An Efficient Method for Simulating Complex Systems with Switching Behaviors Using Hybrid Bond Graphs

Indranil Roychoudhury*, Matthew J. Daigle†, Gautam Biswas*, and Xenofon Koutsoukos*

* Institute for Software Integrated Systems, Dept. of EECS, Vanderbilt University, Nashville, TN 37235, USA.
Email: {indranil.roychoudhury,gautam.biswas,xenofon.koutsoukos}@vanderbilt.edu
† University of California, Santa Cruz, NASA Ames Research Center, Moffett Field, CA 94040, USA.
Email: matthew.j.daigle@nasa.gov

Abstract

Accurate and efficient modeling and simulation approaches are essential for design, analysis, diagnosis, and prognosis of complex embedded systems. This paper presents an efficient simulation scheme for systems with mixed continuous and discrete behaviors. We model hybrid systems using hybrid bond graphs (HBGs), a multi-domain physics-based modeling language that incorporates local switching functions that enable the reconfiguration of energy flow paths. We exploit the inherent causal structure in HBGs to derive hybrid simulation models as reconfigurable block diagram (BD) structures. Considerable computational savings are achieved during simulation by identifying fixed causal assignments when the simulation model is derived. Fixed causal assignments reduce the number of possible computational structures across all mode changes, and this leads to an overall reduction in the complexity of the BD simulation models and in their reconfiguration procedures. This approach has been implemented as a software tool called the **MO**deling and **T**ransformation of **H**BGs for **S**imulation (MOTHS) tool suite. Simulation models of an electrical power distribution system that includes a fast switching inverter system are derived using the MOTHS tool suite, and experimental studies on this system demonstrate the effectiveness of our approach.

1 Introduction

Accurate and efficient modeling and simulation approaches are essential for design, analysis, diagnosis, and prognosis of complex embedded systems. To address these needs, we have developed component-oriented modeling techniques based on hybrid bond graphs [1], and a model-integrated design methodology for efficient simulation that facilitates diagnosis and prognosis experiments [2, 3]. Building accurate and efficient simulation models for hybrid, nonlinear systems is not trivial, especially since the simulation must deal with the computation of nonlinear behavior and system reconfigurations that produce discrete behavior changes. For systems where reconfigurations occur at high frequencies, such as modern electrical power distribution systems with electronic switching and control [4], it is especially important to maintain accuracy in the generated behaviors without sacrificing simulation efficiency.

The bond graph (BG) modeling language allows for multi-domain, topological, lumped-parameter modeling of physical process, such as electrical power systems, by capturing their energy exchange mechanisms [5]. The nodes of a bond graph include primitive energy storage (C and I), energy dissipation (R), energy transformation (TF and GY), and energy source elements (Se and Sf). The n-port I (or C)-fields allow for extended energy storage models, where each I (or C)-field is defined by an $n \times n$ matrix. The connecting edges, called *bonds*, define energy pathways between elements. Each bond has two associated variables: effort, e, and flow, f, and their product represents the rate of energy transfer. In the electrical domain, effort denotes voltage and flow denotes current. Each BG element relates the effort and flow variables on the bonds connected to it. For example, since the voltage drop across a resistor is the product of its resistance and the current flowing through it, the constituent equation relating the effort and flow at a resistor is $e = Rf$, where R is the resistance. BG components are connected to one another using two idealized connection elements, the 0- and 1-junctions. For a 0-junction, the efforts of all incident bonds are equal, and the sum of flows is zero, while for a 1-junction, the flows of all incident bonds are equal, and the sum of efforts is zero. Therefore, in the electrical domain, the 0- and 1-junctions represent parallel and series connections, respectively.

In describing the input-output behavior of a component, the independent variable at each bond must be determined. *Causality* expresses the computational dependencies between the effort and flow variables in the BG components. For example, the causality at a resistor, R, determines whether the direction of computation is $e = Rf$, or $f = e/R$. Similarly, at a 1-junction, causality denotes which bond's flow defines the flow values of all other incident bonds. Visually, causality is represented by a causal stroke at the end of the bond where the effort is imposed. Consequently, a flow is imposed on the other end of the bond. For example, in Fig. 1, bond 2 imposes flow on the 1-junction, when on.

The continuous BG representation has been extended to model hybrid systems by several researchers [6–9]. Our

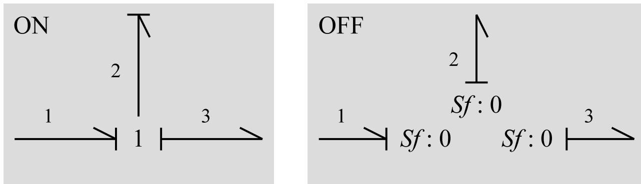

Figure 1: Semantics of a switching 1-junction.

approach, hybrid bond graphs (HBGs), introduces discrete mode changes through idealized *switching junctions* that can turn *on* or *off*, thereby reconfiguring the energy flow paths in the model [10]. In the electrical domain, for example, a switching junction represents an electric switch that can connect or disconnect different circuit components. A two state (*on* and *off*) finite state machine implements the junction *control specification* (CSPEC). Transitions between states may be functions of system variables and/or system inputs. When a switching junction is on, it behaves like a conventional junction. When off, all bonds incident on a 1-junction (or 0-junction) are deactivated by enforcing 0 flows (or efforts) on all bonds incident on that junction (see Fig. 1). The system mode at any given time is determined by composing the modes of the individual switching junctions. The system configuration in each mode implies the causal structure among the system variables.

In our work, we adopt the block diagram (BD) formalism as the computation model for the HBGs because the input-output formulation of each block in a BD can be determined using the causality information captured by HBGs. BDs are also advantageous because: (*i*) the BD formalism is a widely used computational scheme, and (*ii*) BD models preserve the component structure of the model, which facilitates introduction of faults into components for simulation-based diagnosis and prognosis studies.

When a junction switches on/off, it gains/loses its determining bond (see Fig. 1), thereby resulting in a change in the causality assignment, and hence the computational structure, at that junction. Moreover, the causality of adjacent elements may also need to be reassigned as a result of this change. Since mode changes result in causality reassignment in HBGs, we represent hybrid systems using *reconfigurable* BD models [2, 11]. These reconfigurable BD models include switching elements that enable the online reconfiguration of the BD components to account for different causality assignments in different system modes. Every time a mode change occurs, causalities are incrementally reassigned from the previous mode, and the effort and flow links are rerouted by the switching elements to ensure that the computational model matches the new causality assignment [2]. This approach, as is, produces acceptable results when mode changes are infrequent. However, for fast switching systems with frequent mode changes, e.g., electrical power conversion and distribution systems, invoking the procedure for reassigning causality at *every* mode change may produce unnecessary computational overhead, which leads to significant increases in the simulation execution times. Also, the BD models may include extraneous switching elements and signal connections,

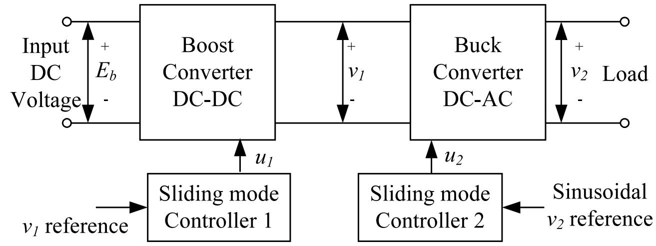

Figure 2: Block diagram of a boost-buck AC inverter.

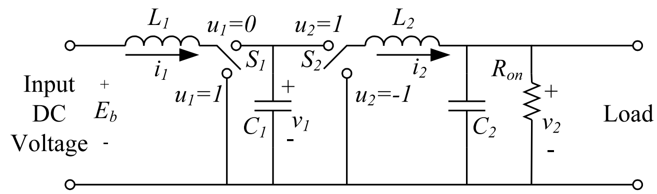

Figure 3: Circuit model of a boost-buck AC inverter.

which account for causality assignments that never occur during the simulation.

Hence, to improve the simulation efficiency, we identify bonds whose causal assignments are fixed across mode changes in the HBG model, and use this information to restrict possible reconfigurations that can occur in the simulation model. As a result, the generated BD models are space-efficient because the number of switches needed for each BD component, as well as the number of possible signal connections are reduced. We also confine the propagation of the effects of mode changes to only the required parts of the simulation model, which are typically small in number, thereby reducing the computational effort required to execute mode changes during simulation. We demonstrate the effectiveness of our approach by applying it to a power conversion and distribution testbed developed for diagnosis and prognosis studies at NASA Ames Research Center [3].

2 Motivating Example: AC Inverter

The Advanced Diagnostics and Prognostics Testbed (ADAPT) models aircraft and spacecraft power distribution systems [3]. It includes batteries for power storage, inverters for DC to AC power conversion, a power distribution network made up of a number of circuit breakers and relays, and a variety of DC and AC loads. In this paper, we focus on the AC subsystem, and develop the fast-switching inverter model to motivate our approach.

The inverter, a two-stage boost-buck DC-AC converter [4], consists of a cascade connection between a boost DC-DC converter with a full-bridge buck DC-AC converter to achieve a transformerless DC-AC step-up conversion (see Fig. 2). The boost converter boosts the input DC voltage to a higher value (190 V, in our case), and the buck converter stage generates the sinusoidal AC voltage. The fast switching in the boost and buck converters are governed by two sliding mode controllers, one for each stage of the inverter.

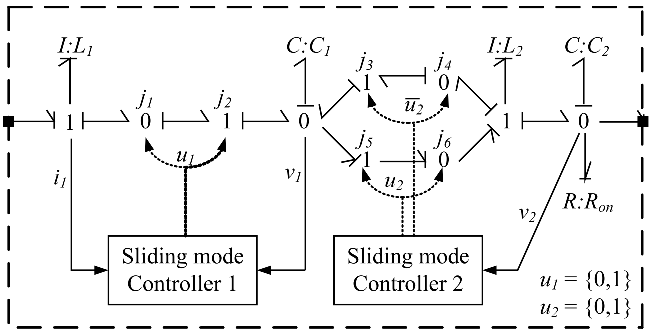

Figure 4: Inverter HBG component model.

Table 1: Inverter Model Parameter Values

Inductances (H)	$L_1 = 0.0022$, $L_2 = 0.075$
Capacitances (F)	$C_1 = 0.0069$, $C_2 = 6 \times 10^{-6}$
Resitances (Ω)	$R_{on} = 489.49$
Sliding mode controller 1 parameters	$\alpha = 0.8$, $\beta = 4.3649$ $\delta = 111.375$, $K = 829.3347$
Sliding mode controller 2 parameters	$a_1 = 15.915$, $a_2 = 0.0048$
Reference Voltages (V)	$v_{1Ref} = 190$, $v_{2Ref} = 120\sqrt{2}\sin(120\pi)$

The equivalent circuit model of the boost-buck DC-AC inverter is shown in Fig. 3, where S_1 is a conventional power switch, and S_2 corresponds to a full bridge switch. The control signals for S_1 and S_2 are u_1 and u_2, respectively. The differential equations for the system can be found in [4], and the model parameters are listed in Table 1. The internal resistance R_{on} accounts for the current that the inverter draws from the battery when it is disconnected from its loads.

The HBG model of the inverter, derived from its circuit model, is shown in Fig. 4. Switch S_1 is represented by the synchronous switching junctions j_1 and j_2, i.e., they share the same CSPEC function. Switch S_2, is represented by the switching junctions $j_3 - j_6$, with j_3 and j_5 having the same CSPEC as junctions j_4 and j_6, respectively. The switching conditions for junctions j_3 and j_4 are logical negations of those for junctions j_5 and j_6.

The sliding mode controllers are also modeled using HBGs. They generate signals that switch the inverter junctions at kilohertz rates. One approach to simulating the HBGs would be to invoke the causal reassignment procedure at every mode change to compute the updated model configurations before the continuous simulation is resumed [2]. However, careful inspection of the inverter HBG model shows that the causality assignments at the switching junctions remain the same when the junctions are on. When the junctions are off, the causal changes do not propagate to adjacent junctions (see the next section for details). Therefore, the calls to the causal reassignment procedure at every mode change are not necessary, and should be avoided since they considerably slow down the execution of the simulation. If we can identify the causality assignments that do not change when reconfigurations occur, the number of calls to the causal reassignment procedure can be reduced, and the model reconfiguration task can be simplified. The knowledge of causality assignments that do not change when reconfigurations occur can also be used to make the simulation models more efficient by not modeling system configurations that will never occur during system operation.

3 Efficient Simulation of Hybrid Bond Graph Models

Efficient simulation models for hybrid systems should meet two primary requirements: *(i)* avoid pre-enumeration of all system modes, especially for systems with a large number of modes, and *(ii)* minimize the amount of computations performed to handle mode changes. Reassignment of causality produces changes in the computational model. But, we can minimize the number of changes during reconfiguration by *(i)* recognizing causal assignments that are fixed across all modes, and *(ii)* not allowing configurations that we can predetermine will never occur. As a result, we reduce the search space for the causal propagations possible as a result of the mode switch and also simplify the simulation models.

3.1 Converting Bond Graphs to Block Diagrams

Fig. 5 shows the possible causal assignments for all BG elements [5]. The Sf, Se, C and I elements each have a unique causal assignment on their incident bonds which remain the same in all modes of system operation. In our work, we assume that the energy storage elements (C and I) are in *integral causality*, i.e., the computational models for these elements are in integral form, e.g., $e = \frac{1}{C}\int f dt$. The corresponding differential form, i.e., $f = C\frac{de}{dt}$, may introduce computational problems during simulation, and also requires knowing a future value to compute the derivative at the current time point [5]. The n-port I- and C-fields also have unique causal assignment across all system modes, and are not shown in Fig. 5 as they are simple functional and structural extensions to the 1-port I and C elements, respectively. The R, TF, and GY elements allow two possible causal assignments each, and each assignment produces a different BD model.

We capture the notion of causality assignment at a junction through the commonly used notion of the *determining bond*.

Definition 1 (Determining Bond) *The* determining bond *of a* 0- *(*1-*) junction is the bond that establishes the effort (flow) value of all other bonds incident on the junction.*

Mapping a junction to its BD model is facilitated by its determining bond. Fig. 6 shows the BD expansion for nonswitching junction (ignoring the off configuration). At a 1-junction, all other bonds' flow values are equal to the determining bond's flow value, and the effort value of the determining bond is the algebraic sum of the effort values of the other bonds connected to this 1-junction, taking into account the direction of these bonds. A nonswitching junction with m incident bonds can have m possible BD configurations.

A standard algorithm for assigning causality to a BG is the *Sequential Causal Assignment Procedure* (`SCAP`) [5]. The basic idea of `SCAP` is to start at elements having a

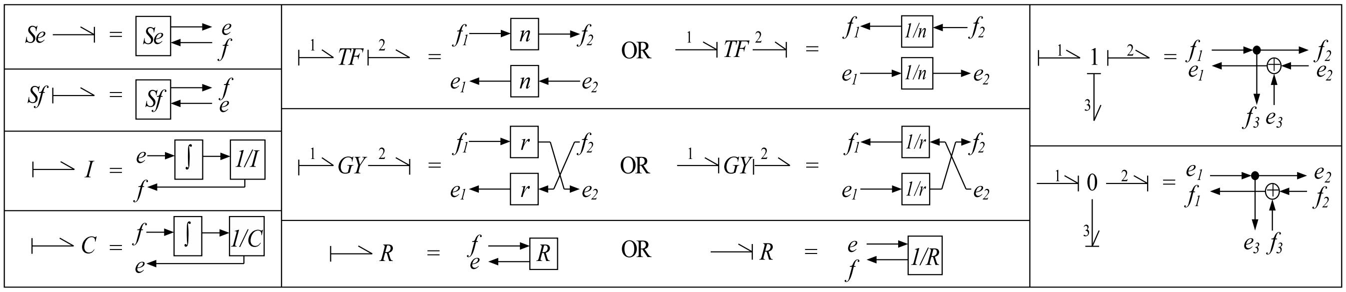

Figure 5: Computational structures for bond graph elements.

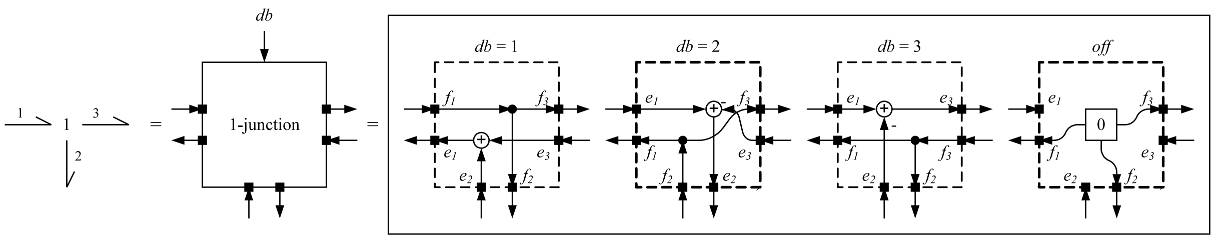

Figure 6: Block diagram expansion of a switching 1-junction (db denotes determining bond).

unique causality, such as energy sources and energy storage elements, to constrain the possible options for determining bonds for the junctions these elements are connected to, which in turn may constrain the options for determining bonds for adjacent junctions. If there exists a *unique* option for a determining bond for a junction, the junction is assigned that determining bond, and the constraints imposed by this assignment are propagated along the BG to further restrict the possible options for determining bonds at other junctions. After the causal changes have been propagated from all energy sources and energy storage elements, if a junction still has multiple options for its determining bond, one of its bonds with unassigned causality is arbitrarily assigned as its determining bond, and the constraints imposed by this assignment are propagated along the BG. `SCAP` terminates when every junction is assigned a determining bond.

Once a BG is assigned causality, a well-defined procedure can be applied for converting the BG structure to a BD model [5]. Based on the assigned causality, each BG element is replaced by the computational structure (see Figs. 5). In the BD model, each bond is replaced by two signals, i.e., the effort and flow variables for the bond. Once all necessary blocks of the BD are instantiated, they are connected appropriately to complete the BD model.

3.2 Converting Hybrid Bond Graphs to Block Diagrams

The BD generation procedure described above needs to be extended for HBGs so that they can handle causality changes that occur due to mode changes. Instead of rebuilding the entire BD model every time mode changes occur, we include switching elements in the individual BD components to reconfigure the computational model during simulation. For example, a switching junction with m incident bonds can switch between $m+1$ possible computational configurations, m corresponding to each incident bond being a determining bond, and one corresponding to the junction being off, in which each outgoing signal is set to zero. With this method, the physical connections between blocks are fixed, but the interpretation of the signal on the connection (effort or flow) changes depending on the causal assignment to the bonds (see Fig. 6).

In some cases, however, the causal assignment for a bond is invariant across *all* possible modes of system behavior, i.e., the causal assignment is *fixed*. For example, a C-element will always impose effort on a 1-junction through the connecting bond. In this case, the BD for the 1-junction does not need to include any switching mechanism to accommodate the possibility of this bond being its determining bond. If all bonds of a junction have fixed causal assignments, then its determining bond is invariant for all modes of the system. In this case, the BD for the junction does not need to include any switching mechanisms because it can assume a fixed structure. In previous work, we have termed a nonswitching junction with this property to have *fixed causality* [2]. Switching junctions, by definition, change causality when they turn off, but this change may not affect adjacent junctions. Therefore, we extend our previous definition of fixed causality to also include switching junctions.

Definition 2 (Fixed Causality) *A junction that does not switch is in* fixed causality *if, for* all *modes of system operation, its determining bond does not change. A switching junction is in* fixed causality *if, for* all *modes in which the junction is on, its determining bond does not change, and for all modes where it is off, the inactivation of its incident bonds does not affect the determining bond of any of its adjacent junctions.*

Fixed causality of bonds and junctions can be identified efficiently using a `SCAP`-like algorithm before we construct the BD model from the HBG. In this algorithm, the causality assignment is first performed at junctions connected to sources and energy storage elements, because the bonds connecting them to these junctions have fixed causality. A 0- (or 1-) junction is in fixed causality if it is connected to a Se (or Sf) or a C (or I) element. Otherwise, a junction is in fixed causality if *(i)* its determining bond connects to a fixed causality junction (either directly, or through a TF or a GY element), or *(ii)* all incident bonds other than its determining bond are connected to fixed causality junctions. Once a junction is determined to be in fixed causality, we propagate this information to its adjacent junctions to check if they too are in fixed causality, or any of their bonds are in fixed causality.

Some additional analysis is required to determine whether a switching junction is in fixed causality. A switching junction is in fixed causality if: (*i*) whenever the junction is on, its determining bond is the same, and (*ii*) when the junction turns off the determining bond for its adjacent junctions should not change, i.e., none of its incident bonds can be a determining bond for its adjacent junctions. If two switching junctions are adjacent and share the same CSPEC, the knowledge that they switch together can help determine if causal changes will propagate when they switch. When we visit a junction for the first time, all its adjacent junctions may not have been checked for fixed causality yet. Hence, the causality may need to be propagated from all adjacent junctions before it can be determined that the junction is in fixed causality.

If a junction has incident bonds with fixed causality, or if the junction itself is in fixed causality, the computational model of the junction block can be reduced by eliminating switching elements and signal connections which account for causality assignments that will never occur during simulation. Consider the 3-port switching 1-junction in Fig. 6. If the junction is not in fixed causality, its implementation can, in general, switch between four possible configurations as mode changes occur. However, if the junction is in fixed causality, the BD representation for this junction has only one valid on configuration, in addition to the off configuration. If the junction is not in fixed causality, for example, if its bond 1 is connected to a Se, its BD representation need not include a configuration with bond 1 as its determining bond. Switched junctions in fixed causality help minimize causality reassignment computations when mode changes occur. Therefore, when a 1-junction in fixed causality changes mode, we know exactly what the causality assignment at this junction is without having to call any external causality reassignment procedure, and can build this into the BD.

3.3 Efficiently Reassigning Causality in Hybrid Bond Graphs

The naïve approach to causality reassignment in HBGs is to run `SCAP` on the new HBG configuration *every* time a mode change occurs, but this approach is likely to be inefficient for the following reasons: (*i*) usually, only a small part of the HBG needs to be reassigned causality, and (*ii*) sometimes, changes in causality do not propagate and the effect of a mode change only produces local changes in the computational model structure.

Our causality reassignment method, called the *Hybrid Sequential Causal Assignment Procedure* (`Hybrid SCAP`) reassigns causality incrementally, starting from the junctions directly affected by the switching, and then propagating the changes only to those junctions whose causal assignments are affected by changes in the adjacent junctions using the causal assignment of the previous mode [2, 11]. At junctions where a unique choice for a new determining bond is not known, an arbitrary choice may be made. But, this choice may lead to an inconsistent assignment if the propagation reaches a fixed causality junction. An inconsistent assignment can also be made if the propagation reaches a junction whose determining bond has been unequivocally assigned for that particular system mode. Such junctions are considered to be in *forced causality*.

Definition 3 (Forced Causality) *A junction is in* forced causality *if it can be assigned only one possible determining bond in a given system mode.*

We use the knowledge of junctions in fixed and forced causality to reduce the search space for the `Hybrid SCAP` algorithm by not propagating causal changes across fixed and forced causality junctions. As a result, expensive backtracking is avoided. The worst case computational complexity of our `Hybrid SCAP` approach is polynomial in the size of the HBG, as it is similar to the `SCAP` algorithm. The average case complexity of our approach, however, is better than that of `SCAP`, since in many cases, only small parts of the HBG change causality.

Consider the inverter HBG model (Fig. 4), where all non-switching junctions are in fixed causality. The incident energy storage elements specify a unique determining bond for these junctions. All switching junctions are also fixed. Consider switching junctions j_1 and j_2. Since they always change modes simultaneously (because they share the same CSPEC), when on, j_1 always imposes flow on its adjacent junction j_2 which is also on. When they are off, the causality assignment of other active junctions are not affected. The case is similar for pairs j_3 and j_4, and j_5 and j_6. Since all junctions are in fixed causality, the mode switchings in the inverter do not require reassignment of causality, because the changes never propagate. Therefore, `Hybrid SCAP` is not invoked, and minimal changes have to be made to the computational model when mode changes occur, thus considerably speeding up the inverter simulation, as we illustrate later.

3.4 Simulating the Block Diagrams

Our modeling and simulation approach defines the reconfigurable BDs and their reconfiguration procedures, and therefore is independent of the underlying fixed or variable-step solver being used by the simulation environment in which the simulation model is executed. Once the reconfigurable BD model is generated in the simulation environment, simulation starts with the BD structure corresponding to the current mode, and the simulation continues till a mode change occurs. If the mode change is attributed to the switching of junctions that are not in fixed causality, the simulation is paused, the `Hybrid SCAP` algorithm is invoked to reassign causality, and the BD is reconfigured accordingly before the simulation resumes. On the other hand, if a mode change is attributed to a switching junction with fixed causality, the BD is reconfigured based on the junction mode, without invoking `Hybrid SCAP`.

4 Software Implementation

We have implemented our methodology using a model-integrated computing approach [12] as the **MO**deling and **T**ransformation of **H**BGs for **S**imulation (MOTHS) tool suite [13]. It consists of a modeling language for building graphical component-based HBG models [1], and a set of model translators, or *interpreters*, that convert these HBGs into block diagram-based simulation models for selected simulation tools. Initially, we have developed an interpreter that creates MATLAB Simulink models, details of which can be found in [2].

Simulation testbeds have a variety of applications. One of the primary objectives of our simulation approach is to facilitate prognosis and diagnosis experiments. Hence, we have developed schemes to inject faults in the simulation model. We model faults as off-nominal changes in system parameters (e.g., a resistance change representing a blockage in a pipe), called *parametric faults*, and off-nominal changes in system mode (e.g., a stuck valve), called *discrete faults*.

Our approach facilitates fault injection in Simulink through a graphical user interface that is constructed automatically during the interpretation process. The user can select the time of fault occurrence (we consider only persistent faults), the fault profile, and the fault magnitude (where applicable). Parametric faults are associated directly with HBG element parameter values (i.e., Se, C, R, etc.), and the available fault profiles include abrupt, incipient, and stuck-at faults. An abrupt fault is a fast change in a system parameter whose time constant is much smaller than that of the system dynamics, and is modeled as a constant bias term that gets added to the nominal parameter value at the time of fault occurrence. An incipient fault is a slow change in system parameters, and modeled as an linear drift term (with constant slope) that is added to the actual parameter value. A stuck-at fault sets the parameter value to a constant value at fault occurrence. Discrete faults cause the system mode to change, so they affect the switching junctions. Our simulation approach allows the injection of *stuck-on*, *stuck-off* faults, introduced by adding fault terms to the CSPEC transition guards. As a special case, sensor faults can typically be modeled as either parametric or discrete. In our approach, we treat them as parametric faults, allowing abrupt (bias), incipient (drift), and stuck-at faults (which can also model discrete faults such as sensor failure). Additionally, we allow the introduction of white Gaussian sensor noise, where the mean and variance can be specified. A change in variance can also be introduced as a sensor fault.

5 Case Study

We demonstrate our modeling and simulation framework using the ADAPT system at NASA Ames Research Center [3]. In this paper, we focus on a subsystem of ADAPT shown in Fig. 7, i.e., a battery driving an inverter that is connected to two loads through two relays. One of these loads is a light bulb, while the other load is a large fan.

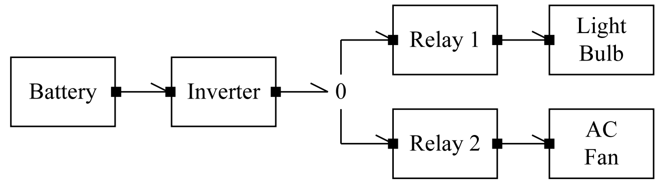

Figure 7: ADAPT subsystem for case study.

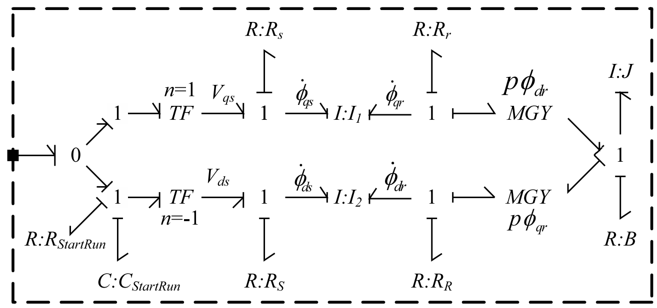

Figure 8: AC fan hybrid bond graph.

We build the system model using a component-based modeling approach [1, 3]. We use the HBG model of the inverter shown in Fig. 4 for our experiments. The HBG model of the lead-acid battery is based on an electrical equivalent circuit model, which captures the nonlinear battery behavior [11]. We omit a detailed description of the battery model due to space constraints. The resistive light bulb load consists of a single power port, a 1-junction, and a R element. The R element parameter value is a constant $234.6966\ \Omega$.

The AC fan is modeled as a single phase, fixed capacitor induction motor [14]. We represent the system using the standard $d - q$ model, described in [15]. We do not present the fan model equations due to space constraints. The model parameters are presented in Table 2. The HBG model of the fan, shown in Fig. 8, is adapted from that described in [16]. Note that the AC fan has two I-fields with parameters,

$$I_1 = \begin{bmatrix} L_{ss} & L_{ms} \\ L_{ms} & L'_{rr} \end{bmatrix}, \text{ and } I_2 = \begin{bmatrix} L_{SS} & L_{ms} \\ L_{ms} & L'_{RR} \end{bmatrix}.$$

Table 2: AC Fan Model Parameter Values

Inductances (in H)	$L_{ss} = 0.275$, $L_{SS} = 0.274$, $L'_{rr} = 0.272$, $L'_{RR} = 0.271$, $L_{ms} = 0.1772$, $L_{mS} = 0.2467$
Inertia (in kg m^2)	$J = 6.5 \times 10^{-4}$
Resistances (in Ω)	$R_s = 163.02$, $R_S = 168.14$, $R'_r = 145.12$, $R'_R = 145.12$, $R_{startRun} = 26$,
Friction (in kg m^2/s)	$B = 4.734 \times 10^{-4}$
Capacitances (in μF)	$C_{startRun} = 21.1$
Other Parameters	$N_S/N_s = 1.18$, $p = 2$

Table 3: Real Time Taken for 1 s of Simulation Time

Method	Real Time
`SCAP` called at every mode change	6054.3 s
`Hybrid SCAP` called at every mode change	6025.2 s
No causality reassignment procedure called	58.3 s

5.1 Experimental Results

For the experimental results, we automatically derived a Matlab® Simulink® model of the subsystem shown in Fig. 7, from the HBG model of the system using the MOTHS tool suite. All experiments were performed on a 2.4 GHz Intel® Pentium Core™2 Duo CPU desktop, with 2 GB RAM. The model was simulated using a fixed-step simulation with a sample period of 7.5 μs.

Fig. 9 shows the results of the simulation. We plot the voltages and currents at the output of the battery and the inverter, as well as the rotational speed of the AC fan. The simulation model was run for 20 seconds of simulation time. In the first configuration, the light bulb is connected to the inverter from $2-5$ seconds, followed by a second configuration where the AC fan is turned on between $7-15$ seconds. An abrupt fault is injected in the light bulb resistance at 3 seconds to demonstrate the usefulness of the simulation approach for diagnosis applications. As we can see, the sliding mode controllers are robust to load changes, and generates true 120 V rms voltage for both the loads. However, the light bulb fault affects the inverter current, and, therefore, the battery current and voltage. The AC fan current shows a phase difference of 0.1346 rad. As can be seen in Fig. 9 when the AC fan is switched on, its speed of rotation increases until it reaches a steady state of about 78.5 rad/s. On turning off, the speed falls to zero.

Table 3 presents the result of an experiment to illustrate the efficiency gained by simplifying a reconfigurable BD model by identifying bonds with fixed causal assignments and junctions in fixed causality, and avoiding the need for causal reassignment for these modes. For this experiment, we assume that the AC fan is the only load and is on for the duration of the experiment. Each column in Table 3 reports the real time taken to simulate 1 second of simulation time for different HBG simulation runs. In all runs, all junctions of the HBG model are in fixed causality. In the first run, we call `SCAP` every time an inverter mode change occurs. Next, we repeat the previous run, using `Hybrid SCAP`. Finally, in the third run, we simulate the HBG without requiring any external calls to `Hybrid SCAP`, since all switching junctions are fixed. As can be seen from Table 3, our enhanced simulation approach, implemented in the third run, is 103.85 times faster than the first run, and 103.35 times faster than the second run. Our simulation approach also resulted in considerable improvements in the efficiency of simulation of a number of other configurations, especially for large systems like the VIRTUAL ADAPT simulation testbed [3]. Further increase in simulation efficiency can be obtained by running our simulation models in the Rapid Accelerator mode of Simulink.

6 Conclusions

In this paper, we have presented an approach for modeling and simulation of complex systems with switching behavior using hybrid bond graphs. The crux of this modeling and simulation framework, implemented as the MOTHS tool suite, is the identification of fixed causality of bonds, which not only avoids unnecessary invocations of the external `Hybrid SCAP` algorithm, thereby gaining increase in simulation efficiency, but also improves the efficiency of the `Hybrid SCAP` algorithm, as well as enables the simplification of the simulation models by removing parts that correspond to configurations that never occur during the simulation. In the future, we will extend our modeling approach and computational model generation schemes to systematically evaluate how our approach performs when applied to other real-world large hybrid systems. We also wish to develop interpreters to generate simulation models in other simulation software, such as Ptolemy [17].

Acknowledgements

This work was supported in part by the National Science Foundation under Grants CNS-0615214 and CNS-0347440; and NASA NRA NNX07AD12A.

References

[1] E.-J. Manders, G. Biswas, N. Mahadevan, and G. Karsai, "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: IEEE CS Press, Mar. 2006.

[2] I. Roychoudhury, M. Daigle, G. Biswas, X. Koutsoukos, and P. J. Mosterman, "A method for efficient simulation of hybrid bond graphs," in *Proceedings of the International Conference of Bond Graph Modeling*, San Diego, California, 2007, pp. 177 – 184.

[3] S. Poll, A. Patterson-Hine, J. Camisa, D. Nishikawa, L. Spirkovska, D. Garcia, D. Hall, C. Neukom, A. Sweet, S. Yentus, C. Lee, J. Ossenfort, I. Roychoudhury, M. Daigle, G. Biswas, X. Koutsoukos, and R. Lutz, "Evaluation, selection, and application of model-based diagnosis tools and approaches," in *AIAA Infotech@Aerospace 2007 Conference and Exhibit*, May 2007.

[4] D. Biel, F. Guinjoan, E. Fossas, and J. Chavarria, "Sliding-mode control design of a boost-buck switching converter for ac signal generation," *IEEE Transaction on Circuits and Systems - I*, vol. 51, no. 8, pp. 1539–1551, 2004.

[5] D. C. Karnopp, D. L. Margolis, and R. C. Rosenberg, *Systems Dynamics: Modeling and Simulation of Mechatronic Systems*, 3rd ed. New York: John Wiley & Sons, Inc., 2000.

[6] J. Buisson, H. Cormerais, and P.-Y. Richard, "Analysis of the bond graph model of hybrid physical systems with ideal switches," *Proc Instn Mech Engrs Vol 216*

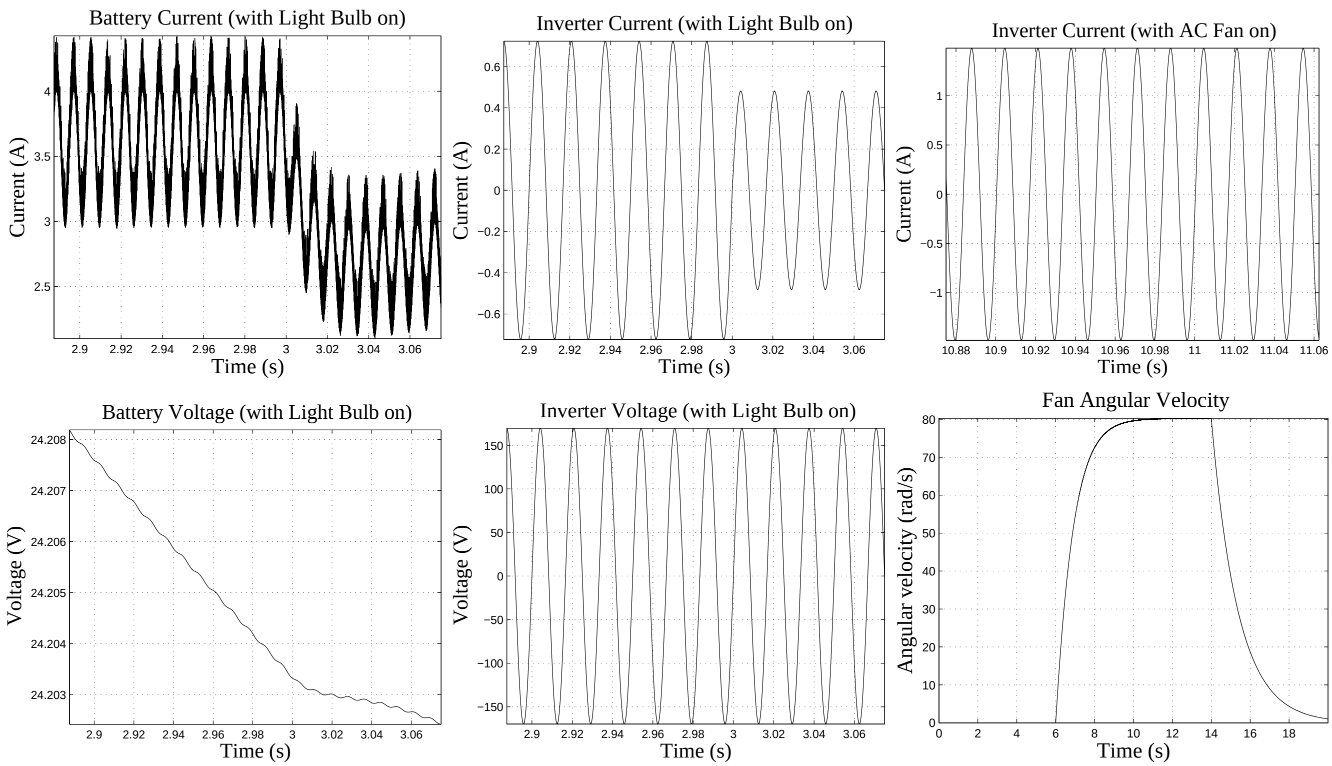

Figure 9: Simulation results.

Part I: J Systems and Control Engineering, pp. 47–63, 2002.

[7] M. Magos, C. Valentin, and B. Maschke, "Physical switching systems: From a network graph to a hybrid port hamiltonian formulation," in *Proc IFAC conf Analysis and Design of Hybrid Systems*, Saint Malo, France, June 2003.

[8] J. van Dijk, "On the role of bond graph causality in modeling mechatronics systems," Ph.D. dissertation, University of Twente, Enschede, The Netherlands, 1994.

[9] W. Borutzky, "Discontinuities in a bond graph framework," *Journal of the Franklin Institute*, vol. 332, no. 2, pp. 141–154, 1995.

[10] P. J. Mosterman and G. Biswas, "A theory of discontinuities in physical system models," *Journal of the Franklin Institute*, vol. 335B, no. 3, pp. 401–439, 1998.

[11] M. Daigle, I. Roychoudhury, G. Biswas, and X. Koutsoukos, "Efficient simulation of component-based hybrid models represented as hybrid bond graphs," in *HSCC 2007*, ser. LNCS, A. Bemporad, A. Bicchi, and G. Butazzo, Eds. Springer-Verlag, 2007, vol. 4416, pp. 680–683.

[12] J. Sztipanovits and G. Karsai, "Model-integrated computing," *Computer*, vol. 30, no. 4, pp. 110–111, Apr 1997.

[13] MOTHS, "http://macs.isis.vanderbilt.edu/software."

[14] G. J. Thaler and M. L. Wilcox, *Electric Machines: Dynamics and Steady State*. John Wiley & Sons, Inc., 1966.

[15] P. C. Krause, *Analysis of Electric Machinery*. John Wiley & Sons, Inc., 1986.

[16] D. Karnopp, "Understanding induction motor state equations using bond graphs," in *Proceedings of the International Conference on Bond Graph Modeling and Simulation*, vol. 35, no. 2, 2003, pp. 269 – 273.

[17] J. Buck, S. Ha, E. A. Lee, and D. G. Messerschmitt, "Ptolemy: a framework for simulating and prototyping heterogeneous systems," *Readings in hardware/software co-design*, pp. 527–543, 2002.

Corresponding Author:

Gautam Biswas, Indranil Roychoudhury, Xenofon Koutsoukos, Institute for Software Integrated Systems, Dept. of EECS, Vanderbilt University, Box 1829, Station B, Nashville, TN 37235, USA. {*gautam.biswas, indranil.roychoudhury, xenofon.koutsoukos*}@*vanderbilt.edu*

Matthew Daigle, University of California, Santa Cruz, NASA Ames Research Center, Moffett Field, CA 94040, USA. *matthew.j.daigle@nasa.gov*

This contribution is a revised and extended version of our paper: "An Efficient Simulation of Hybrid Systems: An Application to Electrical Power Distribution Systems," in *Proceedings of the 22nd European Conference on Modeling and Simulation*, pp. 471-477, June 2008.