Matthias Gruber (2010)

A Generic Framework for Discrete Simulation Based Optimization

Abridgment

The aim of this work is to provide generic, performant software tools for simulation based optimization. The object oriented solution will consist of two main software libraries. The first library is a framework for the implementation of discrete simulation models, including some simple models for demonstrations and comparisons. The second library is a framework for the problem-independent implementation of optimization algorithms with a ready-to-use genetic algorithm to demonstrate the technique.

Discrete Simulation Framework

There are a number of potential hazards in simulation based optimization, some of which originate in the simulation engine itself and shall be avoided in this work. One of these hazards lies in the way random numbers are generated for stochastic simulation experiments: Large models (>> 1.000 entities) are often built dynamically using databases for the configuration and / or parametrization. Dynamic model creation is also required in optimization of layout configurations, no matter the size of the model. In in this context, reproducibility has to be considered very carefully. If the order in which model components are created is not fixed due to the ongoing optimization, unstable sorting algorithms in the model initialization code or due to DBMS / ORM related reasons, the way random numbers or seeds are distributed between simulation entities also changes. The same problem arises if entities with stochastic properties are created dynamically during a simulation run and the order of creation changes due to a changed parameter set which is not an uncommon scenario in optimization. This may cause only subtle changes in the outcome which will not be noticed and lead to invalid experiment results. The problem is of particular relevance during optimization where single simulation experiments run unattended in the background. Therfore a mechanism has to be implemented so that random numbers / seed values can be linked to identifiers.
A very similar problem may arise in the ordering of events, if they are not prioritized explicitely (the engine allowing). Because of that a simulation engine for use in optimization should differentiate between events, event instances and event handlers (similar to the way this is done in programming languages like C#) and allow fine grained priorization in all three layers.

Heuristic Optimization Framework

Usually, in optimization either the target function's inputs and outputs have to be adapted to match the optimization algorithm's data structures or the optimization algorithm has to be adapted to the target function. This problem shall be alleviated in the proposed framework by making use of the generics paradigm present in current programming languages and by employing a sophisticated set of interfaces for both the target functions and the solution algorithms. These interfaces should additionally allow for the tuning of optimization algorithms where this is applicable.

Expected Benefits

The proposed framework will make it possible to design and debug simulation based optimization solutions using Java and different .NET languages with the support of modern development environments. The use of modern, object oriented programming- and software development paradigms and tools like unit testing, contract based programming, generics, lambda expressions, profiling, plugin architectures and others should make the development of simulation models and optimization algorithms less error prone and produce more scalable and highly performant results.