this goal can only be reached by incorporating new optimization techniques and performing experiments with a wide range of optimization problems.

REFERENCES

(References marked with * are available for download from `www.tu-chemnitz.de/~hader`)

Hader, S. 1997. "MASCOT: A Multiagent System for Hybrid Optimization." In *Proceedings of the Third ISIR Summer School on Inventory Modelling in Production and Supply Chains* (Ioannina, Greece). 155–165. *

Hader, S. 1998. "Ein Multiagentensystem zur simulationsbasierten Optimierung." In *Proceedings of the Workshop SiWis '98 - Simulation in Wissensbasierten Systemen* (Paderborn, Germany). Report tr-ri-98-194, Universität-GH Paderborn. *

Hart, W.E. and M.S.Eldred. 1996. "Hybrid Global/Local Algorithm Research." WWW page at `endo.sandia.gov/9234/sd_optim/f2_hybrid.html`.

Müller, J.P. 1996. *The Design of Intelligent Agents. A Layered Approach.* Lecture Notes in Artificial Intelligence 1177, Springer-Verlag, Berlin, Heidelberg, New York.

Powell, D.J.; M.M.Skolnick and S.S.Tong. 1991. "Interdigitation: A Hybrid Technique for Engineering Design Optimization Employing Genetic Algorithms, Expert Systems, and Numerical Optimization." In *Handbook of Genetic Algorithms.* L.Davis, ed. Van Nostrand Reinhold, New York, 312–331.

PVM. 1994. *PVM Users Guide and Reference Manual.* Oak Ridge National Laboratory, Tennessee, USA.

Smith, R.G. 1980. "The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver." *IEEE Transactions on Computers* 29, no.12 (Dec.): 1104–1113.

Syrjakow, M. and H.Szczerbicka. 1995. "Simulation and Optimization of Complex Technical Systems." In *Proceedings of the SCSC'95* (Ottawa, Canada). SCS, 86–95.

Wooldridge, M.J. and N.R.Jennings. 1994. "Agent Theories, Architectures, and Languages: A Survey." In *Proceedings of the ECAI-94 Workshop on Agent Theories, Architectures, and Languages* (Amsterdam, The Netherlands). Lecture Notes in Artificial Intelligence 890, Springer-Verlag, Berlin, Heidelberg, New York, 1–39.

BIOGRAPHY

Sven Hader (* 1969 in Mühlhausen, Germany) studied computer science at the Chemnitz University of Technology in Germany where he received the M.Sc. degree in 1993. Since then he has been a research assistant at the Department of Computer Science in Chemnitz. At present, he is particularly involved in a project to establish a center for logistics and simulation for small and medium-sized companies. Also, he is preparing his Ph.D. thesis on the use of multiagent systems in simulation optimization.