A Framework for Hybrid Modeling/Simulation of Discrete Event Systems

Myung Soo Ahn and Tag Gon Kim

Department of Electrical Engineering
Korea Advanced Institute of Science and Technology
373-1 Kusong-Dong,Yusong-Gu,Taejon 305-701, Korea

Abstract

This paper presents a hybrid modeling/simulation framework within which both accuracy in models and speed in simulation experimentations are obtained. Based on the Zeigler's DEVS formalism and associated system theory, the framework is based on the transformation of selected DEVS models into equivalent analytic ones to simulate both analytic and simulation models within a single environment. For high-speed hybrid simulation, we extended DEVSim++ which is a realization of the DEVS formalism in C++. To exemplify the proposed approach, we demonstrate performance modeling and simulation of a simple communication network.

1 Introduction

Recently discrete event modeling and simulation for performance evaluation of complex systems becomes an important research issue in many areas of system design such as manufacturing systems design, communication networks design, and realtime systems design. The main research objective is to devise a framework for developing accurate performance models and efficient simulation algorithms for fast experimentations with such models[3, 5]. Such a framework is essential to reduce the computation burden for simulating large complex systems.

Two types of models, analytic and simulation, have been applied in system analysis and performance evaluation. Although analytic models have limitations in accuracy due to the unrealistic assumptions made, fast simulation experimentations are possible by employing appropriate numerical analysis technique. On the other hand, simulation models have expressive means powerful enough to achieve high accuracy. However, simulation time for such models is extremely slow compared with that for analytic models due to some virtual management scheme embedded in a simulation algorithm. Thus, it is highly desirable to combine the advantages from both models in an unified framework[5].

Up to date, little research has been reported concerning modeling and simulation methodologies which meet both accuracy in modeling and speed in simulation within an unified framework. Furthermore, no simulation language or environment implementing such methodologies is in place.

The purpose of this paper is to develop a framework within which both accuracy in models and speed in simulation experimentations are obtained. To be specific, for modeling we propose a model transformation scheme which transforms selected simulation models into analytic ones as far as accuracy is preserved. For simulation, we develop a hybrid simulation algorithm and the associated environment implementing such algorithm. Thus, we can simulate performance models of a discrete event system, which consists of simulation models and analytic ones, in a single environment.

We employ the Zeigler's DEVS formalism[8], which supports hierarchical modular descriptions of discrete event systems. Though the set-theoretic formalism has expressive power and the well known simulator algorithm, it lacks of analytic means. To complement

[0]ISBN 0-8186-6440-1.

this shortage, the model transformation scheme transforms a DEVS model into a behaviorally equivalent analytic model in steady state. Both DEVS models and transformed analytic models are simulated in a combined manner using the developed hybrid simulation engine.

Our approach is novel in the sense that :

- it employs a single modeling formalism. Thus, it enables the modelers to develop models within the expressive formalism.
- it is based on the sound mathematical foundations in the behavioral equivalence relation between the DEVS models and the transformed analytic models.

The developed framework can be used to measure the performance of a discrete event system with greatly reduced simulation time. To exemplify the proposed hybrid modeling and simulation framework, we demonstrate performance modeling and simulation of a simple communication network.

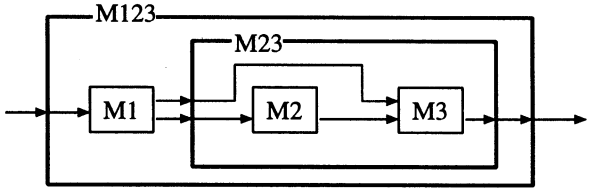
The outline of this paper is as follows. Section 2 presents the concepts of hybrid modeling. Section 3 reviews the DEVS formalism and describes our framework. Section 4 gives an example of application and results. We conclude the paper in section 5.

2 Hybrid Modeling with Hierarchical Discrete Event Models

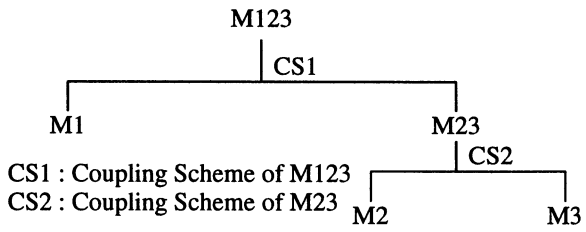
Hierarchical constructions of modular models play an important role in modeling and simulation for design of complex real world systems. In hierarchical composition, higher-level models include low-level models as components. Such higher-level models can be reused as components of yet higher-level models[6].

In such a construction, an atomic model is a basic system component that can function as a self-contained and independent unit. Such a model interacts with other models only through the input and output interface, thereby achieving modularity. Thus, an atomic model may well be characterized by a system that can be defined by an input/output interface and state transitions, which represent the behavior of the model.

Systems may be coupled to build coupled models, which may themselves be employed as components to be coupled with other models to form higher level systems. A coupled model specifies the coupling scheme, that is, input/output connections between component models. A simulation model developed by the hierarchical composition methodology has the structure of a tree, called decomposition tree. Figure 1 shows a hierarchical modeling of a system and associated decomposition tree.



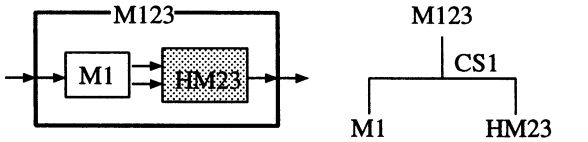
(a) Hierarchical System Model.



(b) Decomposition Tree.

Figure 1: Hierarchical Modular Model.

When such a hierarchical simulation model is given, it is possible to transform some or all of component models into the equivalent analytic models. We call this transforming process as hybrid modeling. The hybrid modeling proceeds in bottom-up fashion. That is, it first transforms the atomic models that meet transformable conditions. After transforming all the possible atomic models, it then aggregates the transformed analytic models using the coupling information. Figure 2 shows an hybrid model which is the result of transformation and aggregation of M12 in Figure 1.



(a) Hybrid Model. (b) Decomposition Tree.

Figure 2: Hybrid Modeling of Hierarchical Model.

Most analytic models require some constraints to make mathematically tractable ones. One of such constraints is the Markov property in continuous Markov chain model. Thus, we need assumptions or simplification methods for transforming atomic simulation models. Two methods can be considered :

- Simplification : state space reduction
  - discretization of continuous state variables;
  - grouping of states at the activity level;
- Generalization : steady state assumption
  - Poisson process assumptions for all event types;
  - Exponential distributions of service times.

But, not all atomic models are transformable. Especially, models representing priority-based processors can not be transformed. Hybrid modeling does not transform such models, thereby preserving accuracy of models. This is the main advantage of hybrid modeling.

The aggregation step at the coupled model level merges two or more transformed analytic models in a single model by using the coupling scheme. Consider two models M1 and M2, shown in Figure 3. M1 acts as a buffer and M2 is a processor. Generally, an atomic model falls in one of these two classes of models : buffer or processor. Using the coupling scheme of two models, we can construct a simple queueing model by merging M1 and M2.

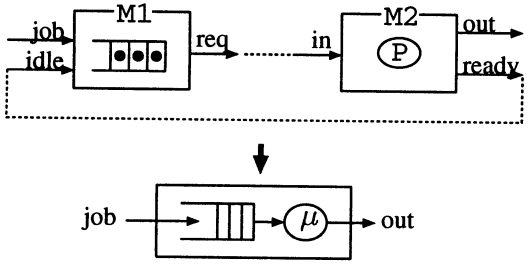


Figure 3: Merging of Two Models into a Single Model.

We repeatedly apply the aggregation process between components models. If all component models are merged into a single model, then the coupled model is transformed in an analytic model. Other coupling information can be useful for merging the models. As an example, a tandem connection of queues can be represented as a single queue[4]. After transforming all the possible models, the hybrid model is simulated in a combined manner.

3 Hybrid Simulation within the DEVSim++ Framework

For hybrid modeling and simulation, we employ the Zeigler's DEVS formalism[8]. As shown in Figure 4, we extend DEVSim++[2] by adding two schemes. The first scheme is a model transformer which transforms

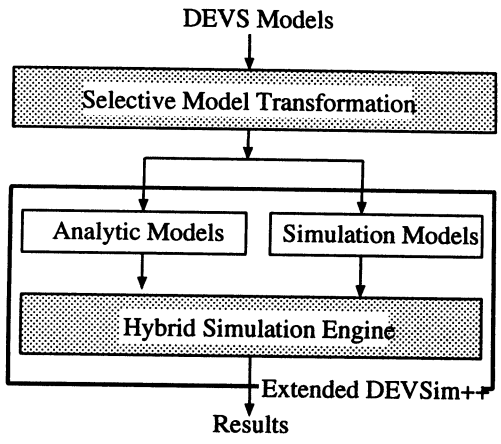


Figure 4: Hybrid Modeling/Simulation with DEVS Models.

selected DEVS models into the equivalent analytic models. The second one is a hybrid simulation engine. This section briefly reviews the DEVS formalism and explains our hybrid simulation environment.

3.1 DEVS Formalism

A set-theoretic formalism, the DEVS formalism specifies discrete event models in a hierarchical, modular form. Within the formalism, one must specify 1) the basic models from which larger ones are built, and 2) how these models are connected together in hierarchical fashion.

A basic model, called an atomic model (or atomic DEVS), has specification for dynamics of the model. An atomic model interprets the behavior of a basic component as a state transition machine. An atomic model AM is specified by a 7-tuple[8]:

$$AM = < X, S, Y, \delta_{int}, \delta_{ext}, \lambda, ta >$$

X : input events set;
S : sequential states set;
Y : output events set;
$\delta_{int} : S \rightarrow S$: internal transition function;
$\delta_{ext} : Q \times X \rightarrow S$: external transition function;
$Q = \{(s, e) \mid s \in S, 0 \leq e \leq ta(s)\}$;
$\lambda : S \rightarrow Y$: output function;
$ta : S \rightarrow Real$: time advance function.

The second form of the model, called a coupled model (or coupled DEVS), defines how to couple (connect) several component models together to form a new model. This latter model can itself be employed

as a component in a larger coupled model, thus giving rise to construction of complex models in hierarchical fashion. A coupled model CM is defined as[8]:

$$CM =< X, Y, M, EIC, EOC, IC, SELECT >$$

X : input events set;
Y : output events set;
M : DEVS components set;
EIC : external input coupling relation;
EOC : external output coupling relation;
IC : internal coupling relation;
$SELECT$: tie-breaking selector.

As proven in [8], the result of coupling DEVS components in a coupled model is itself a atomic DEVS whose state set and input set are cartesian products of all input sets and all total state sets of component models, respectively. Detail descriptions for the definitions of the atomic and coupled DEVS can be found in [8].

3.2 Model Transformation and Simulation Policy

When transforming a simulation model into equivalent analytic one, the transformation should preserve the input/output behavior of the model. For such preservation, we apply the notion of isomorphism or equivalence relation, which is based on one-to-one correspondences between specification structures. If we observe only the input/output behavior of a system, two models are said to be isomorphic or relationally equivalent if input/output behavior of the two cannot be distinguishable in any way[8].

Using the isomorphism, we can represent the steady state behavior of an atomic DEVS model as an equivalent CTMC or as an queueing model. If a model includes a queue for incoming events, it is transformed into a queueing model. Otherwise, an CTMC is used to transform the model.

To set up the isomorphism between a CTMC and the steady state behavior of an atomic DEVS, we make the followings assumptions on atomic DEVS models :

1) the state space of the model is finite;

2) all event types, including internal events, are Poisson process;

3) external and internal transition functions are time-invariant;

4) every state in the state set is reachable from any other states.

Assumption 3 and 4 make the CTMC irreducible and erogodic, thereby solving the CTMC using numerical algorithms. Using the transformed CTMC, steady state probabilities such as mean sojourn time and mean waiting time are obtained. Descriptions on the transformation algorithms can be found in [1].

Analysis of a CTMC with a large state space is very cumbersome and needs vast amount of computation costs. Though a coupled DEVS model can be transformed into an equivalent atomic model, our approach analyzes each atomic DEVS model independently and use the coupling information to merge the transformed CTMCs.

Transformation of DEVS models into queueing models requires some knowledge on the coupling scheme. Actually, two DEVS models, buffer and processor models, are transformed into a queueing network. If we know the rates of incoming jobs of the queueing model, the model acts as a simple delay for the jobs. Otherwise, the rate of the jobs is estimated during the simulation experiments.

Eventually, the transformed analytic models can be represented using the Input/Output Relation Observation(IORO) specification. An IORO observes the behavior of a system in term of input/output relation. It is a structure[8] :

$$IORO =< T, X, \Omega, Y, R >$$

T : time base set;
X : input events set;
Y : output events set;
$\Omega \subset (X, T)$: input segment set;
$R \subset \Omega \times (Y, T)$: I/O relation.

Since the simulation environment manages the simulation time (clock), we just need to describe the three components(X, Y, R) in the structure. As an example of specification, consider a queueing model. It can be represented as :

$$QUEUE_{IORO} =< X, Y, R >$$

X = {job};
Y = {out};
R : (job,t) $\rightarrow$ (out, t + μ + W),

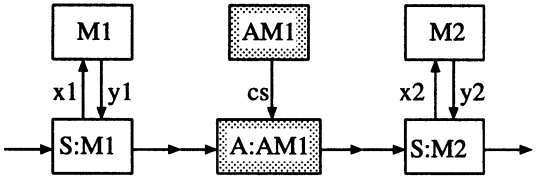
where μ, W are average service time and waiting time in the queue, respectively. In the performance evalu-

ation view point, simulation experiments are the processes of finding the I/O relations of models such as μ, W in the above model.

For hybrid simulation, we combines the abstract simulator algorithms of the DEVS formalism and the analyzers for analytic models. The role of analyzer is to find the average occurrence rates of incoming events and to route the events to the simulators which are responsible for the influencees. To find the influencees, the input and output relations of the associated analytic model are used.

M1, M2 : simulation models;
AM1 : analytic model;

(a) Hybrid Model.

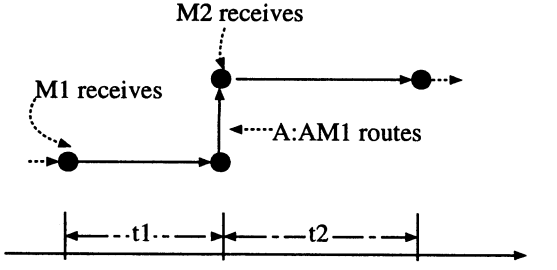


S:M1, S:M2 : simulator for M1 and M2;
A:AM1 : analyzer for AM1.

(b) Association of Models and Simulators.

Figure 5: Hybrid Simulation Strategy.

Figure 5(b) shows the connections between simulators and an analyzer for simulating the model in Figure 5(a). Whenever an event arrives at the analyzer, it counts the arrival to determine the average rate of incoming event. Figure 6 shows the scenario of event flow when an event enters the models M1 of Figure 5.



t1 : processing time in M1;
t2 : processing time in M2;
Overall Performance = t1 + t2 + E[W]

Figure 6: Event Flow during Hybrid Simulation.

After a simulation experiment is finished, the analytic model determines the I/O relation in the steady state using the rates. The term E[W] in the performance of Figure 6 is the results from AM1 in Figure 5. When a model knows the occurrence rates of events, the associated analyzer does nothing during the simulation except the routing of events.

3.3 Extended DEVSim++ Environment

To simulate the transformed analytic models with simulation models, we extended the DEVSim++ environment. The original version of DEVSim++[2] realizes the DEVS formalism for modeling and associated abstract simulator concepts for simulation, all in C++. DEVSim++ is a result of the combination of two powerful frameworks for system development : the DEVS formalism and the object-oriented paradigm.

Since DEVSim++ defines classes for modeling and those for simulation separately, it can be easily evolved by developing new classes. Figure 7 shows class hierarchy of the extended environment. The shaded two classes are newly defined for hybrid simulation.

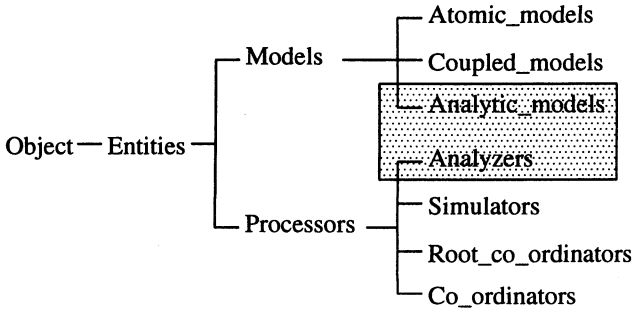


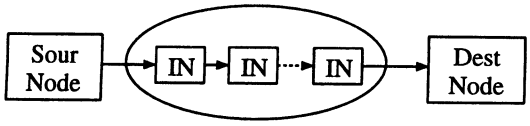
Figure 7: Class Hierarchy of Extended DEVSim++.

The Analytic_models class realizes the transformed analytic models. It has instance variables corresponding to three elements in IORO representation : X for input events set, Y for output events set, and R for input/output relations. To manage the I/O relations, we define a structure for input/output relations. *Analytic_models* also defines methods operating on instance variables. *Analyzers* is assinged to *Analytic_models* in a one-to-one manner.

4 Example and Results

As an application of our approach, we consider a simple communication network. As shown in Figure

8, the system consists of a sending node and a receiving node and a set of intermediate routing nodes. The sending node transmits a number of packets to the destination node through the intermediate routing nodes.



IN : Intermediate Routing Node

Figure 8: Simple Communication Network.

For comparison of computation time and accuracy of our approach with discrete event simulation, we modeled the system within the DEVSim++ environment. By applying the model transformation method, we can transform the intermediate routing nodes into a single queueing network.

We compare the computation times by measuring the simulation times of both models under the condition that two models generate the same number of packets. All experiments are performed in a Sun Sparc 1+ machine with 32MB main memory.

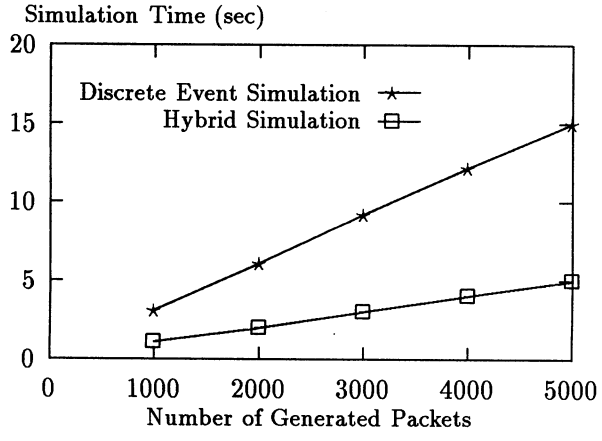


Figure 9: Simulation Time for Varying Number of Generated Packets.

Figure 9 shows the average simulation times when the number of generated packets are varying and systems are configured with two intermediate routing nodes. Each point takes an average of 5 statistically independent simulation runs. The results show that the hybrid approach greatly reduces the computation time.

To compare accuracy between both approaches, we measure the average traveling times of packets, which

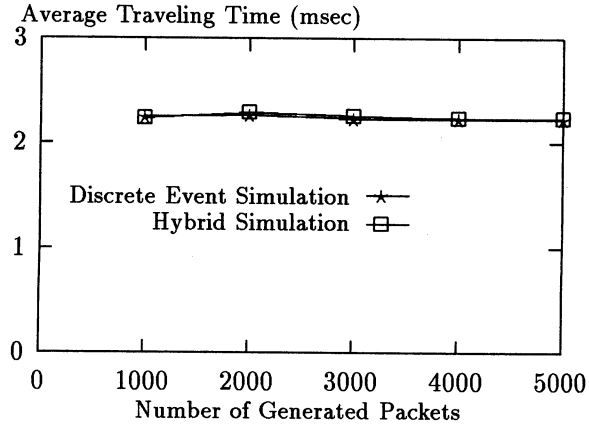


Figure 10: Average Traveling Time of Packets.

are the times between the departure at the source node and the arrival to the destination. From the results of Figure 10, we can conclude that there is little difference between two approaches. Though the simulation model is simple, the results shows practical significance of our approach.

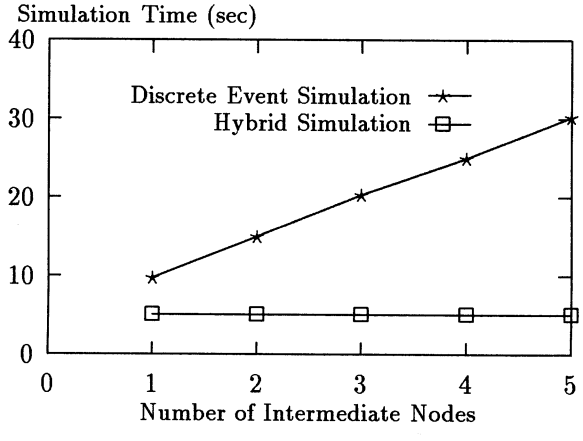


Figure 11: Simulation Time for Varying Number of Intermediate Nodes.

Figure 11 compares the computation costs when the number of intermediate routing nodes are varying. Since the hybrid approach models the tandem-connected nodes as a single queueing model, the computation costs do not increase no longer. But, those for the discrete event simulation continuously increase. From the above results, we can draw a conclusion that our approach has a practical significance, especially when simulating complex systems.

5 Summary and Conclusions

We have proposed a hybrid modeling and simulation framework for high-speed simulation without losing the accuracy of discrete event simulation. The approach is based on a transformation of the steady state behavior of a DEVS model into an equivalent analytic model. For hybrid simulation of the transformed analytic models and DEVS models, we extended the DEVSim++ environment by adding new classes for specifying and simulating analytic models.

Also, we validated the proposed approach by comparing the results with those obtained from simulation experiments with only discrete event simulation models. The results show that our approach can accurately simulate the behavior of a system with greatly reduced simulation time.

Though the approach shows some promising results, there is much work to be done. Currently, an extension of the model transformation method to more general and complex cases is underway. In parallel, we are also extending the DEVSim++ environment to possible automatic model transformation.

Acknowledgements

This work was supported by the Korea Science and Engineering Foundation grant 941-0900-034-2.

References

[1] Myung S. Ahn and Tag G. Kim, "Analysis on Steady State Behavior of DEVS Models", *Proc. of 4th Annual Conf. on AI, Simulation, and Planning in High Autonomy Systems(AIS '93)*, pp. 142 - 147, Sept. 1993.

[2] Myung S. Ahn and Tag G. Kim, *DEVSim++ User's Manual*, Technical Report, TR-CORE-94-1, EE, KAIST, 1994.

[3] Kumar K. Goswami and Ravishankar K. Iyer, "Use of Hybrid and Hierarchical Simulation to Reduce Computation Costs", *Proc. Int. Workshop on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS '93)*, pp. 197-202, Jan. 1993.

[4] L. Kleinrock, *Queueing Systems, Volume II: Computer Applications*, Wiley, New York, 1976.

[5] J.G. Shanthikumar and R.G. Sargent, "A Unifying View of Hybrid Simulation/Analytic Models and Modeling", *Operations Research*, Vol. 31, No. 6, Nov. 1983.

[6] Tag G. Kim and Myung S. Ahn, "Reusable Simulation Models in an Object-Oriented Framework", to appear in *Object-oriented Simulation* (ed: G.W. Zobrist), IEEE Press.

[7] B.P. Zeigler, *Theory of Modelling and Simulation*, John Wiley, NY, 1976(Reissued by Krieger Pub. Co., Malabar, FL. 1985).

[8] B.P. Zeigler, *Multifacetted Modeling and Discrete Event Simulation:* Academic Press, Orlando, FL., 1984.