Using Modelica Models for Qualitative Reasoning

Matthew Klenk and **Johan de Kleer** and **Daniel G. Bobrow** and **Bill Janssen**
Palo Alto Research Center
3333 Coyote Hill Rd
Palo Alto, CA, 94303
klenk,dekleer,bobrow,janssen@parc.com

Abstract

Applications of qualitative reasoning to engineering design face a knowledge acquisition challenge. Designers are not fluent in qualitative modeling languages and techniques. To overcome this barrier, we perform qualitative simulation using models solely written in Modelica, a popular language for modeling hybrid systems. We define the relationship between the results of the Modelica and qualitative simulations and describe how qualitative simulation from numerical models can assist designers. We discuss challenges and solutions for abstracting equations into constraints, determining initial conditions, continuous behavior, and discrete events. In particular, we identify three places where additional constraints should be derived from Modelica equations, and describe how we bridge the gaps between Modelica and existing qualitative simulation work on discrete behavior. Our system has been integrated with the OpenModelica[1] tool and we discuss its potential design applications.

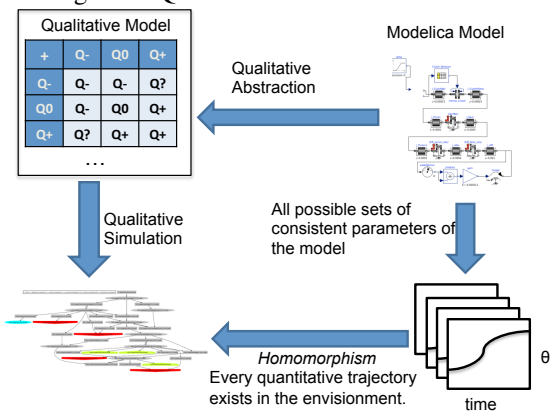
1 Introduction

Over the last half century, industry and academic professionals have developed a plethora of modeling languages and tools to analyze designs. Centered around a particular set of analyses, each tool requires the designer to specify the problem in a particular way and interpret the analysis results with respect to their design question. The languages and analyses of qualitative reasoning have not made inroads into the engineer's practice. Consequently, with a few notable exceptions (e.g., [Struss and Price, 2004]), qualitative reasoning has not been applied in industrial design settings.

We seek to overcome this barrier for the DARPA Adaptive Vehicle Make program[2], which seeks dramatically reduce the cost and time required to design, verify and manufacture complex cyber-physical systems. Our role is to integrate qualitative reasoning into the CyPhy toolchain [Simko *et al.*, 2012] for use by designers and system engineers. The CyPhy toolchain uses Modelica [Fritzson, 2004] to model hybrid systems. Therefore, to enable designers to use qualitative reasoning techniques, it is necessary to automatically translate Modelica models into qualitative models. In this paper, we discuss challenges and solutions to automatically translating Modelica into models for qualitative reasoning [Kuipers, 1994][de Kleer and Brown, 1984].

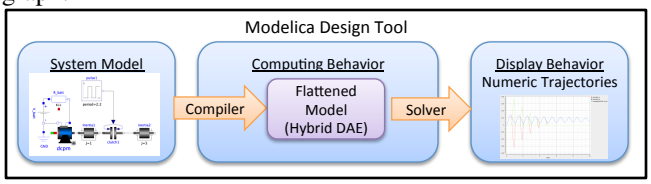
We believe [Weld and de Kleer, 1989] qualitative reasoning helps engineers set up quantitative analyses and interpret the results. Therefore, automating these tasks should free the designer to consider more challenging design problems. Using only qualitative simulation, the following questions can be answered automatically: (1) If a simulation fails to meet the requirements, should the engineer change the parameters or the topology? (2) Does the simulation include numerical instabilities or missed events? In previous work on battlespace planning [Hinrichs *et al.*, 2011], we showed how qualitative simulation could guide probabilistic analysis enabling the guided approach to analyze models in a fraction of the time as traditional methods. In addition to automatically interpreting simulation results and guiding quantitative analyses, qualitative models provide an alternative framework to probability distributions for capturing the inherent uncertainty of modeling that can be reasoned over symbolically. These benefits can be realized by performing qualitative reasoning from quantitative models.

Figure 1: Qualitative simulation semantics



[1] www.openmodelica.org
[2] http://en.wikipedia.org/wiki/Adaptive_Vehicle_Make

Figure 2: The Modelica analysis process begins with the user creating hierarchical numerical Modelica model, frequently with the use of a GUI (such as OMEdit). Next the analysis is performed by compiling the model into a hybrid differential and algebraic representation which is then solved.. The result is a table of numeric values for every model variable over time, which is typically presented to the user in the form of a graph.



When performing model translation, it is necessary to define the relationship between the simulation results. We define that a qualitative abstraction maintains the following relationship between Modelica and qualitative simulations (shown in Figure 1). Given a Modelica model (upper right), we create qualitative model of constraints (upper left) from which we perform a qualitative simulation to produces an *envisionment* (lower left). The meaning of this envisionment is that every consistent assignment of parameters in the Modelica model will result in a quantitative simulation (lower right), and each correct quantitative simulation will correspond to a trajectory in the envisionment. The translation is incorrect if there exists a set of valid quantitative parameters that generate a quantitative simulation that does not correspond to any trajectory in the envisionment. The techniques described in this paper are correct under this definition.

Directly translating the Modelica equations into qualitative constraints results in an envisionment what numerous unrealizable trajectories (i.e., there is no corresponding set of numeric parameter which generates a corresponding quantitative simulation). While unrealizable states and trajectories are an unavoidable problem for qualitative simulation[Struss, 1988], by making the implicit information of equations explicit, we reduce the set of unrealizable trajectories improving the utility of the resulting envisionment. When creating the qualitative model, we expand the set of constraints from those that appear explicitly in the hybrid differential and algebraic equations (hybrid-DAE) with the following three types of quantitatively redundant relations: (1) the continuity and compatibility equations from system dynamics, (2) equalities between higher-order derivatives, and (3) landmark ordering. In addition to continuous behavior, Modelica allows for discrete behaviors. Traditional approaches to qualitative modeling of discrete behavior require additional modeling knowledge not directly accessible in the hybrid-DAE. We describe an approach to overcome this lack of knowledge and show that it respects the semantics in Figure 1. Furthermore, we introduce *pseudo state variables*, qualitative variables that maintain their values through discrete transitions, to reduce unrealizable trajectories.

2 Dynamics Modeling in Modelica

Figure 2 illustrates the modeling and simulation process used to analyze Modelica models. The process begins with the user creating a hierarchical Modelica model, frequently with the use of a graphical user interface. Next, the simulation is performed by compiling the model into a flattened Modelica model that is represented as a set of hybrid differential and algebraic equations (DAE). Given a DAE, an equation solver (e.g., DASSL [Petzold, 1982]) produces a sequence of numeric values for every model variable, which is typically presented to the user in the form of a graph.

We previously identified the Hybrid-DAE as the correct point to abstract the Modelica model into a qualitative model [Klenk *et al.*, 2012]. Abstracting this representation has the advantages of allowing designers to use the Modelica model construction language and models from the Modelica Standard Library in creating their designs.

2.1 Hybrid Differential Algebraic Equations

In Modelica, the continuous-time behavior is governed by differential (1) and algebraic (2) equations with state variables, x, algebraic variables, y, and inputs, u.

$$\dot{x}(t) = f(x(t), y(t), u(t)) \quad (1)$$

$$y(t) = g(x(t), u(t)) \quad (2)$$

Discrete changes occur at events specified by conditions that change in truth value. Conditions on continuous variables are analogous to landmarks in qualitative modeling. Events result in new values for discrete-time variables and a new set of equations to govern the continuous dynamics. In the following section, we describe how these Modelica equations are abstracted into constraints and used for qualitative simulation.

3 Qualitative Simulation with Modelica Models

Our constraint-based qualitative simulator draws on the ideas from established methods[de Kleer and Brown, 1984][Kuipers, 1994]. To perform qualitative simulation with Modelica models, it is necessary perform the following tasks: create the qualitative model, initialize the qualitative state, and simulate the continuous and discrete behavior. In the following sections, we describe each in turn.

3.1 Abstracting the Hybrid-DAE

While the qualitative research community has established methods for generating constraints from differential equations [Kuipers, 1994], the hybrid-DAE produced by the Modelica flattening process requires additional techniques. The established method for abstracting equations is as follows. If the equation consists of three or fewer variables, we create the corresponding qualitative constraint directly. For equations with more than three variables, such as the ramp torque source equation in Figure 3, we replace pairs of variables with *dummy* variables representing their combination and generate the corresponding constraint. We do this recursively until the original equation consists of only three variables. The ramp

torque source equation also includes conditions. Therefore, we use conditional constraints to govern the appropriate equations. For example, $dummy3$ is mentioned in two conditional constraints to either be equal to $ramp1.height$ or $dummy6$ depending on the value of the condition, $dummy4$.

Figure 3: Conditional expressions in Modelica equations are transformed into conditional constraints that set the values of dummy variables.

Modelica equation:

$$\begin{aligned}&torque1.tau = ramp1.offset + \\ &\quad \text{if } time < ramp1.startTime \\ &\quad\quad \text{then } 0.0 \\ &\quad\quad \text{else if } time < ramp1.startTime + ramp1.duration \\ &\quad\quad\quad \text{then } (time - ramp1.startTime) \times \tfrac{ramp1.height}{ramp1.duration} \\ &\quad\quad\quad \text{else } ramp1.height;\end{aligned}$$

Qualitative constraints:

$$\begin{aligned}&torque1.tau = ramp1.offset + dummy1 \\ &\quad dummy1 = \begin{cases} 0.0 & \text{if } dummy2 < 0 \\ dummy3 & \text{if } dummy2 \not< 0 \end{cases} \\ &\quad dummy2 = time - ramp1.startTime \\ &\quad dummy3 = \begin{cases} dummy6 & \text{if } dummy4 < 0 \\ ramp1.height & \text{if } dummy4 \not< 0 \end{cases} \\ &\quad dummy4 = time - dummy5 \\ &\quad dummy5 = ramp1.startTime + ramp1.duration \\ &\quad dummy6 = dummy7 \times dummy8 \\ &\quad dummy7 = time - ramp1.startTime \\ &\quad dummy8 = \frac{ramp1.height}{ramp1.duration}\end{aligned}$$

In the next three sections, we describe additional constraints that we add by analyzing the hybrid-DAE.

Continuity and Compatibility Conditions

System dynamics theorems specify the minimal set equations necessary to enforce the continuity and compatibility conditions (e.g., Kirchoff's Voltage and Current Laws), and this minimal set of equations is contained in the hybrid-DAE. These theorems do not hold for qualitative arithmetic [de Kleer and Brown, 1984]. Therefore, it is necessary to compute the quantitatively redundant node and loop constraints by combining continuity and the compatibility equations. Consider the two Modelica equations shown in Figure 4. By adding these equations together, we derive the qualitative constraint that $g1.p.i = 0.0$, thereby reducing the number of unrealizable transitions in the resulting simulation.

Higher-order Derivative Equalities

Another technique for generating additional constraints concerns equalities between variables with explicit derivatives.

Figure 4: Due to ambiguities in qualitative algebra, quantitative equations for system dynamics must be symbolically combined to create additional qualitative constraints.

Modelica equations:

$$\begin{aligned}r2.i + c1.i + cv1.i &= 0.0; \\ g1.p.i - c1.i - r2.i - cv1.i &= 0.0;\end{aligned}$$

Additional qualitative constraint:

$$g1.p.i = 0.0$$

Consider the equations in Figure 5 modeling a brake attached to a flywheel. While their positions and speeds are equal, the hybrid-DAE represents this in three equations. Converting only those equations to qualitative constraints fails to capture the equality between the speeds. Therefore, for any two variables with explicit derivatives that are equal, we add equality constraints between their derivatives.

Figure 5: If two variables are equal, then their explicit derivatives must also be equal.

Modelica equations:

$$\begin{aligned}brake1.phi &= flywheel1.phi; \\ brake1.w &= der(brake1.phi); \\ flywheel1.w &= der(flywheel1.phi);\end{aligned}$$

Additional qualitative constraint:

$$brake1.w = flywheel1.w$$

Partially Ordered Landmarks

The Modelica hybrid-DAE does not explicitly create quantity spaces. Instead, landmark variables are the result of event conditions. Consider the equation in Figure 6. The equation states that the $startForward$ variable is $true$ when the brake was stuck and the variable sa, a path parameter for the torque applied to the brake, is greater than the maximum torque of the brake, or the brake had begun moving and sa is greater than the sliding friction of the brake. Implicit in this equation are two landmarks for the variable sa ($tau0\_max$ and $tau0$). $landmark1$ and $landmark2$ are created by the process described in Section 3.1. The third constraint provides an ordering for the two landmarks and can be generated either by the parameter values in the hybrid-DAE or passed as arguments to the qualitative simulation system.

3.2 Initialization

During initialization and every discrete event, Modelica determines the values of the model variables, the derivative for each continuous variable, and the `pre` value (i.e., the value in the left limit before the initial instant) for each discrete variable. Initialization is provided with a set of initial values, which, for continuous variables, are assigned to the initial state, and, for discrete variables, are assigned `pre` values.

Figure 6: A Modelica conditional equation specifying two landmarks for the $brake1.sa$ quantity space and the corresponding additional qualitative constraints defining these landmarks and their order.

Modelica equation:

```
brake1.startForward =
 pre(brake1.mode) == Stuck
   and (brake1.sa > brake1.tau0_max or
          pre(brake1.startForward)
          and brake1.sa > brake1.tau0)
     or initial() and brake1.w > 0.0;
```

Qualitative landmark variables and constraints:

$$landmark1 = brake1.sa - brake1.tau0\_max$$
$$landmark2 = brake1.sa - brake1.tau0$$
$$Q+ = landmark1 - landmark2$$

Given these values, Modelica tools solve the equation system to determine a consistent set of values for the other variables. When the user-provided initial values do not uniquely determine the values for the other variables in the system, Modelica tools search for a consistent initial state using defaults (e.g., 0 for continuous variables) and user-provided suggested values. The initial quantitative state is dependent on the numeric values given to parameters. Qualitative simulation explores the partially ordered parameter space. Therefore, it is necessary to determine the set of initial qualitative states using the qualitative constraints.

To satisfy the semantics in Figure 1, our qualitative simulation tool must generate the corresponding qualitative initial state. If there are variables that are not determined by the constraints and initial values, we replace the initial instant with a set of initial instants by performing constraint satisfaction to identify all possible qualitative values for the unknown variables that are consistent with the constraints. Our approach generates an initial state for every consistent set of qualitative values. Because the quantitative solution generated by Modelica is consistent with the equation set, and the qualitative constraints are an abstraction of this equation set, then the corresponding qualitative values for these variables must satisfy the qualitative constraints. Therefore, our generated set of initial instants necessarily includes the single initial state selected by the Modelica, thereby aligning our approach with Modelica.

3.3 Continuous Integration

In constructing the hybrid-DAE, Modelica compilers perform index reduction to arrive at an index-1 DAE. Thus, every continuous-time variable is differentiable with respect to time, and therefore, equivalent to the *reasonable functions* [Kuipers, 1994]. Therefore, by the *guaranteed coverage theorem* [Kuipers, 1994], given a sound abstraction of equations into constraints, the correct continuous behavior of the Modelica model must appear in the envisionment.

The alignment of Modelica simulation and qualitative behaviors has three features of note. First, qualitative state changes in the envisionment occur when any variable or its derivative crosses zero. While Modelica varies its integration step size when searching for *events* (i.e., changes in the conditions of equations), zero crossings that do not effect the dynamics of the system are ignored. Therefore, if the integration step size is too large, it may not be possible to know if a variable crossed zero at the same instant as another. Second, qualitative simulation produces unrealizable transitions and states. It is an active area of qualitative simulation research to determine under what circumstances qualitative constraints simulation do not result in such behaviors [Sachenbacher and Struss, 2005][Yılmaz and Say, 2006] . Third, Modelica simulators are susceptibility to numerical integration errors. These incorrect simulations have no defined relationship with the envisionment. Consequently, when the qualitative abstraction of a Modelica simulation does not occur in the envisionment, we can signal a numeric integration error.

3.4 Discrete Changes

The challenge in aligning Modelica's discrete behavior [Otter *et al.*, 1999] with qualitative simulation lies in the lack of agreement among different qualitative simulation methods. After a short description of Modelica's discrete-time semantics, we discuss the different approaches from qualitative reasoning and illustrate our approach with a concrete example aligning the simulation results.

Discrete-time Behavior in Modelica

In Modelica, discrete changes occur at *events*. These are changes in the conditions of equations resulting in new equations governing the behavior of continuous-time variables and/or new values for discrete-time variables. An event occurs when a condition changes from `false` to `true`. Events in Modelica are governed by the *synchronous data-flow principle* which states the following. First, events take no time. Second, the number of equations equal the number of variables. Third, all variables maintain their actual values until these values have been explicitly changed. Fourth, at every point in time, the solution to the set of *active* equations must be satisfied concurrently. Events may change the values of discrete variables or the set of active equations. Events may cause other events. While sequential, the entire event sequence has no duration.

Previous Approaches from Qualitative Simulation

While numerous approaches to qualitative simulation have addressed issues surrounding discrete events, they all require additional modeling than what is contained in Modelica's hybrid-DAE. For example, QSIM [Kuipers, 1994] requires a transition mapping function when there is a change in operating regions. This function includes the new set of constraints, the variables whose magnitudes are unchanged, the variables whose derivatives are unchanged, and any values that must be asserted. This information is not directly accessible from the hybrid-DAE. De Kleer and Brown's qualitative physics based on confluences [de Kleer and Brown, 1984] uses *modes* and propagates non-local discrete changes through heuristics to provide a causal account of device behavior. Nishida and

Doshita [Nishida and Doshita, 1987] present two methods for handling discrete changes in qualitative simulation: (1) model it as continuous change that happens over infinitesimals, (2) model it as a sequence of mythical instants which may be inconsistent with the model. Iwasaki *et al.* [Iwasaki *et al.*, 1995] formulate instantaneous discrete changes using rules and hyperreal time semantics. Our approach pulls pieces from these approaches to allow discrete transitions using the hybrid-DAE.

Accounting for Modelica Events in Qualitative Simulation

Given a zero crossing, our qualitative simulation approach determines if there has been any change in the conditional constraints. If there has, then a discrete event occurs. The discrete event provides us with a new set of conditional constraints that we use to compute the results of the event as follows: (1) identify which variables are constant through the event, (2) solve for all consistent qualitative states using the new constraints and constant variables, (3) for each new consistent state, if it has the same constraints then return it, otherwise, trigger another discrete event. In previous approaches, step 1 is either performed using heuristics [Nishida and Doshita, 1987] or required as input [Kuipers, 1994]. We use a combination of the state variables specified by Modelica and the difference between the new and old constraints to determine which variables cannot change value discontinuously. Overall, our approach is analogous to Modelica's in that we are searching for a consistent set of values and conditions, with the state variables maintaining their values. Therefore, one of the intervals after the discrete transition must match the continuous-time behavior of the Modelica model after the event.

Alignment of Simulation Results

Consider an example of a locked brake that is subject to an increasing torque until it begins sliding. Table 1 contains a subset of the values generated by OpenModelica for the relevant variables as the brake begins sliding. At $time = 0.25$, the condition for $startForward$, $torque1.tau > 0.25$, signals a zero crossing. The search for consistent equations results in new values for the friction being applied by the brake, $brake1.tau$, its acceleration, and the boolean variables $locked$ and $startForward$. From the consistent set of equations at $time = .25000001$, another zero crossing is detected for the condition $w > 0$. The result of this event is that the $startForward$ is $false$, and the $mode$ is $forward$. After these two events, continuous integration begins and updates the values of the continuous-time variables. Even though events have no duration, OpenModelica increments time by 1E-9 for each set of concurrent events allowing for the sequence of instantaneous events to be recreated.

Figure 7 contains the relevant portion of the qualitative simulation produced by the Modelica model. Our algorithm is analogous to Modelica's with the difference that we record the results of search for the consistent set of equations in the envisionment. Thus, the first zero crossing occurs when $torque1.tau > 0.25$ in situation 2931. At this point, the constraints have changed. Therefore, we create a new situation (situation 3004) with the same values for the state variables, and then we solve for the rest of the variables. In this case, the value of $startForward$ resulting in a new set of constraints being active. This process repeats until situation 3156 which has the same constraints as situation 3080. At this point, the values in this situation correspond to the row in the Modelica simulation results at $time = 0.250000001$ from Table 1. The result of the next qualitative continuous integration triggers another event when $w > 0$. This event proceeds in the same manner resulting in a situation 4022 that corresponds to the Modelica simulation at $time = 0.250000002$.

Pseudo State Variables

One weakness of our approach is that it occasionally results in the ambiguous branching (i.e., many different sequences of instants following a single discrete change). Therefore, we allow the user to specify additional qualitative variables, *pseudo-state variables*, that do not change during discrete transitions. A piece of future work is to more tightly integrate our qualitative simulator with a symbolic equation solver (e.g., Macsyma [Bogen, 1986]) to be able to identify constant variables by their equations with respect to the system's state variables. Even with this extension, discrete transitions may introduce unrealizable ambiguities. As these transitions are extra, we maintain the desired alignment between the envisionment and the results of qualitative simulation.

4 Impact for Designers

By integrating qualitative reasoning into design tools, we foresee many potential benefits for designers. Including qualitative verification [Klenk *et al.*, 2012] where the tool would inform the engineer if the topology could meet the model's requirements (i.e., if a failure behavior is not possible). Other benefits would include highlighting potential failures and identifying irrelevant parameters. For example, a slider-crank mechanism will exhibit a kinematic singularity in a particular context of use. Furthermore, integration with a symbolic solving system such as Macsyma has the potential to reduce the design space further by identifying key parameter inequalities that remove safety requirement violations from the envisionment.

While spurious trajectories are a concern for qualitative simulation in isolation, we intend to use qualitative simulation in conjunction with other reasoning methods. Thus, the existence of spurious states and transitions will be identified by other analyses. Limiting spurious trajectories at a qualitative level is still important because the less spurious trajectories in the envisionment the better guidance qualitative simulation provides for other reasoning methods.

5 Discussion

We believe [Weld and de Kleer, 1989] qualitative reasoning is the fundamental basis upon which engineers reason about physical systems. Qualitative reasoning plays a key role in every facet of designing a system ranging from early stage design [Kurtoglu and Campbell, 2009] through understanding of simulation results, to planning design modifications to meet requirements. Unfortunately, none of the commonly

Table 1: Values generated by OpenModelica during the discrete transition during which brake begins to move. '+' indicates that the value has been truncated for presentation, and the real value is slightly greater than the value in the field. The headings 'phi', 'w', and 'a' correspond to the position, speed, and acceleration respectively of the brake. 'Brake1.tau' is the force applied by the brake against the applied torque of the system, 'torque1.tau'.

time	phi	w	a	brake1.tau	torque1.tau	locked	startForward	mode
0.248	0	0	0	0.248	0.248	true	false	stuck
0.25	0	0	0	0.25	0.25	true	false	stuck
0.250000001	0	0	0.125+	0.125	0.25+	false	true	stuck
0.250000002	1.2E-13	1.2E-10	0.125+	0.125	0.25+	false	false	forward
0.252	2.51E-7	0.000252	0.127	0.125	0.27	false	false	forward

used design/analysis tools provide computational QR support for these tasks. Leaving qualitative reasoning entirely to the human engineer risks missing critical inferences.

Our vision is to integrate qualitative reasoning into the tools and languages used by designers enabling the automation of these tasks. This paper presents a key step in that process by performing qualitative simulation with Modelica models, extending the qualitative constraint abstraction process to account for the representations used by quantitative solvers, and highlighting the challenges of discrete modeling as well as providing a solution that maintains the desired alignment between the simulation results.

In industry and academia, the role of computation in engineering design is rapidly expanding. Clearly, only producing an envisionment will not have much of an impact on the design process. Further exploration is required to determine how best to integrate qualitative reasoning into this process. We believe that an integrated qualitative reasoning design tool will enable the designer to explore the design space more thoroughly with less computation. This will allow the designer to focus more effort on the difficult problems thereby arriving at better design faster.

Acknowledgments

This work was partially sponsored by The Defense Advanced Research Agency (DARPA) Tactical Technology Office (TTO) under the META program and is Approved for Public Release, Distribution Unlimited. The views and conclusions in this document are those of the authors and should not be interpreted as representing the official policies, either expressly or implied, of the Defense Advanced Research Projects Agency or the U.S. Government.

References

[Bogen, 1986] Richard Bogen. *MACSYMA reference manual*. Symbolics, Incorporated, 1986.

[de Kleer and Brown, 1984] J. de Kleer and J. S. Brown. A qualitative physics based on confluences. *Artificial Intelligence*, 24(1):7–84, 1984. Also in: Bobrow, D. (ed.) *Qualitative Reasoning about Physical Systems* (North-Holland, Amsterdam, 1984 / MIT Press, Cambridge, Mass., 1985).

[Fritzson, 2004] P. Fritzson. *Principles of Object-Oriented Modeling and Simulation with Modelica 2.1*. Wiley-IEEE Press, Piscataway, NJ, 2004.

[Hinrichs *et al.*, 2011] Thomas R. Hinrichs, Kenneth D. Forbus, Johan de Kleer, Sungwook Yoon, Eric Jones, Robert Hyland, and Jason Wilson. Hybrid qualitative simulation of military operations. In Daniel G. Shapiro and Markus P. J. Fromherz, editors, *IAAI*. AAAI, 2011.

[Iwasaki *et al.*, 1995] Yumi Iwasaki, Adam Farquhar, Vijay Saraswat, Daniel Bobrow, and Vineet Gupta. Modeling time in hybrid systems: How fast is" instantaneous"? In *International Joint Conference on Artificial Intelligence*, volume 14, pages 1773–1781. Citeseer, 1995.

[Klenk *et al.*, 2012] Matthew Klenk, Johan de Kleer, Daniel G. Bobrow, Sungwook Yoon, John Hanley, and Bill Janssen. Guiding and verifying early design using qualitative simulation. In *Proceedings of the ASME 2012 IDETC and CIE*, Chicago, IL, 2012.

[Kuipers, 1994] Benjamin Kuipers. *Qualitative reasoning: modeling and simulation with incomplete knowledge*. MIT Press, Cambridge, MA, USA, 1994.

[Kurtoglu and Campbell, 2009] Tolga Kurtoglu and Matthew I Campbell. Automated synthesis of electromechanical design configurations from empirical analysis of function to form mapping. *Journal of Engineering Design*, 20(1):83–104, 2009.

[Nishida and Doshita, 1987] Toyoaki Nishida and Shuji Doshita. Reasoning about discontinuous change. In *Proc. AAAI*, volume 87, pages 643–648, 1987.

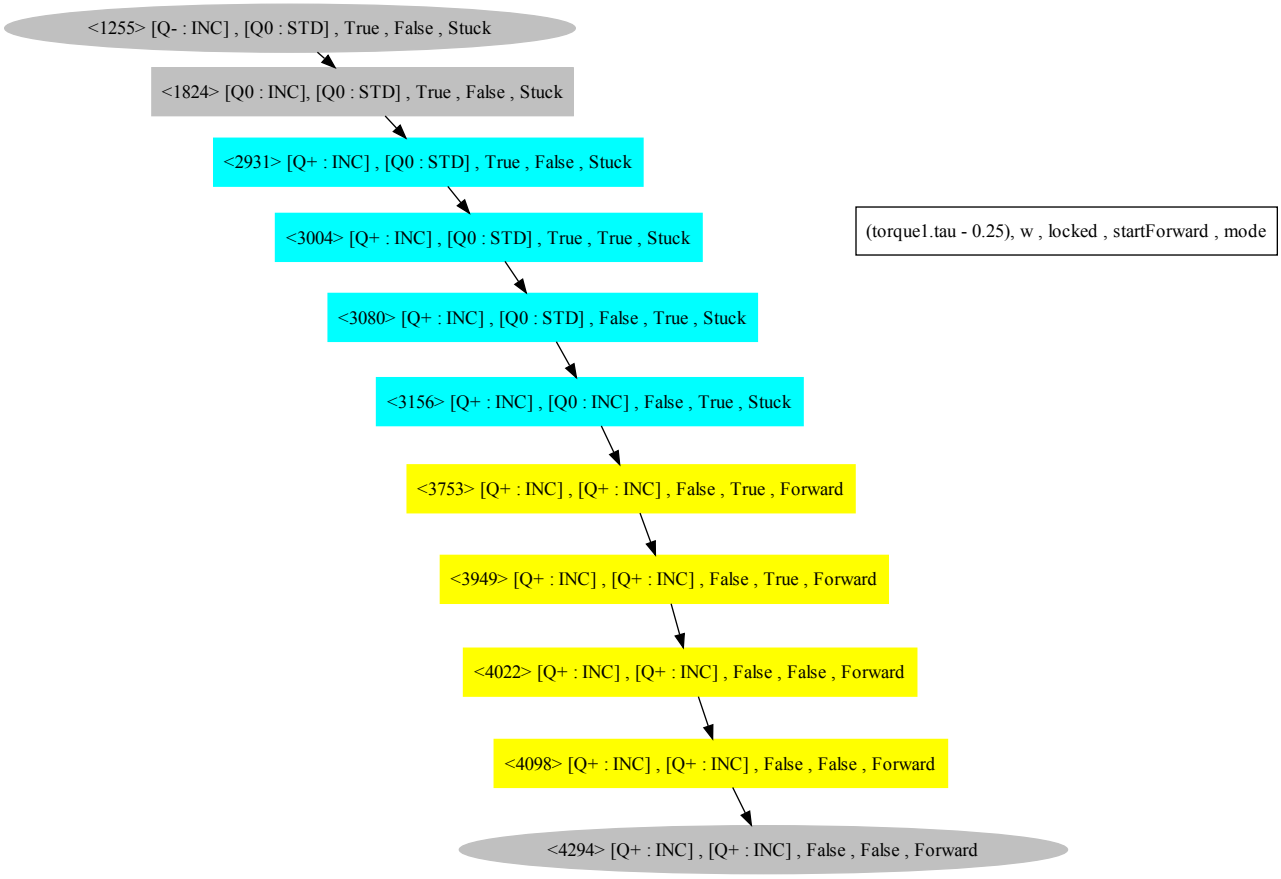
[Otter *et al.*, 1999] Martin Otter, Hilding Elmqvist, and Sven Erik Mattsson. Hybrid modeling in modelica based on the synchronous data flow principle. In *Computer Aided Control System Design, 1999. Proceedings of the 1999 IEEE International Symposium on*, pages 151–157. IEEE, 1999.

[Petzold, 1982] Linda R Petzold. Description of dassl: A differential/algebraic system solver. Technical report, Sandia National Labs., Livermore, CA (USA), 1982.

[Sachenbacher and Struss, 2005] Martin Sachenbacher and Peter Struss. Task-dependent qualitative domain abstraction. *Artif. Intell.*, 162(1-2):121–143, 2005.

[Simko *et al.*, 2012] Gabor Simko, Tihamer Levendovszky, Sandeep Neema, Ethan Jackson, Ted Bapty, Joseph Porter, and Janos Sztipanovits. Foundation for model integration: Semantic backplane. In *Proceedings of the ASME 2012 International Design Engineering Technical Conferences*

Figure 7: A subset of an envisionment in which an increasing torque causes a brake to start sliding. Ovals represent qualitative intervals and rectangles are qualitative instants. The cyan states correspond to the event triggered by the applied torque exceeding the maximum torque of the brake, and the yellow states correspond to the event triggered by the brake sliding.



& Computers and Information in Engineering Conference IDETC/CIE, pages 12–15, 2012.

[Struss and Price, 2004] Peter Struss and Chris Price. Model-based systems in the automotive industry. *AI Magazine*, 24(4):17–34, 2004.

[Struss, 1988] P. Struss. Mathematical aspects of qualitative reasoning. *International Journal of Artificial Intelligence in Engineering*, 3(3), 1988.

[Weld and de Kleer, 1989] D.S. Weld and J. de Kleer. *Readings in Qualitative Reasoning about Physical Systems*. Morgan Kaufmann, 1989.

[Yılmaz and Say, 2006] Özgür Yılmaz and AC Cem Say. Causes of ineradicable spurious predictions in qualitative simulation. *Journal of Artificial Intelligence Research*, 27:551–575, 2006.