Analysis on Steady State Behavior of DEVS Models

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

Zeigler's DEVS formalism supports an unified modeling and simulation framework for discrete event systems, but it lacks of an analytic means for reasoning about system behavior. To provide both analytic and simulation means for the formalism, this paper proposes an approach to analyze the steady state behavior of DEVS models without simulation experiments. By establish correspondence between a DEVS model and a continuous time Markov chain in steady state, the approach transforms the given DEVS model into an equivalent continuous time Markov chain. By analyzing the Markov chain, various steady state probabilities, such as mean sojourn time and mean waiting time, are obtained. We validate the proposed approach by comparing the results from the approach with those from simulation experiments.

1 Introduction

Two types of models, analytic and simulation, have been applied in system analysis and performance evaluation. Analytic models are cost effective when the model of a system is tractable with efficient numerical algorithms. To make an analytic model numerically tractable, a certain degree of simplification of system's details is inevitable. However, this simplification limits the modeling power of an analytic model. Though any level of details can be specified in a simulation model, the computation cost for simulating the system behavior grows rapidly as the model becomes more complex[3].

In recent, many research efforts have focused on the hybrid simulation/analytic models to combine both models into a cost efficient model[6]. To combine the advantages from both models in an unified framework, it is highly desirable to have both analytic and simulation means within a specification formalism. However, little results has been presented in the literature in our knowledge.

Zeigler's DEVS formalism supports specification of discrete event models in a hierarchical, modular manner[8]. It also provides a powerful modeling and simulation framework by giving the abstract simulator concepts. Thus, it can be widely applicable in many areas of system design such as manufacturing systems design, communication networks design, and realtime systems design[1]. While providing the powerful expressive power for simulation modeling, the formalism lacks of an analytic means for the behavior reasoning of the DEVS models.

This paper presents an approach to analyze the steady state behavior of DEVS models without time-consuming simulation experiments. The approach is based on the idea of behavioral equivalence between a DEVS model and a continuous time Markov chain(CTMC) in steady state. The approach proceeds in two steps. First, it transforms each component DEVS model into an equivalent CTMC and then analyzes the CTMC's using coupling information of the DEVS model. Using the CTMC's various steady state probabilities are obtained.

The outline of this paper is as follows. Section 2 presents a brief review of the DEVS formalism. Section 3 describes behavioral characteristics of DEVS models in steady state and identifies the behavioral equivalence between a steady state DEVS model and a CTMC. This section also provides details on the transformation procedure. In section 4, an example of application and results are given. We conclude this paper in section 5.

2 The DEVS Formalism

A set-theoretic formalism, the DEVS formalism specifies discrete event models in a hierarchical, modular form. Within the formalism, two classes of models,

0-8186-4020-0/93 $03.00 © 1993 IEEE

namely atomic and coupled models, are to be specified. An atomic DEVS model is one that can not be decomposed into components while a coupled one can be decomposed in to component models.

A basic model, called an atomic model (or atomic DEVS), has specifications for the dynamics of the model. An atomic model M is specified as:

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

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.

An atomic model interprets the behavior of a system as a state transition machine. The set S includes all possible states of the system and $ta(s)$ maps s to a non-negative real with infinity, which is the time during which the system is allowed to stay in state s if no external event occurs. Atomic DEVS model employs two types of state transitions, internal transition and external transition. The internal transition function δ_{int} defines the next state when no external event occurs during current state. If an external event occurs in the middle of a state, the next state of system is determined by the external transition function δ_{ext}.

The second form of the model, called a coupled model (or coupled DEVS), tells how to couple (connect) several component models together to form a new model. This latter model can be employed as a component in a larger coupled model, thus giving rise to the construction of complex models in a hierarchical fashion. A coupled model DN is defined as:

$$DN =< D, \{M_i\}, \{I_i\}, \{Z_{i,j}\}, SELECT >$$

D : component names set;
for each i in D,
  M_i : DEVS for component i in D;
  I_i : set of influencees of i;
for each j in I_i,
  $Z_{i,j} : Y_i \rightarrow X_j$;
$SELECT$: subsets of $D \rightarrow D$.

The activities of a component model affect the state of the influencees by the output translation function $Z_{i,j}$. 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, 9].

3 Steady State Behavior of DEVS Models

To analyze the steady state behavior of a DEVS model without time-consuming simulation experiments, we need to transform the DEVS model into a behaviorally equivalent analytic model. In the steady state transition view point of a system, two models are said to have a behavioral equivalence if (1) there exists a one-to-one correspondence between each state, and (2) transition probabilities between states of a model are preserved in the other model, and (3) two models have the same sojourn time distribution for each state. Thus, two equivalent models represent the same behavior with different specifications.

By observing the steady state behavior of a DEVS model, we can represent it as an equivalent CTMC. Since the state transitions in a DEVS model are caused by the external and internal events, event occurrence rates correspond to the transition probabilities of CTMC. To set up the behavioral equivalence between a CTMC and a steady state DEVS model, 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 of S is reachable from any other states.

To satisfy assumption 1, we may employ some simplification procedures. One of possible simplification procedure is to drop one or more descriptive, continuous state variables[7]. We will introduce examples of simplified models in the next section. Assumption 2 requires that all state variables and events be random variables and random process, respectively. Moreover, it requires that all events have exponentially distributed interarrival rates to ensure the Markov property of the model. Assumption 3 and 4 make the CTMC irreducible and erogodic, thereby solving the CTMC using numerical algorithms.

The transformation process proceeds in two steps. In the first step, we extract states and transition probability matrix for each atomic model with which a local CTMC is constructed. After constructing the local CTMC's, the external events rates for the component models of a coupled model are fixed using coupling information.

3.1 Atomic Model Transformation

We can represent a CTMC using a structure $< S, Q >$, where S is the state set and Q is the transition rate matrix. Let $< X, S, Y, \delta_{int}, \delta_{ext}, \lambda, ta >$ be an atomic DEVS model to be transformed.

The state space of an equivalent CTMC is the same as that of the DEVS model and the transition matrix Q can be partitioned into two matrices :

$$Q = U + V.$$

The elements of matrix V are transition rates caused by the internal events. If a pair of states (s_i, s_j) satisfy the internal transition function such that

$$\delta_{int}(s_i) \rightarrow s_j,$$

we set

$$u_{ij} = 1/ta'(s_i).$$

$ta'(s_i)$ is the randomized sojourn time distribution from the original time advance function.

An element v_{ij} of the matrix V corresponds to the transition rate from state i to state j, which is caused by the external events. Thus,

$$v_{ij} = \sum_{x \in E_{ij}} \psi_x,$$

where E_{ij} is the set of all external events which cause the transition from state i to state j and ψ_x is the occurrence rate of event x. An event x is included in E_{ij} if there exists a pair of state satisfying the external transition function, that is,

$$\delta_{ext}(s_i, e, x) \rightarrow s_j.$$

We associate a variable with each external event in this step.

Figure 1 summarizes the above procedure for the transformation of an atomic DEVS model into an equivalent CTMC. In this step, the occurrence rates of external events are unknown, so we associated a variable with each external event. We resolve these unknown variables in the next step using the coupling information between atomic models. Solving the CTMC constructed, the steady state probability distribution and mean recurrence time for each state are obtained.

---

input : $< X, S, Y, \delta_{int}, \delta_{ext}, \lambda, ta >$;
output : a CTMC $< S', Q >$;
1) initialize S' to S;
2) initialize $q_{ij}, 1 \leq i, j \leq |S|$, to 0;
3) for each external event and (s_i, s_j) pair
  such that $\delta_{ext}(s_i, e, x) \rightarrow s_j$
    $q_{ij} = q_{ij} + e_x$;
  endfor
4) for each internal event and (s_i, s_j) pair
  such that $\delta_{int}(s_i) \rightarrow s_j$
    $q_{ij} = q_{ij} + 1/ta'(s_i)$;
  endfor
5) for each state
    $q_{ii} = -\sum_{j, j \neq i} q_{ij}$;
  endfor

Figure 1: Atomic model transformation procedure.

3.2 Analysis of Coupled DEVS Models

As mentioned in the previous section, the coupled atomic DEVS models can be transformed into an equivalent atomic model. Thus, we can construct a CTMC corresponding to a coupled model. The space space of the CTMC consists of the cartesian product of the states of all the CTMC's of component models. The state space of resulting CTMC will more rapidly grow than the system complexity. To analyze a CTMC with a large state space is very cumbersome and needs vast amount of computation costs. In our approach, we analyze each atomic DEVS model independently and use the coupling information to relate event occurrence rates.

The couplings between component models in a coupled model can be classified into three classes :

1) request/acknowledge(queued) couplings;

2) synchronization couplings;

3) notifying couplings.

A pair of couplings between models i and j is an queued coupling if the sequence of operations is as follows. Model i invokes a service from model j through coupling "req" and it waits for the acknowledge signal. Upon receiving the request, model j queues the requested job. Model j sends an acknowledge over the coupling "ack" when the component is ready for the requested job. Figure 2 shows this class of couplings and state transitions.

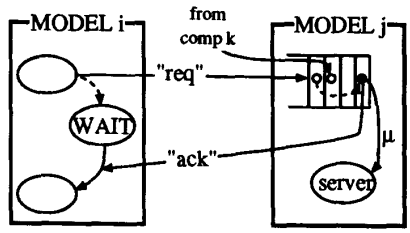

Figure 2: State transition by queued coupling.

The average sojourn time of $WAIT$ becomes the expected waiting time in the queue and can be estimated using a queueing model.

The second class of couplings are used to synchronize states between two component models. Figure 3 shows the state transitions caused by a pair of synchronization signals.

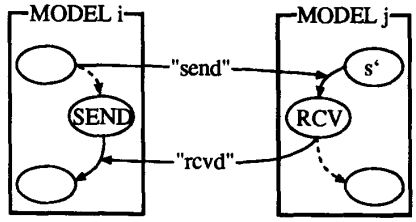

Figure 3: Synchronization of two models using signal.

The sojourn time of state $SEND$ is same as that of state RCV. Once the sojourn time of RCV determined, we set $ta(RCV)$ as $SEND$'s sojourn time and we can disconnect the "rcvd" signal coupling. But, the "send" signal will affect the transition rate of s'.

A component model may send the notification of an event to influencees and does not wait for a return signal. We call this class of couplings notifying coupling. In this case, the event occurrence rate is the mean recurrence time of the state at which the source component sends a signal.

Using the above classification and event occurrence rates, the couplings between models are simplified to the notifying coupling. The rates of notifying signals can be determined if steady state probabilities for each CTMC are determined. Next section describes an example of application.

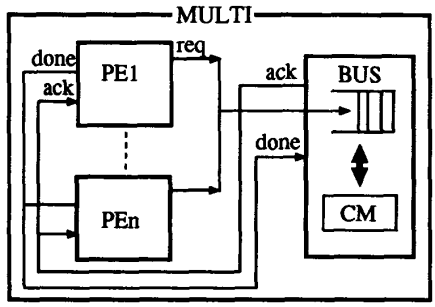

Figure 4: Single bus multiprocessor system.

4 Example and Results

As an application of our approach, we consider the single-bus multiprocessor system with a shared common memory. The system consists of p processing elements with private memory and a single bus through which the elements access the common shared memory. The architecture is shown in figure 4. When a set of processing elements needs to access common memory through the single bus, a bus contention occurs. Only one element can connect the memory at any instant. Since such bus contentions degrade the performance of system, the effect of contentions on the performance has become the major concerns of researchers. A number of research efforts on the performance study of the architecture have been reported in the literature. We will compare the result of our approach with one of them.

To model the system within the DEVS formalism, we first develop two atomic models as the components of the systems. The atomic model PE describes the behavior of a processing element. We assume that all processing elements are identical. The atomic BUS model specifies the single bus and the common memory. Once the atomic model are developed, we then construct a coupled DEVS model MULTI. The coupled model MULTI specifies the coupling between the component atomic models. The coupling scheme is shown in figure 4.

Figure 5 shows the phase transitions of each atomic models. Since the state variable PHASE directly captures the state transitions of the models, we simplify the state sets of each atomic model as possible range of variable PHASE. Thus, the state set of PE becomes $\{phase_1 = LOCAL, phase_2 = WAIT, phase_3 = ACTIVE\}$ and $\{phase_1 = IDLE, phase_2 = SACK, phase_3 = TRANS\}$ for BUS.

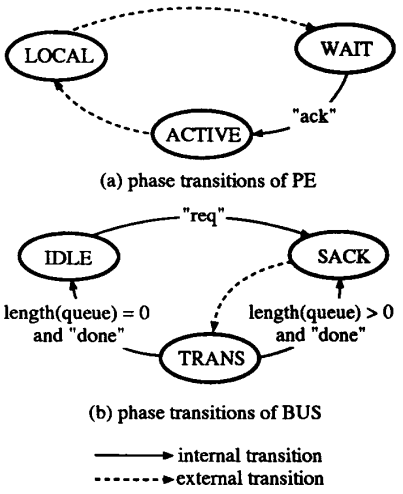

Figure 5: State transition of components.

Each processing element operates at one of three states. Initially, an element executes a job at LOCAL state which do not require the common memory access. After executing the job during exponentially distributed random time, the element requests bus to access common memory and then passivates in the WAIT state to wait for an acknowledge signal. Upon receiving an acknowledge signal, the element accesses the common memory. BUS model serves the bus requests from processing elements in the first come first served mode. The following is the atomic DEVS specification of PE :

$X = \{ack\};$
$Y = \{req, done\};$
$S = \{WAIT, LOCAL, ACTIVE\};$

$\delta_{ext}(WAIT, e, "ack") \rightarrow ACTIVE;$

$\delta_{int}(LOCAL) \rightarrow WAIT;$
$\delta_{int}(ACTIVE) \rightarrow LOCAL;$

$\lambda(LOCAL) \rightarrow "done";$

$ta(LOCAL) \rightarrow exponential(\lambda_l);$
$ta(ACTIVE) \rightarrow exponential(\mu);$
$ta(WAIT) \rightarrow \infty;$

Applying the atomic model translation procedure, we can obtain the transition rate matrices for the atomic models PE(Q_{PE}) and BUS(Q_{BUS}) :

$$Q_{PE} = \begin{vmatrix} \lambda_l & -\lambda_l & 0 \\ 0 & -\psi_{ack} & \psi_{ack} \\ \mu & 0 & \mu \end{vmatrix}$$

$$Q_{BUS} = \begin{vmatrix} -\psi_{req} & -\psi_{req} & 0 \\ 0 & -\lambda_a & \lambda_a \\ \eta\psi_{done} & (1-\eta)\psi_{done} & -\mu \end{vmatrix}.$$

η in Q_{BUS} is the probability that the queue is empty.

From the couplings knowledge of figure 4, we can find that the pair of coupling "ack" and "done" synchronizes $ACTIVE$ state of a PE and $TRANS$ state of BUS. Thus, the sojourn time of $TRANS$ becomes that of $ACTIVE$. Also, the pair of coupling "req" and "ack" constructs a queued coupling which can be models as an $M/M/1$ queue with a finite number of customers, namely processing elements. Using the well known solution methods for such queues[4], we can easily find the average waiting time and queue length.

The probability for finding no request in the queue, η, is given by [4]

$$\frac{1}{\sum_{i=0}^{n}(n!(n-i)!)(\lambda_l/\mu)^i},$$

where n is the number of processing elements. And the average waiting time in the queue, $E[W]$, is given by [4]

$$\frac{n/\mu}{1-\eta} - \frac{1}{\lambda_l} - \frac{1}{\mu}.$$

The inverse of this waiting time is the rate of the "ack" signal to a processing element. By solving all unknowns in Q_{PE} and Q_{BUS}, transition rates and sojourn times of states are obtained. These probabilities enable us to estimate system behavior in steady state.

In order to validate our approach, we performed simulation experiment with the developed DEVS model in the DEVSIM++ environment[1]. We compared the processing power P which is one of major performance indices of bus-connected multiprocessors and is defined as the average number of processors in $LOCAL$ state. Each simulation result takes an average of 10 statistically independent simulation runs and summarized in tables 1 and 2. The notations used in the tables are as follows :

n : number of processing elements;

P_a : results from our approach;

n	P_s	P_a	% error
2	1.62	1.62	0
5	3.59	3.58	-0.3
10	4.98	4.91	-1.4
15	5.07	5.00	-1.4
20	5.09	5.00	-1.8

Table 1: Processing power for varying number of processing element.

λ_i/μ	P_s	P_a	% error
0.1	4.36	4.36	0
0.2	3.59	3.58	-0.3
0.3	2.89	2.87	-0.7
0.4	2.35	2.33	-0.9
0.5	1.94	1.93	-0.5

Table 2: Processing power for varying load.

P_s : results from simulation experiments;

% error : $(P_a - P_s)/P_s \cdot 100$.

Table 1 shows the processing power for varying the number of processors. The offered load, λ_i/μ, is set to 0.2. The results show that our approach can accurately estimate the steady state behavior of DEVS models. Comparisons for different offered load are shown in table 2. The number of processing elements is set to 5.

5 Summary and Conclusions

We have presented an approach to analyze the steady state behavior of DEVS models without simulation experiments. The approach is based on a transformation of a DEVS model into an equivalent CTMC in steady state. Also, we validated the proposed approach by comparing the results with those obtained from simulation experiments. The results show that our approach can accurately estimate the steady state behavior of DEVS models.

Our approach can be applied in many areas. It can be used to figure out the operation of a system in the early stage of system design with inexpensive computation cost. When used in a hybrid modeling and simulation, it can replace a complex state transition DEVS model with a simple probabilistic model. This will greatly reduces the simulation time.

Currently, an extension of the proposed approach to more general and complex cases is underway. In parallel, we are plan to develop a hybrid modeling and simulation framework within which both analytic and simulation models are simulated. We believe that the investigation of sound mathematical foundations in the coupled model analysis is a good topic for further research.

References

[1] Myung S. Ahn and Tag G. Kim, "DEVS methodology for Evaluating Time-Constrained Message Routing Policies", *Discrete Event Dynamic Systems: Theory and Applications*, 3, pp 173-192, 1993.

[2] J.R. Clymer, *Systems Analysis Using Simulation and Markov Models*, Prentice-Hall, Englewood Cliffs, NJ, 1990.

[3] P. Heidelberger and S.S. Lavenberg, "Computer Performance Evaluation Methodology", *IEEE Trans. on Computers*, Vol. 32, No. 12, Dec. 1984.

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

[5] M. Ajmone Marsan, G. Balbo, and G. Conte, *Performance Models of Multiprocessor Systems:* MIT Press, Cambridge, MA, 1986.

[6] 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.

[7] B.P. Zeigler, *Theory of Modelling and Simulation:* John Wiley, New York, 1976(Reprinted by Krieger Pub. Co., Malabar, Florida, 1985).

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

[9] B.P. Zeigler, *Object-Oriented Simulation With Hierarchical, Modular Models : Intelligent Agents and Endomorphic Systems* Academic Press, Inc , 1990.