Support of Event-Graph Modelling by the MMT E-learning System

Martin Bicher[1] , **Gasper Music**[2], **Irene Hafner**[3], **Felix Breitenecker**[1]

[1] *Institue of Analysis and Scientific Computing, Vienna University of Technology, Austria*
martin.bicher@tuwien.ac.at
felix.breitenecker@tuwien.ac.at
[2] *Faculty of Electrical Engineering, University of Ljubljana, Slovenia*
gasper.music@fe.uni-lj.si
[3]*dwh GmbH Vienna, Austria*
irene.hafner@dwh.at

Abstract

Due to steadily increasing industrial requirements a proper education in mathematical modelling is more and more often a prerequisite finding technical professions. Thus universities are required to provide high quality lectures and teaching materials to guarantee the qualification of their graduates within their field of study. For engineering students it is important to learn basic concepts of discrete-event modelling, as the understanding of event-operation and scheduling provides the base for designing management and control strategies. Very often the concept of event-graph modelling is used to teach the basis of discrete-event simulation in a graphical manner. The paper gives an idea how methods of event-graph modelling can be taught to students using E-learning concepts. In this special case the E-learning platform MMT (Mathematics, Modelling and Tools) is used, which can be applied as a virtual laboratory. The basic ideas of event-graph modelling, how event-graph based examples are prepared for the E-learning platform and the basic properties of the MMT-server are presented in this paper. Simple examples are shown in addition.

1 Introduction

Especially for engineering students nowadays a proper education in discrete-event modelling is very important as it provides the base for designing management and control strategies. So universities are challenged to offer high quality lectures in order to guarantee that the qualification of graduates satisfies the requirements of industrial professions.
Regarding discrete-event simulation one of the most challenging parts of the modelling process is the formal description of the model, which is on the one hand essential for the implementation and on the other hand necessary for a comprehensible documentation of the modelling process. One of the most figurative description forms for these kind of models is the so called event-graph modelling, formally described in section 2, which are very commonly used within queueing-theory.
Event-graphs are a description form for either stochastic or deterministic dynamic discrete event models. Though they are very simple to understand they are yet very flexible and really complex behaviour can be observed.
In order to support lectures dealing with event-graphs targeted E-learning should be used. Commonly used E-learning platforms usually provide the opportunity to offer teaching material and sometimes create tests with multiple-choice or numerical questions, where students can proof their knowledge. However, regarding the pictorial representation and the complex and maybe stochastic behaviour of event-graph models, these opportunities might not be sufficient. Therefore the E-learning system MMT (Mathematics Modelling and Tools) system, in detail explained in section 3, was developed based on the idea, that students can additionally experiment on pre-implemented, accurately tested and high quality simulation examples, by changing certain parameters in a browser window. The model is calculated on a fast server in behind and the results are presented either in textual, image or video form directly within the browser window.
To integrate user friendly event-graph examples to the MMT server, a new event-graph library was developed in MATLAB, which is roughly explained in section4.

2 Event-Graph Modelling

Already mentioned event-graph modelling is one of most common ways to describe a discrete-event model in figurative form. First, those models have to fulfil the following conditions in order to be described by an event-graph.

- Occurrence of a finite number of events occur during simulation-time.

- A finite number of observed state variables remain constant during the time between each two sequential events. Thus the variables change their value only directly at the events.
- The model is dynamical - i.e. events are taking place at a certain time.
- The time when an event takes place can depend on parameters, random variables and the state variables itself. Thus the so called event-list, containing the set of events to come, is updated each time the variables change.

The first two conditions are based on fundamental rules for discrete-event simulation. Usually the set of events can be clustered by condition of occurrence and change of the state variable. Events caused by the same conditions and causing the same change of the state variable are hereinafter clustered and called event-type.
Fulfilling the conditions a discrete-event model is defined by a finite set of possible event-types, linked to conditions when they occur and how they change the state variables, and a finite set of state variables with corresponding initial conditions at $t = t_0$. These ideas inspire defining so called nodes, usually represented by circles - one for each of the event-types. Below the node usually the impact of the event-type on the state variable is quoted. The causal relationship between the events is defined by so called edges, represented by directed arrows, connecting the nodes. Conditions for the occurrence of an event are quoted directly above the arrow. Before these ideas are introduced using a simple example, the formal description of an event-graph is given (see also [1]).

2.1 Formal Description of an Event-Graph Model

To present the formal description we mainly use the notation of Yucesan and Schruben 1992 [4].
Formally a quadruple
$G = (V(G), E_s(G), E_c(G), \Psi_G)$ is called event-graph or simulation-graph if

- $V(G) = \{E_1, \ldots, E_k\}$ is a finite set of event-types (called event-vertices),
- $E_s(G) = \{s_1, \ldots, s_{n_s}\}$ is a set of scheduling edges,
- $E_c(G) = \{c_1, \ldots, c_{n_c}\}$ is a set of cancelling edges (For modelling reasons $E_s(G) \cap E_c(G) = \emptyset$ holds. The set $E(G) := E_s(G) \dot{\cup} E_c(G)$ shall be called the set of all edges of the event-graph.),
- $\Psi_G : E(G) \to V(G) \times V(G)$ is a, not necessarily injective, function, called incidence function, assigning an ordered pair of vertices to a given edge.

Furthermore the edges, each linked to two vertices by the incidence function, are responsible for the causal relationship between those two events. They denote if, how and when the second of the two events is scheduled (or cancelled from the schedule) after the occurrence of the fist one. So far the basic construct of the event-graph is defined which does not contain any definition how state changes take place.
Let $\Omega \in \mathbb{R}$ be the so called state-space, containing all possible states of the model. Furthermore a model M based on an event-graph G is called event-graph model if the following Functions seen in Table 1 are defined and used. Finally the seven-tuple $M = (\mathcal{F}, C, \mathcal{T}, \Gamma, \mathcal{P}, \mathcal{A}, G)$

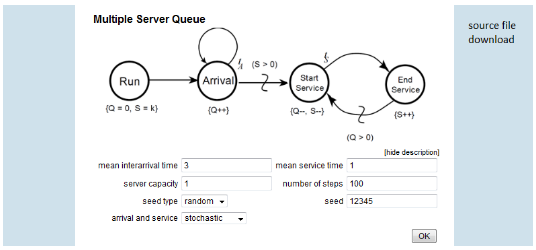
Table 1: Defining Functions for an Event-Graph Model

Function	**Use**
$\mathcal{F} : V(G) \times \Omega \to \Omega$	Specifies the state change for each event-type.
$C : E(G) \times \Omega \to \{0, 1\}$	Specifies the condition of the edge.
$\mathcal{T} : E(G) \times \Omega \to \mathbb{R}^+$	Specifies the delay time of the edge.
$\Gamma : E(G) \times \Omega \to \mathbb{R}^+$	Specifies priorities regarding execution of the edge.
$\mathcal{P} : V(G) \times \Omega \to \mathbb{R}^+$	A function defining event parameters.
$\mathcal{A} : E(G) \times \Omega \to \mathbb{R}^+$	A function defining edge attributes.

is called event-graph model, well-defined and can be simulated in a unique way.
As this definition is a of rather theoretical nature, it is usually hard to understand especially for students. So similar to most simulation techniques the idea of event-graphs is best introduced studying examples.

2.2 Example: Multiple Server Queue

Figure 1 shows how a part of the browser screen looks like, when opening the MMT-example "Multiple Server Queue". It contains an implementation of a classical event-graph model, well designed for event-graph basic lectures. The main aspect of the model is the simulation of a queue consisting of objects waiting for service. As the model is called multiple server queue, after the queuing process the objects are finally handled at one of



Multiple Server Queue

mean interarrival time	3	mean service time	1
server capacity	1	number of steps	100
seed type	random	seed	12345
arrival and service	stochastic		

Figure 1: MMT Example of a Multiple Server Queue

k servers. There are many ways to interpret this model with real life aspects. It can be thought of ...

- ... data packages waiting for being handled by a k-kernel processor.
- ... a line of cars waiting in front of k identical toll stations.
- ... customers waiting in a shopping mall in front of k cash desks.

In all cases the set of parameters is equal as there is a number of servers k, a rate of incoming objects t_A and a service time t_S, the time one server requires to handle one object.
We now focus on the event-graph illustration in the centre of Figure 1 and explain the ideas of this event-graph. First of all the set of all possible event-types is given by

$$V(G) = \{\text{Arrival}, \text{Start service}, \text{End service}\}$$

as between those events the queue is idle. To guarantee that the event list does not remain empty causing the simulation to immediately break down, the so called "Run" event is added to the list as well. In this simple example the positions, orientations and conditions of the scheduling edges are nearly self explanatory if one thinks about the causal relationships between the event-types.
The lower section of the browser window seen in Figure 1 is reserved for experimenting with input parameters for the simulation. In this case the modelling type can be changed from deterministic (i.e. deterministic arrival and service time) to stochastic using $t_A \sim Exp(\text{mean}(t_A))$ and $t_S \sim Exp(\text{mean}(t_S))$. The exponentially distributed random numbers are gained by a random number generator and can either be calculated by a "random" seed (dependent on the global time), or by a user defined seed.
Pressing the "ok" button, in this case, a MATLAB engine is started in the background calculating the multiple server queue with the chosen input parameters. An output example plot is shown in figure 2. The arrival time (mean 1) and the service time for 8 servers (mean 10) are distributed exponentially. Before going into specific

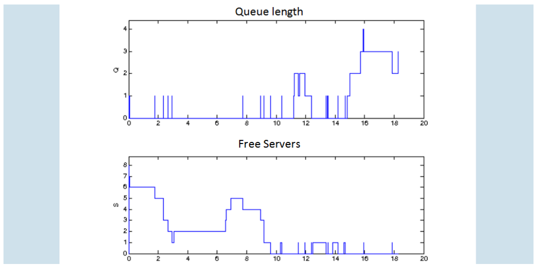


Figure 2: MMT Example Multiple Server Queue - Plot

code details a closer look at the MMT server is taken.

3 MMT Server

The MMT-server was developed and is maintained by dwh-Vienna and is mainly used for lectures at the Vienna University of Technology. Additionally it is used within external courses for modelling and simulation too. The web-interface provides the basis for high quality E-learning. Students as well as lecturers benefit. The most important properties of the server are presented here (see also [2] and [3]).

3.1 Virtual Laboratory

The MMT system provides the opportunity to upload MATLAB, Simulink (Simscape, Simmechanics), Java and AnyLogic models to create a virtual laboratory. The models can be called from the internet-browser, simulation takes place in the background and the results are then sent back to the browser window again in text, picture and/or video form. Specified parameters for the models can be changed directly within the MMT-page by changing the values in provided HTML text-boxes. Thus programming skills are not required for experimenting with high quality modelling and simulation examples.
Currently about 500 MATLAB, 50 Simulink, and 30 Java/Anylogic models can be tested and experimented. Examples written within the free-ware programming languages R and Octave are planned to be included too.

3.2 Upload/Download of Files

As other E-learning systems, also MMT provides the opportunity to offer lecture notes, images or pdf explanations for download. Using an unique node-based course-oriented administration system grants, that each participating student gets access only to those files intended for her/him.

3.3 Improve Programming Skills

As programming skills are becoming more and more important within technical professions the MMT server provides the opportunity downloading each source code, used for the models, and manipulate it at the home-pc. The administrators of the server work hard to guarantee that every code fulfils high quality standards and is fully commented.

4 Inclusion of the Event Graph Examples

As discrete event simulation is only a small subspace of modelling theory, no discrete event software is supported by the MMT system. Thus MATLAB had to be used, unfortunately not providing event-graph packages. In order to solve that problem two completely new event-graph class files were implemented at the Faculty of Electrical Engineering in Ljubljana, which are capable to interpret and simulate a very intuitive implementation of an event-graph model. Some of the basic features are presented here:
An event-graph EG is initialized by the command

$$>> EG = EventGraph;$$

With respect to the formal definition of event-graphs first of all the vertices have to be added by

$$>> addEvent(EG, \text{eventname}, \text{state changes});.$$

All state changes and conditions for the event are defined here too. Finally the edges can be added by

$$>> addSchedulingEdge(EG, \text{edgename}, e_1, e_2, \text{options});.$$

The variables e_1 and e_2 donate the events to be connected by the scheduling edge. Options specifies time delay and conditions for scheduling. After the Defintion of the graph and the specification of options and initial values the simulation can be started by

$$>> EG.state0 = [\text{initial value}\}];$$
$$>> [t, \text{state variable vector}] = simulate(EG, \text{steps});.$$

5 Summary and Outlook

Regarding examples like the multiple server queue, the event-graph library improves the usage of the MMT server, as discrete event simulation is a very important topic within the area of control design modelling. In collaboration with the University of Ljubljana the Vienna University of Technology is currently working on extending the currently implemented and uploaded event-graph examples to finally receive a complete E-learning course for teaching the basis of discrete event simulation. As the current examples only support static picture or text output also animated output is planned.

Acknowledgment

The presented work has been performed within "E-learning in the field of modelling, simulation and control system design" research project, co-funded in the frame of scientific and technical cooperation agreement between the Republic of Slovenia and the Republic of Austria.

References

[1] S. Chick, P. J. Sanchez, D. Ferrin, D. J. Morrice, Paul Hyden, Lee W. Schruben, Theresa M. Roeder, Wai Kin Chan, and Mike Freimer. Advanced event scheduling methodology. In *Proceedings of the 2003 Winter Simulation Conference , Volume: 1.*

[2] I. Hafner, M. Bicher, T Peterseil, S. Winkler, U. Fitsch, Nicole Nagele-Wild, Wolfgang Wild, and F. Breitenecker. Achievements in result visualization with the computer numeric e-learning system mmt. In *Proceedings of the European Modeling and Simulation Symposium, 2012*, Vienna, 2012.

[3] I. Hafner, M. Bicher, S. Winkler, and U. Fitsch. MMT - an e-learning system based on computer numeric system for teaching mathematics and modelling. In *Preprints MATHMOD 2012 Vienna - Full Paper Volume*, Vienna, Austria, 2012.

[4] Enver Yücesan and Lee Schruben. Structural and behavioral equivalence of simulation models. *ACM Trans. Model. Comput. Simul.*, 1992.