Efficient Simulation of Component-Based Hybrid Models Represented as Hybrid Bond Graphs

Matthew Daigle, Indranil Roychoudhury, Gautam Biswas, and Xenofon Koutsoukos

EECS Department/ISIS, Vanderbilt University
Nashville, TN 37235, USA
matthew.j.daigle@vanderbilt.edu

1 Introduction

Modern engineering systems consist of a large number of interacting components with nonlinear, hybrid behaviors. This makes the building of accurate and computationally efficient simulation models very challenging. Recently, researchers have adopted component- [1] and actor-oriented [2] frameworks for modeling of large systems. Such frameworks consist of mathematical models for specifying individual component behavior and formal models of computation for defining component interactions and are used for simulating hybrid and embedded systems

In our work, we adopt the Hybrid Bond Graph (HBG) paradigm [3], an extension of the Bond Graph (BG) modeling language [4], for component-based modeling of embedded systems. HBGs is a domain-independent topological modeling language that captures interactions among the different processes. This topological information is very effective in parameterized component-based modeling of hybrid systems and offers significant advantages for model-based fault diagnosis [5].

The challenge we address in this paper is translating HBG models to computationally efficient simulation models. The causal structure inherent in BG models provides the basis for conversion of BGs to efficient computational models. For HBGs, discrete events cause dynamic changes in the causal structure which affects the computational model during execution. This paper presents a method for efficient simulation of HBG models by converting them to reconfigurable block diagram models using a *hybrid sequential causal assignment procedure*. We demonstrate the technique by generating a computational model of an electrical power system in MATLAB ® Simulink ® [7].

2 Translating Hybrid Bond Graphs to Block Diagrams

BGs are domain-independent, topological, lumped-parameter models that capture the energy exchange mechanisms in physical processes [4]. The nodes of a bond graph model energy storage energy dissipation energy transformation and input-output elements Connections in the system are modeled by two idealized

elements: 0- (or parallel) and 1- (or series) junctions. The connecting edges, called *bonds*, are energy pathways between elements. Parameters of nonlinear BG elements are expressed as algebraic functions of other system variables and external input signals, called *modulating functions*. HBGs extend continuous BG models by allowing discrete changes in configuration by turning junctions on and off [3]. A finite state machine implements the junction *control specification*. Each state maps to an *on* or *off* state of the junction, and the transition guards are expressed as boolean functions of system variables and inputs. When a controlled junction is on, it behaves like a conventional junction. In the off state, all bonds incident on a 0- (1-) junction are deactivated. The system mode at any time is determined by composing states of the individual switched junctions. Details of the modeling language are presented in [8].

There exist two primary challenges in generating simulation models from HBG representations. First, we should avoid pre-enumeration of model configurations. A HBG model with m components, each with n_i switching junctions, where $i = 1, 2, \ldots m$, defines $2^{\sum_{i=1}^{m} n_i}$ different system modes (or model configurations). When $\sum_{i=1}^{m} n_i$ is large, it is infeasible to pre-enumerate all the model configurations before running the simulation. Therefore, mode changes, and reconfiguration of the BD model, must be performed during run-time. Second, we should avoid algebraic loops. In component-based modeling, the underlying mathematical model is usually a set of differential-algebraic equations (DAEs). The DAE models may include algebraic loops. However, generating a fixed-point solution may become computationally expensive if the fixed-point method needs many iterations to converge to a solution.

Causality Assignment BG models imply a causal structure, and algorithms like the *Sequential Causal Assignment Procedure* (`SCAP`) [4], applied to well-formed BG models, assign causal directions to all bonds in the model. The causal direction of a bond determines the functional relation between the associated effort and flow variables, and, therefore, define the input-output relations of these variables for each BG element. The input-output relations can be represented as block diagrams (BDs), a widely used graphical, computational scheme. Using causality, the derived BD model will have a minimized number of algebraic loops [4].

Given causal assignments to the bonds, there is a direct mapping between a BG and its BD expansion. For HBGs, however, causal assignments may change when junctions switch state. To avoid the costly pre-enumeration of system modes, we implement an efficient BD reconfiguration scheme that recomputes the causal assignments incrementally, starting from junctions that switch state, and propagating causal assignment changes till a new consistent assignment is derived. Corresponding changes are made only to those blocks in the BD structure that have changes in the causal assignments of their incident bonds.

The *Hybrid Sequential Causal Assignment Procedure* (`Hybrid SCAP`) dynamic reassigns causality to HBGs. We assume that the new states of all junctions are available before `Hybrid SCAP` is applied. The algorithm starts with a queue of

switched junctions. `Hybrid SCAP` picks one junction off the queue, makes all forced causal assignments, and propagates effects of these assignments, making all forced changes and continuing until no further causal assignments are forced. Junctions with incomplete causal assignments to their bonds are added to another queue. Once the first queue is emptied, the algorithm picks elements off the second queue, makes an unforced choice to complete the causal assignments, and propagates its effects to make any forced changes that result from the chosen assignment. This process continues until all bonds with unassigned causality have been assigned causality. Because the propagation is local, only a subset of the bonds needs to change causal assignments. Details can be found in [6].

Implementation In MATLAB Simulink, we have explored two different implementation approaches: *implicit switching* and *explicit switching*. In implicit switching, we use conditional statements to model the variable input-output relations for a block element whose incident bond(s) can change causality. The switching of the data flow between blocks is, therefore, implicit in the model. Multiple modes can be represented concisely in code form, so are not explicitly enumerated, resulting in compact models. However, this approach produces algebraic loops in the Simulink model, because the input-output directional structure is buried the code. For simulating these models, Simulink invokes a fixed point solver. Though all of these structures ensure a unique solution, the solver incurs a computational overhead which affects the simulation efficiency.

In explicit switching, we use switching elements to enumerate the possible data flow paths and computational structure for each configuration. At run time, the appropriate switches are triggered to activate the new effective block diagram structure. Compared to the previous method, the models created by this approach have many more atomic blocks because multiple BD expansions are enumerated for each element. Since the data flow paths are made explicit for each configuration, no additional algebraic loops are created, however, the switching elements incur an overhead associated with zero-crossing detection.

3 Case Study: Electrical Power System

We apply our modeling and simulation framework to the Advanced Diagnostics and Prognostics Testbed (ADAPT) system deployed at NASA Ames. The system consists of three subsystems: *power generation*, consisting of a solar panel and battery chargers, *power storage*, consisting of three sets of lead-acid batteries, and *power distribution*, consisting of a number of AC and DC loads and AC to DC converters. Relays configure the system in different modes of operation, e.g., charge and/or discharge modes of the batteries, as well as different power supply and load configurations. Because of the large number of components and possible configurations, it is infeasible to pre-enumerate all possible modes of operation. We have developed HBG models of these components using our approach that allows adding and removing components as well automatically generating the simulation model [6].

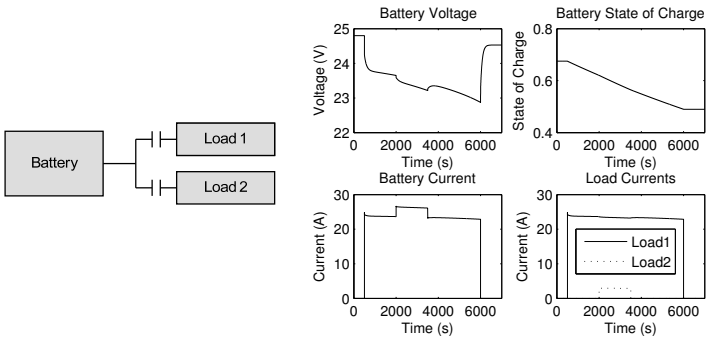

Fig. 1. Example model and simulation results

We present the simulation results for a battery supplying power to two DC loads in parallel, with relays that enable the loads to be switched on- or offline (see Fig. 1). The Simulink model was run for $7,000$ seconds of simulation time with the battery discharging through different load conditions. For this configuration, the explicit switching implementation executed about 20% faster than the implicit switching implementation. This difference in the simulation efficiency is consistent for a number of other configurations of the system.

References

1. Liu, J., Lee, E.A.: A component-based approach to modeling and simulating mixed-signal and hybrid systems. ACM Trans. Model. Comput. Simul. **12**(4) (2002) 343–368
2. Lee, E.A., Neuendorffer, S., Wirthlin, M.J.: Actor-oriented design of embedded hardware and software systems. Journal of Circuits, Systems, and Computers **12**(3) (2003) 231–260
3. Mosterman, P.J., Biswas, G.: A theory of discontinuities in physical system models. J Franklin Institute **335B**(3) (1998) 401–439
4. Karnopp, D.C., Margolis, D.L., Rosenberg, R.C.: Systems Dynamics: Modeling and Simulation of Mechatronic Systems. Third edn. John Wiley & Sons, Inc., New York (2000)
5. Mosterman, P.J., Biswas, G.: Diagnosis of continuous valued systems in transient operating regions, IEEE Transactions on Systems, Man and Cybernetics, Part A, **29**(6), (1999) 554–565.
6. Daigle, M., Roychoudhury, I., Biswas, G., Koutsoukos, X.: Efficient simulation of component-based hybrid models represented as hybrid bond graphs. Technical Report XXXX, Institute for Software Integrated Systems, Vanderbilt University, Nashville, TN, USA (2006).
7. MATLAB/Simulink: (http://www.mathworks.com/products/simulink/)
8. Manders, E.J., Biswas, G., Mahadevan, N., Karsai, G.: Component-oriented modeling of hybrid dynamic systems using the Generic Modeling Environment. In: Proc of the 4th Workshop on Model-Based Development of Computer Based Systems, Potsdam, Germany, (2006).