Proceedings of the 2014 Winter Simulation Conference
A. Tolk, S. Y. Diallo, I. O. Ryzhov, L. Yilmaz, S. Buckley, and J. A. Miller, eds.

USING IMPRECISE COMPUTATION FOR VIRTUAL AND CONSTRUCTIVE SIMULATIONS

Jonas Mellin
Cesar Tadeu Pozzer
Ilona Heldal

University of Skövde
Box 408, 541 28
Skövde, SWEDEN

Per M. Gustavsson

Combitech/Saab
Box 46, 541 21
Skövde, SWEDEN

ABSTRACT

In this work, we raise three critical questions that must be investigated to ameliorate composability of virtual simulation models and to enable adoption of systematic and stringent real-time techniques to enable more scalable simulation models for virtual and constructive simulation. The real-time techniques in question enable us to separate between policies and mechanisms and, thus, the simulation engine can decide dynamically how to run the simulation given the existing resources (e.g., processor) and the goals of the simulation (e.g., sufficient fidelity in terms of timing and accuracy). The three critical questions are: *(i)* how to design efficient and effective algorithms for making dynamic simulation model design decisions during simulation; *(ii)* how to map simulation entities (e.g., agents) into (real-time) tasks; and *(iii)* how to enable a divide and conquer approach to validating simulation models.

1 INTRODUCTION

In virtual simulation (e.g., driving training), we often have to deal with complex large-scale scenarios with sufficient details to make the training as useful as possible. *Trainees* participate in training sessions following scenarios that are run by trainers supported by an environment. A critical aspect of a training environment and the scenarios is the fidelity with respect to anticipated situations.

Dynamic handling of design decisions during simulation is a key issue to balance between achieving the goals of the simulation support and use the available resource in the best way. In virtual simulation, the motivation is that the trainees focus is limited and, thus, only a part of the simulation must be simulated with a high degree of fidelity and the rest can be simulated with less degree of fidelity. For example, in driving training, only the *computer manikins* (i.e., simulation-controlled vehicles) close to the *avatars* (representing the trainees) are interesting to be simulated in more detail compared to distant computer manikins. Further, it is possible to run seemingly large simulations on scarce resources such as surf boards. This dynamic handling of design decisions ameliorates the effect of static design decisions that may impact the trainers ability to develop useful scenarios. In real-time systems, this is done by *(i)* separating policies (e.g., only simulate entities in focus of the trainee) and *(ii)* mechanisms (e.g., best-effort scheduling of activities on resources to ensure that the most critical activities are performed at the right time).

The aim is to study if and how systematic and stringent handling of dynamic design decisions in the simulation model can be used to improve the scalability of resource usage of the simulation model. **A critical question** is if there are resource-efficient algorithms for making dynamic design decisions so that the simulation scales better with respect to the fidelity requirements compared to a simulations based on static design decisions. For example, is there a scalable way to determine (quickly) with limited and scarce resources what parts of the simulation that are in focus?

If we can use existing methods such as scheduling of tasks to resources (e.g., processors) for imprecise computation (Liu et al. 1991), which enables dynamic design decisions in real-time systems, in a useful way in virtual simulations, then this technique can either be used to get more fidelity or use cheaper hardware to run the simulation. Essentially, there is an opportunity to allow greater freedom in planning simulations as well as improving scalability of the simulation model.

2 SOLUTION STRATEGY

Scheduling for imprecise computation requires that the simulation model is somehow executed as a set of *tasks* (that are single-threaded algorithms computing some response to an event), where each task is associated with a temporal scope (e.g., periodicity, deadline) as well as alternative algorithms (e.g., the simulated decision making of close computer manikins can be more complex and resource-demanding compared to distant manikins). More elaborate examples are available (Pozzer et al. 2014).

Agent-oriented simulation is chosen as the main strategy due to the following features: *(i)* handling of avatars and computer manikins maps well into agents; *(ii)* agents are implementable as tasks as well as *(iii)* agent-orientation is useful for validation of the simulation model. The adaptability of feature (*ii*) is due to that agents can be viewed as a set of tasks that responds to changes in the simulation state by updating the state. Each task should have a temporal scope as well as possibility to specify alternative algorithms and their resource requirements to enable scheduling for imprecise computation. Concerning validation (feature (*iii*)), agent-oriented simulation enables us to apply a divide and conquer strategy where the parts are validated under the assumption that if all parts are valid, then the whole model holds.

The critical question in feature (*ii*) is how agents should be mapped into tasks. For example, we can partition agents with respect to computer manikins and avatars, where tasks can represent sets of agents (computer manikins and avatars). Further, if we use a layered approach separating between strategies, tactics and operation of the computer manikins' decision making/behavior, then we can partition with respect to these layers (e.g., a task can represent a set of layers). A hybrid approach is to use both these perspectives (e.g., a task can represent a set of manikins from the perspective of a set of layers). Essentially, it should be possible to specify alternative behaviors in a configurable and verifiable way as well as in a way that supports efficient handling of dynamic choices of alternative algorithms.

The critical question in feature (*iii*) is under what circumstances does the assumption hold and, thereby, allowing us to apply a divide and conquer approach to validating the simulation model. Essentially, if we can demonstrate sufficient fidelity in all situations by considering that the agents and their tasks do not break critical properties, then this holds. For example, if we know that deadlines will be met for tasks where disbelief in the simulation model is imminent if things do not happen at the right time, then we know that fidelity holds from this aspect. We plan to adopt the MARS approach (Kopetz et al. 1989) simplifying validation significantly (due to periodic tasks, state messaging, timely distribution).

ACKNOWLEDGMENTS

Cesar Tadeu Pozzer has been granted scholarship from CNPq, from CISB, and from Saab AB.

REFERENCES

Kopetz, H., A. Damm, C. Koza, M. Mulazzani, W. Schwabl, C. Senft, and R. Zainlinger. 1989. "Distributed fault-tolerant real-time systems: the MARS approach", *IEEE Micro* 9:25–40.

Liu, J. W. S., K-J. Lin, W-K. Shih, and, A. C. Yu. 1991. "Algorithms for Scheduling Imprecise Computations", *IEEE Computer* 24:58–68.

Pozzer, C. T., J. A. Amorim, P. M. Gustavsson, J. Mellin, I. Heldal, and A. T. Azevedo. 2014. "Imprecise Computation as an enabler for Complex and Time Critical HLA Simulation Networks", In *Proceedings of Simulation Interoperability Workshop.* To appear <www.sisostds.org>.