Proceedings of the 2012 Winter Simulation Conference
C. Laroque, J. Himmelspach, R. Pasupathy, O. Rose, and A.M. Uhrmacher, eds

A FRAMEWORK FOR AGENT-ORIENTED PARALLEL SIMULATION OF DISCRETE EVENT SYSTEMS

Tao Zhang
Oliver Rose

Universität der Bundeswehr München
Department of Computer Science
D-85577 Neubiberg, GERMANY

ABSTRACT

Event-oriented serial simulation is a major method for discrete event systems while real-time parallel simulation is used mainly in agent-based models. Combining the advantages of these two methods, an agent-oriented parallel simulation approach using process interaction worldview is proposed on the basis of an agent-based model. Activations and delays are conveyed in the form of messages between simulator and agents. The simulation clock advances in a sequence of activation points. All concurrent activations are sent to agents at a time and associated agents respond in parallel. A brief framework is developed by means of multi-threading and synchronization technology and applied to analyze a queuing system where the results show the validity of the proposed method.

1 INTROUDCTION

In the ABM actions are taken by autonomous agents following their inner clocks without external instruments. In essence they show a behavior of a real-time simulation which is less efficient than event-oriented simulation. A general view is that event-oriented simulation is against the autonomy principle of the agents due to centralization of event handling. However, event handling is only in charge of simulation time which is independent and never effected by agent. Therefore, it is possible to apply the event-oriented simulation paradigm to ABM. Process interaction worldview is one type of event-oriented simulation method for DES, in which the notion of "process" corresponds closely to lifecycle of agent. In this paper, an agent-oriented parallel simulation using process interaction worldview is proposed and a corresponding framework is developed.

2 AGENT AND AGENT-BASED MODEL

An agent is composed of attributes, behaviors, a behavior controller, and a message handler. Attributes are characteristics of the agent. The agent adapts to the environment by changing its attributes. Behaviors are the activities that an agent does with purpose. The purpose of the behavior is either to change itself or to change other agents. The behavior controller is built for controlling behavior, such as adding behavior, removing behavior and changing behavior's state according to the environment. Through the message handler, the agent communicates with other agents. A message handler includes one message queue and two methods, "send" and "receive". The message has a given format and contains typically sender, receivers, send time, key words, and content. An agent-based model includes the agent environment, the agent manager, and a set of distributed agents. The agent environment is a medium for communication among the agents. A new agent needs to register with the manager and reports its physical state.

3 PROCEDURE OF AGENT-ORIENTED PARALLEL SIMULATION

Firstly, two concepts, delay and activation point are given. Delay is a period of time in which the logical state of agent is not changeable. When the delay occurs, the agent will create an activation point and stop

978-1-4673-4781-5/12/$31.00 ©2012 IEEE

running. There are two types of delay, conditional delay and non-conditional delay. Activation point is the time position where the agent will be after activation. The agent is activated at this point and performs actions until a new delay will occur. An activation point has such a given format including activation time, activation agent and key words.

The simulation time advances according to the earliest activation point. All concurrent activation points are activated at a time and associated agents respond in parallel. When the **model state** is ready (see definition in section 4), the simulation time advances to the next earliest activation point. After the model state is ready again, the remaining conditional activation points need to be activated repeatedly until the model does not move on anymore. We use a simulator to manage activation points and control the simulation time. The simulator is also developed as an agent. Activation points are conveyed in the form of messages between simulator and agents. Three activation point lists are created in the simulator: conditional activation list, future activation list and current activation list. The simulator puts new received activation points into the appropriate list. The earliest activation points are moved from the future list to the current list every time the simulation time advances.

4 A BRIEF FRAMWORK FOR AGENT-ORIENTED PARALLEL SIMULATION

A framework for agent-oriented parallel simulation is developed in the Java programming language. In the framework, agents are developed by means of multi-threading to facilitate to run in parallel to other agents. Two physical states, active and blocked are provided for individual threads. While a delay occurs in the agent, the thread will be **blocked** and gives up control of the CPU. After the delay, the thread becomes **active** again and starts running. The agent environment is implemented as the data shared by the agent threads. The agent manager provides the model state to the simulator. Two model states are implemented: ready and not ready. If the model state is **ready**, the physical states of all agents are blocked. If there exists one agent where the state is active, the model state is **not ready**. Communication among agents is achieved by sharing data between threads. The safety of the data sharing is guaranteed by synchronization technology. The communication and behavior control are already considered in the framework. What the user needs to do is only agent abstractions, behavior definitions, and cooperation design and activation point selection.

5 APPLICATION AND CONCLUSION

The queuing system $M/M^r/1$ with batch service has a lot of concurrent activations and is therefore well suited to validate the approach. Three types of agent are abstracted from the queuing system: customer source, customer, and server. The customer source generates customers. The queue is part of the server agent. Because a server is also an active entity, activation points of the whole system are simplified to two types: customer arrival and service completion. The simulator runs 1000 days and spends 54 seconds which is naturally less than the time spent by the general real-time ABS. The comparison between theoretical values and statistical values of simulation results shows that the simulation results match the theoretical results very well.

In another poster (Zhang and Rose 2012), the proposed framework is applied to the simulation of production processes which is a more complex queuing network. Nearly 3000 agents are working simultaneously. The efficiency of the simulation is still acceptable.

REFERENCES

Zhang, T., O. Rose. 2012. “Developing an Agent-oriented Parallel Simulator for Production Processes”. In *Proceedings of the 2012 Winter Simulation Conference*, Edited by C. Laroque, J. Himmelspach, R. Pasupathy, O. Rose, and A.M. Uhrmacher, -. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.