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

ENHANCING UNDERSTANDING OF DISCRETE EVENT SIMULATION MODELS THROUGH ANALYSIS

Kara A. Olson and C. Michael Overstreet

Department of Computer Science
Old Dominion University
Norfolk, VA 23529-0162

ABSTRACT

Simulation is used increasingly throughout research, development, and planning for many purposes. While model output is often the primary interest, insights gained through the simulation process can also be valuable. Insights can come from building and validating the model as well as analyzing its behaviors and output; however, much that could be informative may not be easily discernible through these existing traditional approaches, particularly as models continue to increase in complexity.

This research extends current work in model analysis and program understanding to assist modelers in obtaining more insight into their models and the systems they represent. Results indicate these tools and techniques, when applied to even modest simulation models, can reveal aspects of those models not readily apparent to the builders or users of the models.

1 SUMMARY

The automated analysis of model specifications is an area that historically has received little attention in the simulation research community but which can offer significant benefits. This is particularly true for analysis intended to provide modelers and model users addition information about their models. Since a usual goal in simulation is enhanced understanding of a system, model analysis can provide insights not otherwise available. This work developed new approaches for the simulation community to complement current methods used to gain insights into models, their behaviors, and the systems they represent.

Different analysis techniques can yield different potential discoveries. With static analysis, an object (such as code or a list of specifications) is analyzed without executing it; with dynamic analysis, data is collected during execution of the object of interest (normally code).

Static analysis can often reveal characteristics of a model not readily apparent from observing merely its run-time behavior. No finite number of runs can necessarily discover what is possible; however, from static analysis, one can reveal the possibility of infrequent situations.

From static analysis, one may discover that event A can appear to cause event B, but dynamic analysis often can reveal specifically which events caused which events, which cannot always be determined prior to run-time. In combination, if static analysis suggests that event A can cause event B, but dynamic analysis reveals that this does not happen, this may be of interest.

Results indicate these code analysis techniques, when applied to even modest simulation models, can reveal aspects of those models not readily apparent to the builders or users of the models. These analyses can often reveal important aspects of systems that are not readily observable in model-driven animations or in examining data produced by simulations during execution. This work has provided both model builders and model users with additional techniques that can give them improved understanding of their models not otherwise available.

The contribution of this research is the creation and presentation of automatically-derived observations that could potentially enhance a modeler or model user's understanding, that does not necessitate that the

modeler have programming expertise or even a technical background. A significant point of this research is that the created tools do not necessitate that a modeler or model user be able to encode the model or have any coding expertise. While some of the information presented here could be produced by existing software development tools, most modelers today do not have the technical background to use these tools or to make use of the reports such tools can produce. Continuing, one of the key aspects here is the focus on *model* aspects rather than *simulation* aspects. As modeling and simulation continues to be used increasingly often in research and as models continue to increase in complexity, these types of tools will continue to increase in value.

Automatic tools have been created and demonstrated to reveal new insights into models and the systems they represent.

A simulation log is generated, without any user action or programming effort, that notes each action and the simulation time; an additional printer-friendly version is also created.

"Trip lines" concern any boolean expression of model variables of which the modeler wants to be notified the first time it becomes true – for example, if a queue length becomes greater than 10 or a wait time becomes greater than one hour. Two options are available: a trip line that can be tripped exactly once or one that can be tripped and reset under specified conditions.

A list is generated of all scheduled, unscheduled, triggered, and untriggered events.

Total simulation time and a summary with respect to each event are tallied and presented for each simulation run. For each event (action cluster) in the run, its number of occurrences, events scheduled, number of times scheduled, events triggered, and number of times triggered, are presented.

The static action cluster interaction graph is generated, showing which events can cause which events.

The dynamic action cluster interaction graph is generated, with edges labeled according to event frequency during a given run.

And, a dynamic action cluster interaction graph is created for every time step of the simulation run and combined in to a multi-page document, providing a visual representation of the entire simulation run.

The work described here provides modelers with new views of their models; significantly, these views can be generated without additional work or knowledge on modelers' part. In the simulation community, little work had been done on exploring automatic generation of different views and representations of existing models, especially to the extent presented here. These new views can provide additional insights into models and the systems they represent.