Modeling Discontinuous Behavior with Hybrid Bond Graphs

Pieter J. Mosterman
Center for Intelligent Systems
Vanderbilt University
Nashville, Tennessee 37235
pjm@vuse.vanderbilt.edu

Gautam Biswas
Department of Computer Science
Vanderbilt University
Nashville, Tennessee 37235
biswas@vuse.vanderbilt.edu

Abstract

This paper discusses a technique that combines bond graph energy-flow modeling and signal-flow modeling schemes for modeling discontinuities in physical systems. This enables the generation of complex, multi-mode behaviors without violating the energy flow principles imposed by bond graphs. Mode switching is achieved by *controlled junctions* which assume one of two states: *on* and *off*. The control function of the junctions are specified as state transition graphs or tables. The modeling methodology presents a common framework for modeling abrupt switching elements, piecewise linear components, and subsystems that undergo structural changes without having to pre-enumerate system modes.

1 Introduction

Recent advances in model-based and qualitative reasoning have led to researchers developing large scale models of complex, continuous systems, such as power plants, aircraft, and space station subsystems. Configuration and individual component description changes cause multi-mode behavior in complex systems[4]. A primary reason for component description changes is the representation of complex non linear behaviors as simpler, discontinuous, piecewise linear behaviors[16].

Biswas and Yu[2] have discussed the effectiveness of the bond graph modeling language in defining formal but generic models of physical systems. These models are based on energy transfers between mechanisms of the system and conservation of energy within the system, but they do not gracefully accommodate mode-switching of components as part of the modeling framework. This paper extends and formalizes previous work on the direct method for analyzing discontinuous behavior by Nishida and Doshita[16], the finite state models for representing discontinuous behavior by Broenink and Wijbrans[4], and ideal switching elements by Strömberg, Top, and Söderman[20, 21] by developing a signal flow model on primary bond graph models to incorporate *mode-change* behaviors in systems. Signal flow models incorporating finite state automata have been used extensively in real-time system analysis[8]. Finite state automata can also model ideal switching elements with complex sequential behavior while avoiding spurious junctions and incorrect causal relations. At the same time, the relations between bond graph mechanisms and corresponding physical components of the system are maintained. Therefore, they provide an integrated framework for hierarchical modeling and behavior analysis of complex, continuous systems.

2 Modeling Physical Systems

The ability to dynamically construct parsimonious system models is the key to developing computationally efficient behavior generation methods. This requires model building methodologies that satisfy *compositionality*, *graceful extendibility*, and *genericity*[6]. *Bond graphs*[18, 19] introduce the notion of generic physical mechanisms and state variables in modeling. This provides a well-defined formal vocabulary for describing system components and their interactions and creates a framework for building physical models over a wide range of domains[22].

Bond graphs, as a modeling language, provide a set of highly organized domain independent structures that are based on a small number of primitive elements: resistances or dissipators (R), energy storage elements (capacitors C and inertial elements I), ideal sources of effort S_e and flow S_f, and transformational elements (transformer TF and gyrator GY)[10, 19]. These elements take on different physical forms in different domains, but interactions between them are expressed in a domain independent way: as energy transfers which are represented as *directed bonds*. Each bond has an associated *effort* and a *flow variable*, where *effort* $\times$ *flow* = *power*, the rate of energy transfer. Connections between multiple elements are established by energy conserving *junctions*, which can be of two types: *common flow* (i.e., series) or 1-junctions, and *common effort* (i.e., parallel) or 0-junctions.

Systematic procedures exist for building bond graph models of physical systems, deriving causal structure from the model[19], and building complex system models by composition[2]. Analytic models which express system behavior as a set of first order differential

equations[19] form the basis for deriving a spectrum of qualitative to quantitative models from this framework. Compared to other qualitative reasoning approaches, bond graphs use additional energy conservation and continuity constraints to dramatically reduce the number of spurious behaviors generated[20, 22]. Bond graphs support hierarchical model construction schemes by allowing compilation of these models from a library of sub-systems[2].

Typically, qualitative simulation methods (e.g., [7, 11]) impose *continuity constraints* to ensure meaningful behavior generation. Computational complexity in behavior generation is reduced by *linearizing* model constraints, and complex non linear behavior of devices such as oscillators and transistors, is simplified and represented as *multiple-piecewise* linear behaviors. In other situations, abstractions in time are employed to simplify physical component behaviors, so components such as electric switches, hydraulic valves, diodes, and mechanical clutches, exhibit *abrupt* changes in behavior. In either case, the physical components are said to operate in *multiple-modes*[1]. Abrupt changes in structure, such as breaks in pipes and wires, and overflow from tanks, also cause discontinuity in system behavior. In qualitative simulation systems, such as QSIM[11], a set of QDE's define the continuous behavior of a single mode of the system. Discontinuous changes are handled by defining multiple QDE sets, and a higher level control structure (meta-model) determines when to switch QDE sets during behavior generation.

Broenink and Wijbrans[4] attribute discontinuities in physical system behavior[1] to

1. piecewise continuous characteristics, e.g., simplifying the relation between current and voltage at transistor junctions to piecewise linear components,
2. physically possible structural changes in a system, e.g., a valve closing that isolates two connected tanks, and
3. physically impossible structural changes, primarily attributed to time-scale abstraction that makes certain changes in a system look instantaneous, e.g., a ball hitting a wall, where the time scale of the collision is considered to be instantaneous, and the ball seems to undergo an abrupt change in velocity.

In earlier work, Nishida and Doshita [16] propose similar causes for discontinuous behavior: (i) discontinuous input, (ii) mode transitions of multi-mode devices, and (iii) positive feedback without time delay. We contend that all discontinuities in the modeling of physical systems can be attributed to *abstracting* component behavior to simplify the relations among parameters (e.g., a transistor) or to simplify the time-scale of the interactions (e.g., the bouncing ball). Our goal is to derive a uniform approach to analyzing discontinuous system behavior.

[1] In reality all physical systems exhibit continuous behavior.

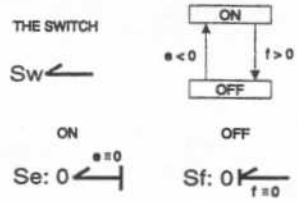

Figure 1: **The Switch**

Nishida and Doshita[16] point out two properties of discontinuous systems: (i) the causal structure of the system may change during a discontinuous change, and (ii) a number of discontinuous changes may occur one after another. The change of causality, which may be drastic when a series of changes are involved, is easily derived once an energy-flow bond graph is established in the new situation[3]. However, it is often not clear how to propagate a series of discontinuous changes. Analytic system models do not support this kind of information, and this becomes a source of problems during dynamic behavior generation. Therefore, a dedicated formalism is required to address this issue in the bond graph framework.

3 Bond Graphs and Discontinuous Behavior

Like qualitative simulation approaches, the bond graph methodology is geared toward modeling and analysis of continuous physical systems. To accommodate abrupt switching and non linear behavior, initial attempts were made to introduce switching elements into bond graphs in the form of *modulated transformers* (MTF)[10, 19] and non linear resistive elements[9]. However, these methods resulted in non intuitive causal structures and produced energy dissipation, therefore, could not model ideal switching elements.

More recently, Broenink and Wijbrans[4] introduced the concept of *switching bonds*, which are used to connect or disconnect sub-models based on principles of energy transfer. The switching bond is connected between two junctions and their positions are determined by a systematic modeling process. A *control box*, implemented as a global finite-state automaton, connected to the switching bond determines its *on* and *off* states. The shortcomings of this methodology are that it can cause hanging junctions, and changing boundary conditions due to switching are incorrectly handled.

To address these problems, Strömberg, Top and Söderman[20, 21] introduced an ideal switching element into the bond graph modeling language. A graphical description of the switch is shown in Fig. 1. Discontinuous changes are handled by enforcing either *zero flow* (switch *off*) or *zero effort* (switch *on*) on a junction. In both these modes, *power* = *effort* × *flow* = 0, which is the key in representing physical discontinuities using an ideal switching element (no energy loss)[23]. Note that the *on* and *off* modes of the switch

are degenerate forms of the effort- and flow-source elements of the bond graph, respectively.

However, ideal switches are found to have problems[14].

- It is unnatural to consider switches as bond graph elements; unlike other bond graph elements they have no energy-related functions,
- Switches represent *transient* elements; their behavior is based on control logic rather than physical concepts.
- A related point is that they obscure hierarchical compositional modeling schemes, which allow a system to be defined in terms of multiple models, each model representing an individual mode of system behavior. Dynamic behavior generation may involve model-switching to account for the different behavioral modes. A number of ideal switching elements may be employed to achieve model switching dynamically, but, in this representation, the link between bond graph elements and parts of the physical system become obscure and non-intuitive.

Our solution to this problem is to employ a hybrid representation scheme that combines

1. traditional energy-related bond graph elements to model the physical components of the system, and
2. control flow models based on switching junctions whose *on–off* characteristics are modeled by finite state automata.

This work extends intuitive notions presented in [1].

4 The Hybrid Modeling Approach

Previous work on real-time systems[24] demonstrates that system behavior is a function of time and event dependency, therefore, these system models must incorporate dynamic views with different system configurations and states. In general, a system with n such components each with k behavior modes, can assume k^n overall configurations or behavior modes, however, in practical situations, only a small fraction of the modes are physically realized. In previous approaches, e.g., [1, 4, 11], this knowledge has been exploited to pre-enumerate global behavior modes and specify transition functions between the modes as rules or state transition tables. However, this approach will not work for systems whose range of behaviors have not been pre-enumerated. Recent compositional modeling approaches[6, 15] overcome this problem and build system models *dynamically* by composing model fragments. Our approach adopts this methodology, and implements a dynamic but rigorous model switching methodology in the bond graph modeling framework.

Instead of identifying a global control structure and pre-enumerating bond graph models for each of these modes, we translate the overall physical model to one bond graph model that covers the energy flow relations within the system. Next, the discontinuous mechanisms and components in the system are identified. Each discontinuous mechanism is modeled locally as a *controlled junction* which can assume one of two states – *on* and *off*[2]. A local control mechanism, implemented as a *finite state automaton* and represented as a *state transition graph* or *table*, is used to model each discontinuity. A controlled junction behaves like an *idealized switch*. The input to its control mechanism are *effort* values from selected *0-junctions* and *flow* values from selected *1-junctions* and its output is a control signal that determines the *on/off* state of the junction. The set of local control mechanisms associated with controlled junctions constitute the *signal flow model* of the system. The signal flow model performs no energy transfers, therefore, it is distinct from the bond graph model that deals with the dynamic behavior of the physical system variables. Interactions between the two models are established by tapping appropriate effort and flow energy variables from bond graph junctions and generating output signals that determine the *on* and *off* states of controlled junctions. To summarize, bond graph models deal with energy-related behavior whereas signal flow models describe the *transient*, i.e., mode-switching behavior of the system.

A *mode* of a system is determined by the combination of the on/off *states* of all the controlled junctions in its bond graph model. Each mode defines a physically valid energy model whose bond graph elements can be mapped back to components and mechanisms of the physical system. Note that the system modes and transitions are dynamically generated, and do not have to be pre-enumerated.

4.1 The Modeling Language

The modeling language has two components:

1. traditional energy-related bond graphs[10, 19], and
2. finite state automata for modeling controlled junctions.

Bond graphs are *well-grounded* in physical reality[2, 19, 23] and finite state automata represent a formal methodology for modeling time-varying discrete systems. The combination of these two models which interact through the controlled and regular junctions define overall system behavior.

4.2 Controlled Junctions

When active (*on* state), controlled junctions behave like normal 0- or 1-junctions. Note that for a 0-junction the effort value associated with all connected bonds are equal and the flow values sum to 0. For a 1-junction the flow values are equal and the effort values sum to 0. A deactivated (*off*) 0-controlled junction forces the effort value at all connected bonds to become 0, implying that there is no energy transfer across this junction. Similarly, a deactivated 1-junction forces flows to become 0. In both cases, the

[2]Traditional bond graph junctions are always in the *on* state.

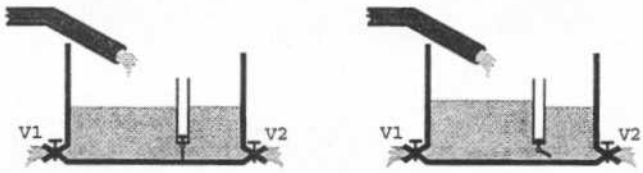

Figure 2: **The bi-tank system with memory**

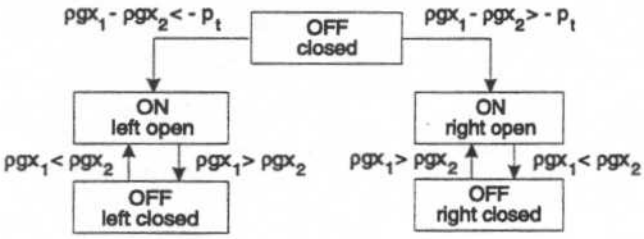

Figure 3: **CSPEC State Transition Diagram: Complex bi-tank system**

controlled junction exhibits ideal switch behavior, and modeling discontinuous behavior in this way is consistent with bond graph theory[18, 23]. Deactivating controlled junctions can affect the behaviors at adjoining junctions, and, therefore, the causal relations among system variables.

Controlled junctions are marked with subscripts (e.g., 1_1, 0_2) and the corresponding control logic is depicted as dark circles in the bond graph representation (Fig. 4). Their input, effort and flow energy variables tapped from 0- and 1-junctions, respectively are shown as arrows. Their output, a single control signal (shown as a dotted line), sets the associated controlled junction to *on* or *off*. The junctions define the *interactions* between the energy-flow and signal-flow models of the system.

The control logic for determining the *on* and *off* state for a controlled junction is implemented as a finite state automata, and represented as a state transition graph or table. The two representations are logically equivalent. We call this *control specification* function *CSPEC*[8] which is actually a finite state machine whose internal communication is asynchronous and overwriting. Each controlled junction has one associated state transition diagram which generates an *on* or *off* signal as output. However, the state transition diagram may have several internal states that map onto the *on* and *off* states of the controlled junction.

The use of controlled junctions is illustrated in the complex bi-tank system shown in Fig. 2. To avoid unnecessary complexity, the effect of the left compartment overflowing is not considered in the discussion that follows. The latch between the tanks can go either way from its initial upright position when the *pressure differential* between the tanks exceeds a threshold value p_t. An open switch can close, but it does not open in the opposite direction. For example, if the pressure differential caused the switch to open to the right, it remains open in that direction but closes if

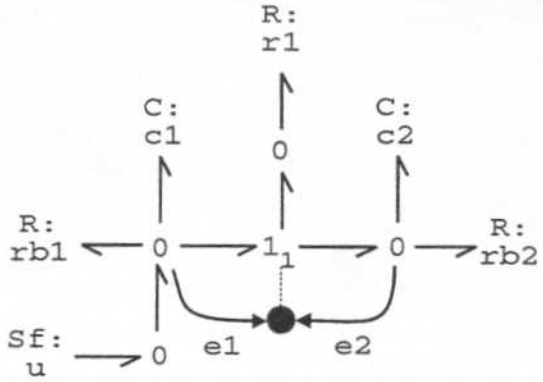

Figure 4: **Bond Graph Model: Complex bi-tank system**

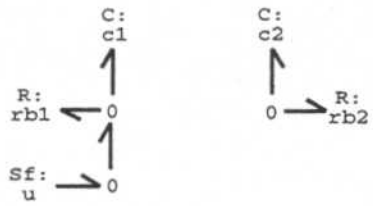

Figure 5: **Bond Graph Model: Initial System State**

the pressure differential becomes 0. If the pressure differential reverses, the latch does not open to the left. This behavior can only be modeled using sequential logic, because the *open state* of the switch determines all future behavior. Fig. 3 illustrates the corresponding state-transition diagram. x_1 and x_2 represent the height of the liquid column in the left and right compartments, respectively, ρ is the density of the fluid, and g is the gravitational constant.

The bond graph model of the energy-related part of the system is shown in Fig. 4. c_1 and c_2 are the capacitances (storage capacity) of the left and right tank, respectively. The two outflow valves are modeled as fluid resistances, with values rb_1 and rb_2, respectively. The flow source into the left compartment, S_f has a constant inflow rate u. The valve separating the two compartments operates in one of two modes: (i) when open, it allows resistive flow (resistance value r_1) through it, and (ii) when closed it disconnects the two compartments.

The CSPEC representation for the valve modeled as a controlled junction is shown in Fig. 3. If the latch is closed, i.e., there is no flow of liquid, the 1_1 junction is deactivated, and the flow at this junction is now 0. Energy transfer across this junction becomes 0, which implies that the bonds incident on the junction can be eliminated in this system mode. The net result is that the two compartments of the tank become independent subsystems (Fig. 5) thus producing a seamless implementation of the mode-switching process.

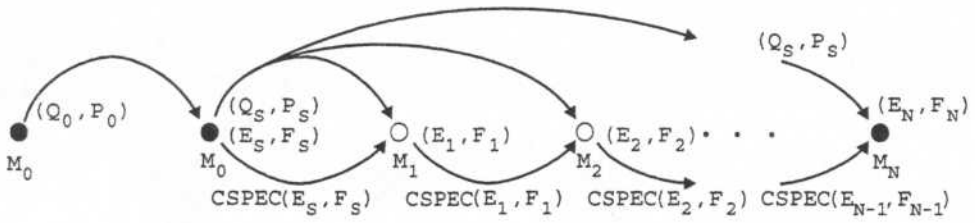

Figure 6: **A Sequence of Mythical Transitions**

4.3 Mode Switching in the Hybrid Model

Nishida and Doshita[16] discuss how a single discontinuous change can propagate in a system causing a complex chain of (possibly discontinuous) events. Model abstractions make it difficult to explain these changes in stepwise causal terms, and this problem was addressed by taking the system through a sequence of *mythical*[3] (i.e., instantaneous) instants where a sequence of component mode changes occur (the direct method). A system with k controlled junctions can theoretically assume 2^k possible configurations, each of which can be described as a mode of the system.

However, not all of these modes may be physically realizable because modes

1. may not be *reachable*, and
2. mode may be *transient*.

In the hybrid modeling scheme, a non reachable mode corresponds to a set of on-off values for the controlled junctions in the system that cannot be derived from a given initial state of the system. A mode is transient when it instantaneously changes to another. A sequence of transient modes can occur during behavior generation, and mode switching ends when a non-transient mode is reached. In simulation, this switching process happens instantaneously and so the intermediate modes are not considered to be real behavior states that the system goes through.

As discussed earlier, mode switching is initiated when the *CSPEC* function of a particular controlled junction indicates a change of state for that junction. This results in a new bond graph configuration, and effort and flow values for the bonds have to be computed for this configuration. This is done by invoking the simulator, *SIMUL*, using an initial state defined by the energy distribution variables just before the controlled junction switched. The new effort and flow values may trigger further switching, and, if that happens, the current configuration of the system becomes a mythical state.

The algorithm for mode switching through mythical states, *MMA*, is based on the premise that all switching is ideal and represented as signal flows. Therefore, not only is energy conserved during the switching process, but no instantaneous changes in effort and flow values can occur during changes through the mythical states of the system. Therefore, when multiple junction switches are handled in sequence, the effort and flow values at each of the intermediate modes, are computed from the energy distribution values at the start of the switching process, and *not* the distribution values computed for the last mythical state. This computation process is illustrated in Fig. 6. Note that the intermediate modes, M_1, M_2, etc., are mythical modes (depicted by open circles) that facilitate determination of the actual system mode, M_N. From a physical, energy-related viewpoint, the system directly transitions from M_0 to M_N, maintaining continuity and conservation of energy. Only the signal-flow model exhibits discontinuous behavior, which is attributable to the abstract concept of switching. In real systems, the act of switching may incorporate energy loss, and this can be handled by incorporating resistive elements in the bond graph model.

The Mythical Mode Algorithm *MMA* is summarized below. E_i and F_i are the set of effort and flow values, and Q_i and P_i are the corresponding energy values (generalized *displacement* and *momentum*, respectively) in the system at time step i ($i \geq 1$).

1. Calculate the energy values (Q_s, P_s) and signal values (E_s, F_s) for bond graph model M_0 using (Q_0, P_0), values at the previous simulation step as initial values.
2. Use *CSPEC* to infer a possible new mode given (E_s, F_s).
3. If one or more controlled junctions switch states then:
   (a) Derive the bond graph for this configuration.
   (b) Assign causal strokes using the SCAP algorithm[19].
   (c) Calculate the signals (E_i, F_i) for the new mode, M_i, based on the initial values (Q_s, P_s).
   (d) Use *CSPEC* again to infer a possible new mode based on (E_i, F_i) for the new mode, M_i.
   (e) Repeat step 3 till no more mode changes occur.
4. Establish the final mode, M_N, as the new system configuration.

5 The Simulation System

The simulation algorithm has three key modules:

[3]The concept of mythical time was first introduced by de Kleer and Brown[5].

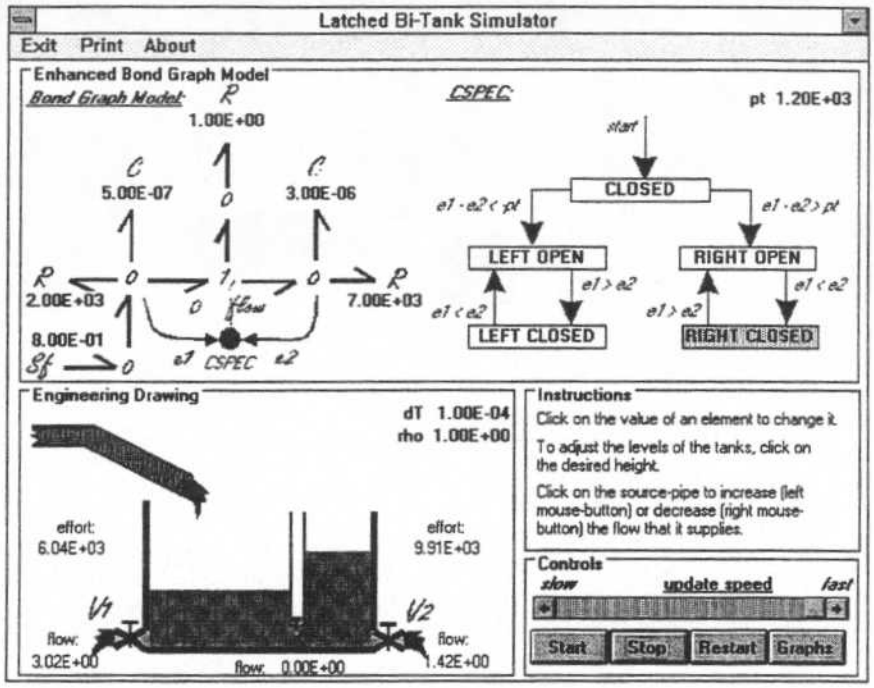

Figure 7: **The Simulator Interface**

1. *SIMUL*, the continuous system simulator,
2. *CSPEC*, that determines switching of controlled junctions, and
3. *MMA*, the mythical mode algorithm.

Simulation models of two systems, (i) a latched bi-tank system (Fig. 2), and (ii) a diode-inductor circuit (Fig. 9) were developed under Microsoft Windows using Visual Basic 3.0 Professional Edition[13]. The system models are incorporated as a matrix of equations which are derived from the bond graph model manually and inverted using Mathematica. Note that this derivation process is already fully automated in systems like CAMAS[3]. Integration is implemented as a zero-order approximation and careful selection of the time step produced good results inspite of this simplifying approximation.

5.1 Simulating the latched bi-tank

Fig. 7 shows the latched bi-tank simulator window. The mode switch implementation sets the flow in the controlled junction to 0, which eliminates this variable from the set of equations in the active mode. The simulation uses a color code to depict the current state of the controlled junction (*green* implies the switch is *on* and *red* implies that it is *off*).

Simulation of the system using the parameters displayed in Fig. 7 produces the behavior description shown in Fig. 8. At the start, the latch is closed and there is no flow between the compartments. This represents the closed state in the CSPEC (Fig. 3). The resulting bond graph consists of two independent containers (Fig. 5). Both of the containers are connected to dissipating elements (valves) and the left container has an input flow source.

When the level of fluid in the left container becomes high enough, the threshold pressure (p_t) is exceeded and the latch opens to the right. The system moves into its *right open* state and the 1_1 junction now behaves as a regular 1-junction. Because of the low resistance connection (the parameters are defined in such a way that the time constant ($R \cdot C = 5 \times 10^{-7}$) of the flow through the valve is small compared to the time step chosen (10^{-4})), the levels of fluid in the two compartments will almost instantaneously move to the same value. Fig. 8 shows this initial exponential increase and sharp decrease of fluid-level of the left container as a little hub. The levels of fluid in both of the containers now move to an equilibrium steady-state. Being a first order system (the two energy storing elements are dependent), there is no overshoot at the equilibrium level and the latch does not oscillate.

Next, the flow supplied by the source S_f is increased to 10 times its initial value. The latch is already in its *right-open* state, therefore, the system exhibits a first order exponential behavior towards a new equilibrium level for both containers. Then the input flow is reduced to half its current value, and the level of fluid in the left compartment falls much faster than the right compartment because of its smaller time constant (10^{-3} for left compartment and 21×10^{-3} for the right compartment). As soon as the pressure in the right compartment exceeds that of the left, the valve closes in its *right-closed* mode, i.e., it cannot open to the left. The right compartment now achieves a new equilibrium level. The resulting bond graph is that of the two independent compartments, i.e., the initial mode of the system (Fig. 5). When the level of fluid in the right compartment falls below the level in the left compartment, the latch again opens up to the right.

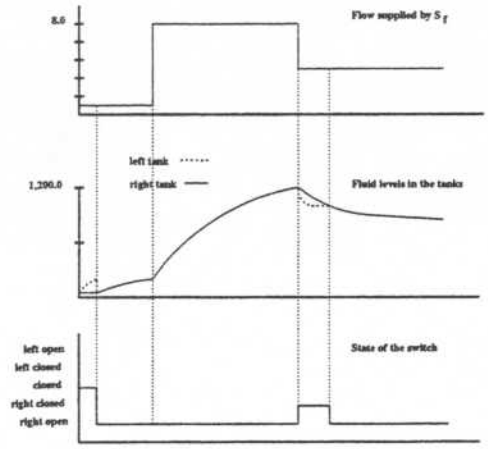

Figure 8: **Simulation Results: Bi-tank System**

The levels of fluid in the left and right compartments then achieve a new equilibrium level together.

5.2 Simulating the Diode-Inductor System

The diode-inductor example (Fig. 9)[12] demonstrates the effectiveness of the hybrid modeling technique when multiple controlled junctions cause the simulation to go through a set of mythical states during mode transition. In this circuit, a voltage source drives an inductor when an ideal switch is on. The other components, the resistor, diode, and inductor are also considered to be ideal. Turning off the switch abruptly interrupts the energy supply. Without the diode this interruption would cause a large voltage drop over the inductor (such circuits are used to induce the high voltages needed to drive spark plugs of automobiles[17]).

The bond graph model of the system is shown in Fig. 9. Both the switch and the diode are modeled as automata that control the state of 1–junctions. Note that the switch is externally controlled. The *CSPEC* definitions for these controlled junctions are specified as follows.

Junction 1_1 – the switch.
switch *on* $\rightarrow$ junction *on*.
switch *off* $\rightarrow$ junction *off*.

Junction 1_2 – the diode.
$e_4 < -V_{diode}(0.7V) \rightarrow$ junction *on*.
$f_5 \geq 0 \rightarrow$ junction *off*.

Note that the CSPEC definitions are local, therefore, no global control structure needs to be known beforehand. Given an initial state, all system modes that are reachable from this state will be generated dynamically by simulation. The four possible modes of the system are:

1. 00 - switch open/diode not conducting
2. 01 - switch open/diode conducting
3. 10 - switch closed/diode not conducting
4. 11 - switch closed/diode conducting

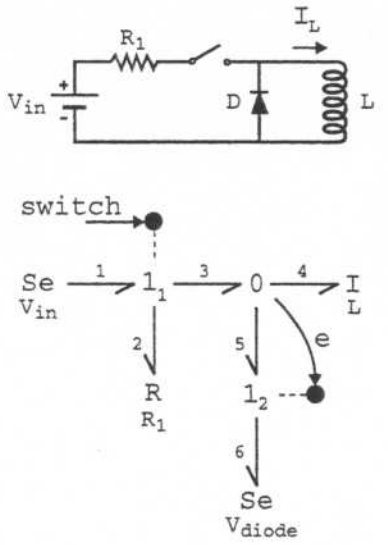

Figure 9: **The Diode Inductor Circuit and its Bond Graph**

CSPEC and *MMA* are applied to effect mode switching in the system. Initially the switch is open, the inductor has no stored energy, the diode is not conducting, and the system is in mode 00. At time step 10 the switch is closed, the system moves into the 10 mode, and all effort and flow values are recalculated for the new configuration. No further mode transition occurs, and the inductor charges as shown. At time step 100, the switch is reopened. Step 3(c) of *MMA* recomputes all efforts and flows for the new mode 00. The inductor has built up a flux (i.e., energy p), therefore, disconnecting it induces a large negative voltage ($-\infty$ in the limit). This causes the diode to come on instantaneously, so mode 00 becomes a mythical state, and the system switches to mode 01 where the effort and flow values are recomputed based on the initial flux of the inductor. Again the new values do not cause another mode change so *SIMUL* is active and Fig. 10 shows that the inductor discharges through the diode. Note that the signal values computed for the mythical mode are only used for switching (with no loss of energy), therefore, the infinite negative voltage is never reached. If it were, the stored energy of the inductor would be released instantaneously, producing an incorrect energy balance in the overall system.

With time, the flow (current) through the inductor decreases to zero. At time stamp 315 the current value is ≤ 0. Therefore, the current through the diode is ≥ 0 (opposite sign) and this causes a final transition: the system again switches to mode 00, and since there is no stored energy in the system, this becomes the final state. The spike observed in simulation is an artifact caused by the time step used for simulation. In this simulation, between the two time steps the current went from a small positive to a small negative value before the transition took place. Thus when the system transitioned to the 00 state the small current in the inductor went to 0 instantaneously, which resulted in the spike shown.

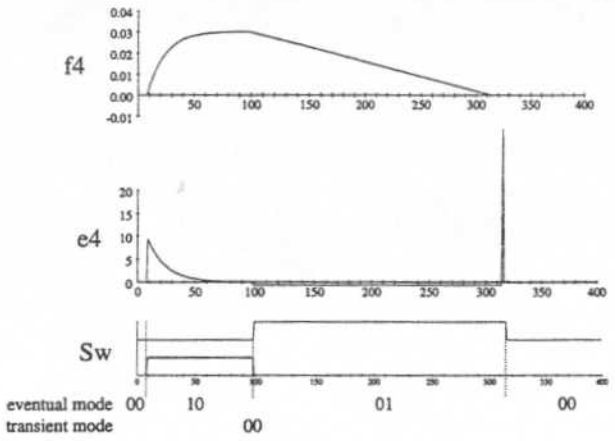

Figure 10: **Simulation Results: Diode Inductor with parameter values** $V_{in} = 10V, R_1 = 330\Omega, L = 5mH$.

This result conforms with the results presented in [12] with a more formal and general approach (*MMA*) in the framework of our hybrid modeling scheme.

6 Conclusions

This paper has discussed a comprehensive modeling methodology for physical systems that exhibit multiple discontinuous behavior modes. Our primary contribution is the introduction of systematic and uniform mode-switching methods into a formal compositional modeling scheme based on the bond graph methodology. The goal, to provide a modeling language that ensures the development of consistent, rigorous and complete models, is achieved by combining: (i) the bond graph scheme to model the energy-related aspects of system behavior, and (ii) finite-state automata to model signal-flows that cause configuration changes in the bond graph model to produce discontinuous behavior. Interaction between the two components of the model are restricted to signals that act on so-called *controlled junctions*. These signals are an integral part of the bond graph language.

Furthermore, the *MMA* algorithm developed for switching system modes through multiple switching states

- ensures energy conservation within the system, and
- disallows instantaneous effort and flow changes when the system changes from one mythical mode to another.

Therefore, the signal flow model does not directly cause any energy flows or changes; it handles mode changes. Energy exchange between components in a mode or between modes (only the real ones) of the system are handled by the energetically correct bond graph model, thus ensuring no spurious behaviors are generated.

The strict definition of the interaction between the energy-flow and signal-flow components of the modeling methodology is of paramount importance in generating valid physical models. The approach presented supports modeling discontinuities caused by (i) abrupt switching, such as in idealized valves and diodes, (ii) mode switching caused by parameter value changes, such as the change from laminar to turbulent flow in a pipe when the Reynolds number goes above a threshold value, and (iii) configuration switching caused by changes in sub-system models.

The advantages of the described method are twofold:

1. Focusing on the energy model instead of the external control models (as is done in QSIM and CC) allows for dynamic model composition.
2. The inherent integrity checks enforce physically correct models.

Dynamic model composition is extremely important for complex systems that include a large number of discontinuous components. As discussed earlier, such systems can exhibit an exponential number of modes, therefore, pre-enumerating modes is not a feasible modeling approach. Moreover, a large number of these modes are not physically achievable, but that cannot always be determined before hand. The inherent integrity checks ensure the physical correctness of a model and aid the modeler in building a correct model. Furthermore, the conservation and continuity constraints help reduce the number of spurious behaviors that are generated.

As a next step, this approach needs to be applied to comprehensive modeling and analysis within a compositional modeling framework. Our long-term goal is to employ these methodologies in building robust models for automated diagnosis and design of complex, engineering systems.

References

[1] G. Biswas, S. Manganaris and X. Yu, "Extending Component Connection Modeling for Analyzing Complex Physical Systems," *IEEE Expert*, vol. 8, no. 1, pp. 48-57, 1993.

[2] G. Biswas, and X. Yu, "A Formal Modeling Scheme for Continuous Systems: Focus on Diagnosis," *Proc. IJCAI-93*, Chambery, France, pp. 1474-1479, August 1993.

[3] J.F. Broenink, Computer Aided Physical Modeling and Simulation: Bond-Graph Approach, *Ph.D. thesis*, University of Twente, Febodruk, Enschede, The Netherlands, 1990.

[4] J.F. Broenink and K.C.J. Wijbrans, "Describing Discontinuities in Bond Graphs," *Proc. Intl. Conf. on Bond Graph Modeling*, pp. 120-125, San Diego, CA, 1993.

[5] J. deKleer and J.S. Brown, "A Qualitative Physics Based on Confluences", *AI Journal*, vol. 24, pp. 7-83, 1984.

[6] B. Falkenhainer and K. Forbus, "Compositional Modeling: Finding the right model for the job," *AI Journal*, vol. 51, pp. 95-143, 1991.

[7] K.D. Forbus, "The Qualitative Process Engine," in *Readings in Qualitative Reasoning about Physical Systems*, D. Weld and J. deKleer, eds., pp. 220-235, Morgan Kaufmann, San Mateo, CA, 1990.

[8] D.J. Hatley, and I.A. Pirbhai, Strategies for Real-Time System Specification, Dorset House Publishing Co., New York, N.Y., 1988.

[9] D.C. Karnopp, "General method for including rapidly switching devices in dynamic simulation system models," *Trans. Society for Computer Simulation*, col. 2, pp. 155-168, 1985.

[10] D.C. Karnopp, D.L. Margolis, and R.C. Rosenberg, Systems Dynamics: A Unified Approach, 2nd edition, John Wiley, New York, 1990.

[11] B. Kuipers, "Qualitative Simulation," *Artificial Intelligence*, vol. 29, pp. 289-338, 1986.

[12] F. Lorenz and H. Haffaf, "Combinations of Discontinuities in Bond Graphs," *Proc. ICBGM '95*, vol. 27, no. 1, pp. 56-64, Society for Computer Simulation, San Diego, CA, 1995.

[13] Microsoft Corporation, Visual Basic, Version 3.0 Professional Edition, One Microsoft Way, Redmond, WA, 98052-6399, 1993.

[14] P.J. Mosterman and G. Biswas, "Behavior Generation using Model Switching: A Hybrid Bond Graph Modeling Technique," *Proc. ICBGM '95*, vol. 27, no. 1, pp. 177-182, Society for Computer Simulation, San Diego, CA, 1995.

[15] P. Nayak, L. Joscowicz and S. Addanki, "Automated Model Selection using Context-Dependent Behaviors," *Proc. AAAI-91*, AAAI/MIT Press, Menlo Park, CA, pp. 710-716, 1991.

[16] T. Nishida, and S. Doshita, "Reasoning about Discontinuous Change,", *Proc. AAAI-87*, pp. 643-648, Seattle, WA, 1987.

[17] E.F. Obert, Internal Combustion Engines, 2nd ed., The Haddon Craftsmen, Inc., Scranton, PA, 1950.

[18] H.M. Paynter, Analysis and Design of Engineering Systems, The M.I.T. Press, Cambridge, MA, 1961.

[19] R. Rosenberg, and D. Karnopp, Introduction to Physical System Dynamics, McGraw Hill, New York, NY, 1983.

[20] J.E. Strömberg, J. Top and U. Söderman, "Variable causality in bond graphs caused by discrete effects," *Proc. Intl. Conf. on Bond Graph Modeling*, pp. 115-119, San Diego, CA, 1993.

[21] J.E. Strömberg, U. Söderman and J. Top, "Bond graph supported estimation of discretely changing parameters," *Proc. Intl. Conf. on Bond Graph Modeling*, pp. 47-52, San Diego, CA, 1993.

[22] J.L. Top, J.M. Akkermans, and P.C. Breedveld, "Qualitative reasoning about physical systems: an artificial intelligence approach," *Journal of the Franklin Institute*, vol. 328, no. 5/6, pp. 1047-1065, 1991.

[23] J.L. Top, Conceptual Modeling of Physical Systems, *Ph.D. thesis*, University of Twente, CIP-DATA Koninklijke Bibliotheek, Den Haag, 1993.

[24] S.M. White and J.Z. Lavi, "Embedded Computer System Requirements Workshop," *Computer*, April 1985.