Behavior Abstraction for Tractable Simulation

Daniel J. Clancy and Benjamin Kuipers
Department of Computer Sciences
University of Texas at Austin
Austin, Texas 78712
clancy@cs.utexas.edu and kuipers@cs.utexas.edu

Abstract

Most qualitative simulation techniques perform simulation at a single level of detail highlighting a fixed set of distinctions. This can lead to intractable branching within the behavioral description. The complexity of the simulation can be reduced by eliminating uninteresting distinctions. Behavior abstraction provides a hierarchy of behavioral descriptions allowing the modeler to select the appropriate level of description highlighting the relevant distinctions. Two abstraction techniques are presented. Behavior aggregation eliminates occurrence branching by providing a hybrid between a behavior tree representation and a history based description. Chatter box abstraction uses attainable envisionment to eliminate intractable branching due to chatter within a behavior tree simulation.

Introduction

The diagnosis and design of physical systems are difficult tasks due to the interaction of complex system components. Both tasks require a description of the possible behaviors of the system. Quantitative reasoning techniques are often unable to derive this information due to incomplete knowledge and the inability of numerical simulation to guarantee a description of all possible behaviors. Qualitative reasoning techniques [Weld and de Kleer, 1990] use incomplete knowledge to derive a qualitative description of the possible behaviors of the system.

Most qualitative reasoning paradigms perform simulation at a single level of detail highlighting a fixed set of distinctions. For many complex dynamical systems, this results in an intractable set of possible behaviors. Frequently, the distinctions made are irrelevant to the modeler. Furthermore, a great deal of complexity is added to the simulation computing the consequences of these distinctions.

Many of the behaviors from a complex behavior tree can be summarized by describing sets of similar behaviors. We are interested in characterizing the possible behaviors of a device via a lattice of finite descriptions that highlight different distinctions at various levels of detail. Figure 1 shows a portion of this lattice for a QSIM [Kuipers, 1986] simulation of a variant of a Proportional Integral (PI) controller. The lattice allows the modeler to perform a trade-off between tractably simulating the model and generating sufficient detail in the qualitative description for inference or query answering. By using on-the-fly abstraction, making explicit only those behavioral distinctions that are necessary for a particular task, the computational complexity of a qualitative simulation is reduced and the resulting behavioral descriptions are simplified by eliminating uninteresting details. As the modeler moves through the abstraction space more detailed descriptions can be computed.

Techniques have been developed that allow the modeler to view the behavior of the system from various perspectives and at various levels of abstraction. These abstraction techniques retain the QSIM soundness guarantee ensuring that all true behaviors of the system are included in the qualitative description. This paper will discuss two of these methods:

Behavior aggregation observes similarities between states in the behavior tree. Similar behaviors are combined into a single aggregate behavior during simulation to eliminate occurrence branching. Occurrence branching is due to the complete temporal ordering of events whose order is not constrained by the QDE. This builds on work done by Williams [1986] and Fouché and Kuipers [1991].

Chatter Box Abstraction eliminates chatter from a behavior tree simulation by performing a limited envisionment when variables begin to chatter. Only the chattering variables are allowed to vary in the envisionment. The resulting envisionment graph is abstracted into a single qualitative state that is included in the behavior tree.

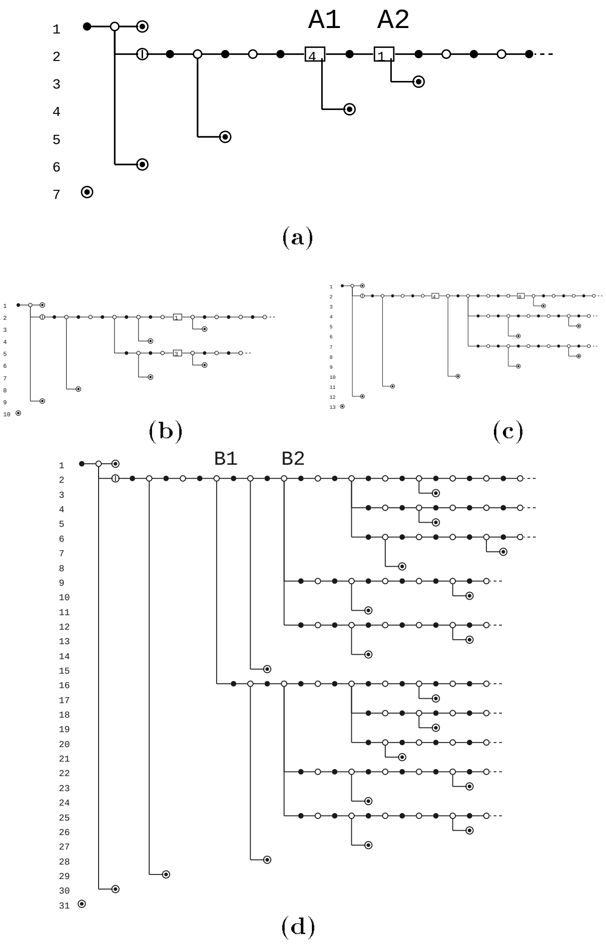


Figure 1: Partial lattice of aggregated behavior trees for the PI Controller

- In the most abstract tree (a, at top), aggregated state A1 abstracts the two way branch at state B1 in the original behavior tree (d). Aggregated state A2 abstracts the three-way branch at state B2.
- At the intermediate levels, tree (b) expands state A1 from (a); tree (c) expands state A2 from (a).
- Other aggregate states appear in trees (b) and (c).

Qualitative Simulation

Qualitative simulation techniques derive a behavioral description of a physical device from a structural representation. Constraints are specified for a set of variables via a qualitative differential equation (QDE). These variables are described by a set of qualitative values each containing a qualitative magnitude and a direction of change. The domain of each qualitative magnitude is described by a quantity space that specifies a finite, totally ordered set of landmark values. A qualitative magnitude is either a landmark or an open interval between landmarks. The simulation uses the QDE to derive a qualitative description of the system behavior.

A total *envisionment*, used in Confluences [de Kleer and Brown, 1984] and Qualitative Process Theory [Forbus, 1984], describes all possible states of the system. The qualitative states are represented as nodes in a directed graph with edges connecting temporally adjacent states. An attainable envisionment is the subgraph that is reachable from an initial state.

QSIM [Kuipers, 1986] uses a tree of qualitative states to represent the behaviors that follow from a set of initial values. Each path through the *behavior tree* represents a different system behavior. New landmarks are created during the simulation identifying newly discovered critical values within the quantity spaces of the state variables. The behavior tree representation along with the new landmarks are used by behavior-based filters [Fouché and Kuipers, 1992, Lee and Kuipers, 1988] to eliminate spurious behaviors from the tree. Semi-quantitative reasoning techniques [Kuipers and Berleant, 1988, Berleant and Kuipers, 1991, Kay and Kuipers, 1992] provide a more detailed description of the behaviors by inferring quantitative bounds for the newly introduced landmarks within the behavior tree.

Williams [1986] describes an alternate simulation technique based upon individual variable histories. A *history* defines a sequence of qualitative values for a single variable. A *concise history* is a sequence of such values in which each value is distinct from its neighbors. The behavior of the system is described in terms of a history for each variable. Histories are temporally correlated only when necessary as op-

Fouché and Kuipers [1991] apply state-based and behavior-based abstraction techniques to attainable envisionment graphs to provide a hierarchy of descriptions. A more abstract graph results with one state for each equivalence class.

- The state based techniques eliminate distinctions by combining states into equivalence classes by focusing on certain distinctions in the qualitative states.

Focus On:	Branches Eliminated:
Qualitative Magnitude	Eliminates chatter except around a landmark.
Qualitative Derivative	Eliminates some occurrence branching.
Interesting Variables	Eliminates distinctions caused by intermediate variables including occurrence branching and chatter within these variables.

- The behavior based abstraction method eliminates occurrence branching by collapsing Single Input Single Output (SISO) subgraphs within the envisionment graph. All paths through the SISO subgraph must have identical concise histories.

Figure 2: Summary of abstraction techniques from [Fouché and Kuipers, 1991]

posed to the complete temporal ordering provided by a behavior tree or envisionment representation.

These techniques for describing the behavior of a system are effective in different situations. Our techniques integrate these three methods. Behavior aggregation provides a hybrid between a behavior tree representation and a history based description, while chatter box abstraction uses an envisionment to eliminate intractable branching due to chatter within a behavior tree simulation.

Behavior Aggregation

Behavior aggregation attempts to eliminate distinctions which do not affect the subsequent behavior of the system. Figure 2 summarizes various abstraction techniques for envisionment graphs developed by Fouché and Kuipers [1991]. Behavior aggregation extends the behavior based abstraction techniques to other types of occurrence branching and performs the aggregation on the fly (i.e. while the simulation is occurring) during a behavior tree simulation. Behavior aggregation is being extended to include other abstraction techniques.

Occurrence branching results when the temporal ordering of a set of events is unconstrained by the QDE. An event occurs when a variable crosses a landmark or its derivative reaches zero. Following the events that cause the occurrence branch, the behaviors return to qualitatively equivalent states and the subsequent behaviors are essentially identical. Slight variations may occur due to the introduction of landmarks along certain behaviors and not others. This branching needlessly increases the complexity of the simulation and of the behavior tree.

In its simplest form, occurrence branching results when the events occur among unrelated variables. In this case, the only qualitative distinction among the behaviors is the temporal ordering of the events. In more complicated instances, the variables are related via a derivative relationship. Distinct histories result when the occurrence branch involves a variable approaching a landmark while its derivative approaches zero. Once the derivative reaches zero, the variable no longer attains the approaching landmark. This is referred to as *landmark attainment* occurrence branching. Figure 3 describes two instances of this within the PI-controller simulation.

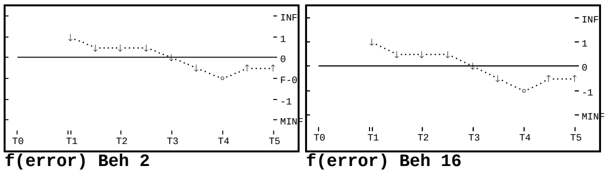
Behavior aggregation eliminates occurrence branching by combining the behaviors in the SISO subtree into a single *aggregate state*. An aggregate state represents an open time interval which describes the behavior of the system at various levels of detail using individual histories for each variable. The temporal correlation between events is eliminated. The simulation is continued for this subtree from a single abstract state representing the common properties.

Occurrence branching can occur at various points throughout the simulation. This results in a lattice of behavioral descriptions depending upon which instances of occurrence branching are eliminated. Figure 1 shows a portion of this lattice for the PI-controller simulation. The user can move through this abstraction space either manually or automatically to select the appropriate level of description. Behavior aggregation initially performs the simulation at the highest level of abstraction. The more detailed levels of description are only calculated when requested by the modeler. This lattice of descriptions will be extended further as more abstraction techniques are developed.

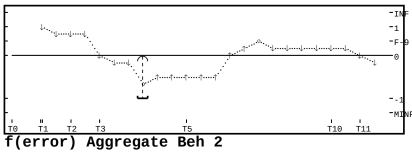
Aggregate States

An aggregate state is a single state within the behavior tree which describes a set of similar segments of behaviors with distinctions resulting from occurrence branching. It abstracts the shared properties of these behavior segments by using a history based description to characterize the behavior of

- Two behaviors (a) from branch B1 in figure 1d showing a landmark attainment branch on -1 at time t4. In the first behavior, the derivative reaches zero before the landmark is attained while in the other behavior these two events occur simultaneously. All other variables have identical histories. An abstract state is formed at (t4 t5) when the magnitudes are equivalent. The completed aggregate behavior (b) eliminates landmark F-0 since it cannot be matched against a landmark in the other behavior. The qualitative magnitude at t4 becomes [-1 0).

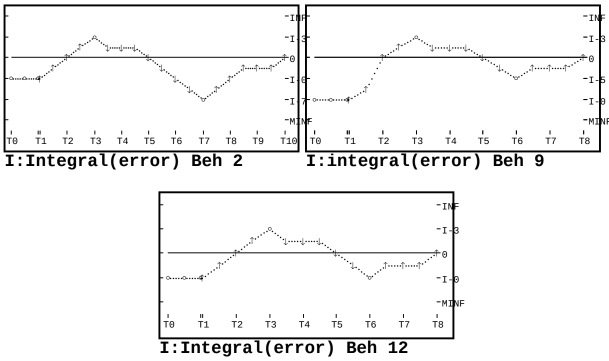


(a)

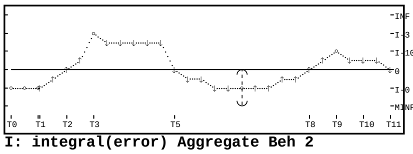


(b)

- Three behaviors (c) from branch B2 in figure 1d showing a landmark attainment branch on I-0 between t6 and t8. The first behavior exceeds the landmark, the second does not reach it while the third becomes steady at the landmark. An abstract state is formed once the magnitude rises above I-0. The completed aggregate behavior (d) eliminates landmarks I-5 and I-7 since they cannot be matched. The qualitative magnitude at t6 becomes (minf 0).



(c)



(d)

Figure 3: Original and aggregated behaviors for the PI-controller

each variable independently. In general, each variable's concise history is the same in each of the aggregated behaviors. In some cases, however, not all of the histories for a single variable are identical (e.g. landmark attainment occurrence branching in figure 3.) An aggregate state describes each variable's behavior over the abstracted interval at three levels of detail.

History Graph The set of histories for each variable over the abstracted interval can be combined into a graph with a single starting point and single ending point. This representation retains all of the information in the abstracted portion of the behavior tree except for the temporal correlation of events in different variable histories. The history graph begins at the initial point of the occurrence branch and branches only if the histories diverge as in landmark attainment occurrence branching. If the histories are identical, this combination results in a single unique concise history.

A Single Aggregate History The history graph is abstracted to a single concise history. The union of the qualitative values at each branch in the history graph is used to form this history. This history is used in the standard QSIM behavior display.

Summary Value A single qualitative value which provides an upper and lower bound for the histories over this interval. This level of description is used by various filters within QSIM.

As other abstraction techniques are developed, the history graph will prove particularly useful since it allows the modeler to view the behavior of each system variable independently.

The Algorithm

Behavior aggregation maintains a record of all qualitatively equivalent states within the behavior tree throughout the simulation. Equivalent states are combined into a single abstract state when they form a *spanning set* for a subtree within the behavior tree. This subtree is collapsed into an aggregate state and the simulation continues from the abstract state. The QSIM soundness guarantee is retained since the abstract state is equivalent to the states which it replaces. The four main steps within the algorithm are:

1. Determining the qualitative equivalence of states.
2. Combining equivalent states into a single abstract state.
3. Selecting subtrees to aggregate.
4. Abstracting the subtree into an aggregate state.

Qualitative Equivalence The QSIM algorithm defines a qualitative state by a set of qualitative values and a quantity space (qspace) for each state variable. Two states are considered qualitatively equivalent if each variable has (1) the same qualitative magnitude with respect to the *joint qspace* for that variable, and (2) the same direction of change. The joint qspace for a variable is defined as the intersection of the qspaces for the states being compared. The joint qspace eliminates any landmarks which have been introduced since these behaviors diverged.

Combining Equivalent States Behavior aggregation combines qualitatively equivalent states into a single minimally abstract state. The quantity space for each variable is the joint qspace for that variable in the equivalent states. Landmarks which have been introduced since the behaviors diverged must either be matched against equivalent landmarks in the other states or eliminated. The qualitative values in the abstract state are defined over these new quantity spaces.

A minimally abstract state matches as many landmarks as possible thereby minimizing the number of eliminated landmarks. Two landmarks can be matched if they are defined within the same interval of the joint qspace and have been created for the same reason. Landmarks are created for various reasons including region transitions and changes in the direction of change. Figure 3 demonstrates the elimination of landmarks from the abstract state. Following the creation of an aggregate state, the simulation will be continued from this minimally abstract state.

Selecting Subtrees for Aggregation Behaviors within a tree can be combined in various ways to highlight different distinctions. Currently, behavior aggregation is applied when a set of qualitatively equivalent states form a spanning set for a subtree within the behavior tree. A spanning set must contain one state from each path in the subtree. Methods of loosening this restriction are currently being investigated. The behavior segments extending from the root of the subtree to the spanning set are combined into an aggregate state.

Aggregating a Subtree A single aggregate state is formed from the selected subtree. An aggregate state uses a history based approach to describe the behavior of each system variable over the abstracted time interval. For each variable a set of concise histories defined over the same quantity space is derived from the behavior tree. For most variables these concise histories will be identical. The concise histories are combined into a history graph with a single beginning point and a single ending point. Each history graph is summarized by a single aggregate history which eliminates branching by using a more abstract qualitative description. The most abstract description of the behavior by an aggregate state consists of a single summary value for each variable. The summary value is the union of the values taken on by that variable in its history.

The spanning set algorithm generalizes and extends the Single-Input Single-Output (SISO) subgraph used by Fouché and Kuipers [1991]. The states that comprise the spanning set are represented by a single state in an envisionment graph. When they are combined into an abstract state, a SISO subgraph is formed in the behavior tree. This subgraph is then collapsed into an aggregate state. Fouché and Kuipers require all paths within the SISO subgraph to have identical concise histories for the abstraction to be performed. This restriction is eliminated in behavior aggregation. In addition, behavior aggregation is performed on the fly during a behavior tree simulation. It reduces the complexity of the simulation while allowing for the introduction of new landmarks and the application of behavior-based filters.

Chatter Box Abstraction

A major source of uninteresting distinctions within qualitative simulation is chatter. Chatter occurs when the derivative of a variable is constrained only by continuity. The simulation branches on all possible values for the qualitative derivatives of the chattering variables resulting in intractable branching. Figure 4 demonstrates this intractable branching while the phenomenon of chatter is described in more detail in figure 5.

Chatter box abstraction eliminates chatter within a behavior tree simulation by using an envisionment to abstract the chattering region into a single qualitative state. A *chatter box* is a region within the state space of the model in which the qualitative derivatives of potentially chattering variables are allowed to vary while the qualitative values of the other variables remain the same. When the behavior tree simulation enters a potential chatter box, an attainable envisionment is performed from this state. The envisionment is limited to the region of the state space identified by the chatter box. States exiting this region are suspended from simulation during the envisionment. If the envisionment graph exhibits chatter, then it is abstracted into a single state which is inserted into

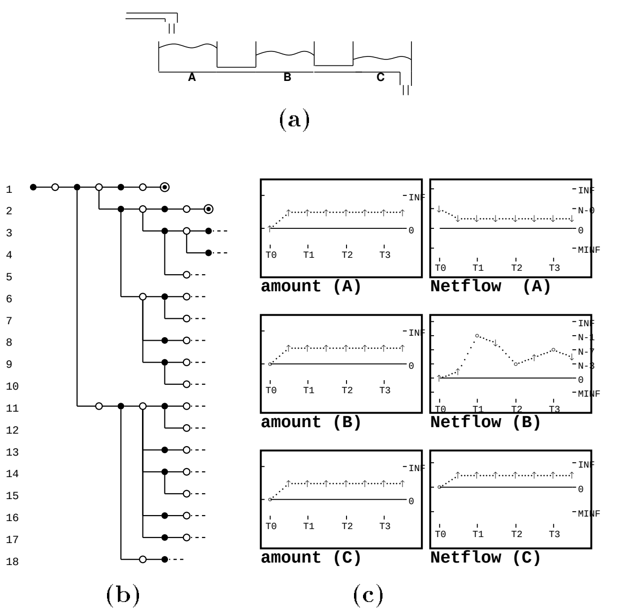


Figure 4: Intractable branching due to chatter in the simulation of a W tube.

- In a qualitative model of three tanks arranged in sequence connected by tubes (a), *NetflowB(t) = InflowB(t) - OutflowB(t)* is constrained only by continuity in the interval $(0, \infty)$.
- The simulation branches on all possible trajectories of *NetflowB(t)* while all other variables are completely uniform (c). This results in an infinite behavior tree (b) with intractable branching.
- Other techniques (higher order derivatives and ignoring qdirs) have been used to eliminate chatter in other variables within the model. These techniques are unable to eliminate the chatter in *NetflowB(t)*.

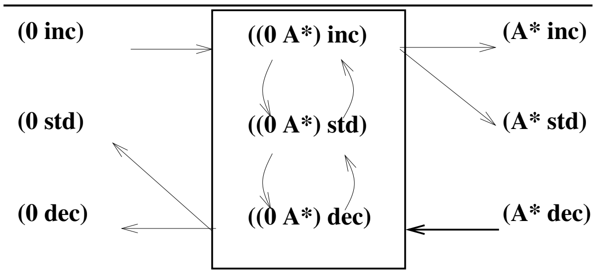


Figure 5: Possible qualitative value transitions for a QSIM variable within the interval (0 A*).

- Chatter occurs when a variable constrained only by continuity is changing within an interval. The variable remains within the boxed region with its derivative alternating between *inc*, *std*, and *dec*.
- After entering the interval (0 A*), the simulation branches depending upon whether the variable becomes steady within the interval or exits the region. If the variable becomes steady another branch occurs depending upon whether the variable begins to increase or decrease. This branching sequence continues throughout the simulation as long as the variable remains within the region.
- Chatter box abstraction abstracts the boxed region of the state space into a single time-interval state with its successors being the states that exit the chatter box.

the behavior tree. States within the envisionment graph will differ only in the qualitative derivative of the chattering variables. The states that exit the chatter box during the envisionment are then used as the successors of this abstract state. Figure 6 demonstrates the chatter box algorithm on the W tube model.

Previous Solutions

Two methods have been developed for eliminating chatter within a QSIM behavior tree simulation [Kuipers and Chiu, 1987, Kuipers et. al., 1991]. Neither of these techniques completely eliminate the problem. The Higher Order Derivative (HOD) technique uses the second and third order derivatives of the chattering variables to determine the direction of change for unconstrained variables within an interval. Expressions for these derivatives must be derived by algebraic manipulation of the existing equations. Deriving these expressions from the model is not always possible. The HOD method also adds the assumption that monotonic functions described by M+ or M- constraints are well behaved in a particular way [Kuipers et. al., 1991]. Finally, this technique only eliminates chatter when it is a spurious behavior (i.e. not a true behavior) of the system.

Ignore Qdirs eliminates chatter by ignoring distinctions in the derivatives of the chattering variables. This abstraction technique is applied throughout the simulation. This can lead to over abstraction reducing the constraining power of the model. In addition, it requires the modeler to select the derivatives to be ignored prior to simulation.

Chatter box abstraction supersedes the need for ignoring qdirs by selectively abstracting the derivative of a variable only when needed. Since an envisionment is performed over the chattering region the constraining power of the model is not reduced. In addition, it can be applied in many cases where the higher order derivative solution is not applicable.

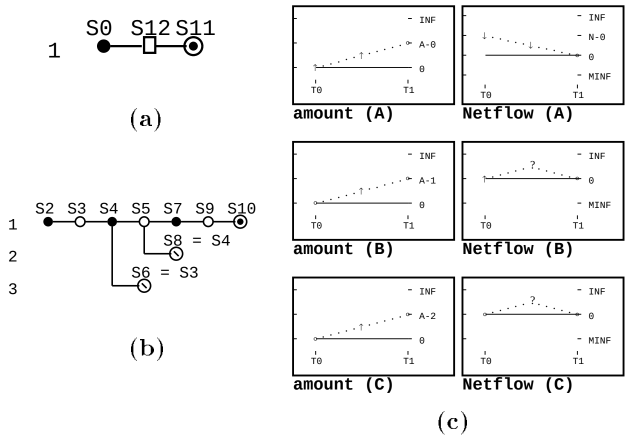


Figure 6: Simulation of a W Tube using Chatter Box Abstraction

- Using chatter box abstraction, QSIM produces a behavior tree (a) with a single behavior (c).
- When the region of the state space which exhibits chatter is entered, an attainable envisionment limited to this region is performed. There are 7 potentially chattering variables within this model. In this example, higher order derivatives are used to eliminate chatter in all of the variables except *NetflowB(t)* and *NetflowC(t)*. The envisionment graph (b) has 3 behaviors and 9 states. If desired, chatter box abstraction can eliminate chatter in all 7 of these variables without using higher order derivatives resulting in an envisionment graph with 50 behaviors and 74 states.
- The original successor of S-0 is used to create S-2 and begin the limited envisionment. The states between S-2 and S-9 in the envisionment graph (b) are abstracted into a single time-interval state (S-12) which is inserted in the behavior tree as a successor of S-0. A single state (S-10) exits the chattering region and is copied to create S-11 in the behavior tree. Both *NetflowB(t)* and *NetflowC(t)* are $(0 \, std)$ in this state. The envisionment graph provides more detail about the behavior of the system within this region.

Chatter Around Landmarks

The existence of a landmark within the chattering region leads to a more complicated version of chatter. The simulation branches at the landmark resulting in a change in the qualitative magnitude as well as the direction of change. Previous methods for eliminating chatter have been unable to deal with this phenomenon.

During limited envisionment, the qualitative magnitude of the chattering variable is allowed to vary within an open interval bounded by adjacent landmarks. We can extend this approach to eliminate chatter around a landmark by allowing the open interval to include a landmark (even zero!). The behavior language must also be extended to allow qualitative magnitudes bounded by non-adjacent landmarks in the abstract state.

Chatter box abstraction automatically eliminates chatter around landmarks introduced during the simulation. However, it currently requires the modeler to identify any user landmarks around which chatter is likely to occur prior to the simulation. The behavior is only abstracted if the system actually exhibits chatter.

Conclusions

Qualitative reasoning techniques provide a mechanism for reasoning from incomplete knowledge about the set of possible behaviors of a physical system. These techniques can be used in a number of tasks including diagnosis, design, explanation, and question answering. Qualitative reasoning techniques, however, tend to reason at a single level of abstraction. This can lead to a large number of possible behaviors and a possibly intractable simulation.

Behavior abstraction reduces the complexity of the qualitative simulation by eliminating uninteresting distinctions. A lattice of behavioral descriptions allows the modeler to select the appropriate level of description highlighting the relevant distinctions. Techniques are being developed to automatically search the lattice for a sufficiently detailed behavioral description to respond to a query. Two abstraction techniques have been presented. Behavior aggregation combines a behavior tree simulation with a history based representation to eliminate occurrence branching. Similar behaviors are combined into a single aggregate behavior. Chatter box abstraction combines a behavior tree simulation with an envisionment to eliminate intractable branching due to chatter. A more complete characterization of the effectiveness of these techniques to tractably simulate models is currently being developed.

Other abstraction techniques are being developed to further reduce the complexity of the qualitative simulation including extending behavior aggregation to other phenomenon besides occurrence branching. One technique uses an envisionment as a guide to focus a behavior tree simulation [Clancy and Kuipers, 1992]. The behavior tree simulation is performed only for behaviors lying within a selected subgraph of the envisionment graph. The complexity of the simulation depends upon the selected behaviors. Thus, intractable branches that are not of interest are eliminated from the simulation. In addition, static evaluation methods are being investigated which identify loosely connected components within the QDE. The simulation of these components can be performed independently and combined only when the components interact.

Acknowledgments

This work has taken place in the Qualitative Reasoning Group at the Artificial Intelligence Laboratory, The University of Texas at Austin. Research of the Qualitative Reasoning Group is supported in part by NSF grants IRI-8905494, IRI-8904454, and IRI-9017047, by NASA contract NCC 2-760, and by the Jet Propulsion Laboratory.

References

D. Berleant and B. J. Kuipers. Bridging the gap from qualitative to numerical simulation. Technical Report AI91–158, Artificial Intelligence Laboratory, University of Texas at Austin, Austin, Texas 78712, March 1991.

D. J. Clancy and B. J. Kuipers. Aggregating Behaviors and Tractable Simulation. In *AAAI Design from Physical Principles Fall Symposium Working Notes*, pp 38-43, Cambridge, MA, 1992.

J. de Kleer and J. S. Brown. A qualitative physics based on confluences. *Artificial Intelligence*, 24:7–83, 1984.

B. Falkenheiner and K. D. Forbus. Setting up large-scale qualitative models. In *Proceedings of the Seventh National Conference on Artificial Intelligence (AAAI-88)*, pages 301-306, 1988.

K. D. Forbus. Qualitative process theory. *Artificial Intelligence*, 24:85–168, 1984.

P. Fouché and B. J. Kuipers. Towards a Unified Framework for Qualitative Simulation. In *Proceedings of the Fifth International Workshop on Qualitative Reasoning about Physical Systems*, 295-301, 1991.

P. Fouché and B. J. Kuipers. Reasoning about energy in qualitative simulation. In *IEEE Transactions on Systems, Man and Cybernetics*, 22, 1992.

H. Kay and B. J. Kuipers. Numerical Behavior Envelopes for Qualitative Models In *Proceedings of the Sixth International Workshop on Qualitative Reasoning*, 1992.

B. J. Kuipers. Commonsense reasoning about causality : Deriving behavior from structure. *Artificial Intelligence*, 24:169–204, 1984.

B. J. Kuipers. Qualitative simulation. *Artificial Intelligence*, 29:289–338, September 1986.

B. J. Kuipers and D. Berleant. Using incomplete quantitative knowledge in qualitative reasoning. In *Proceedings of the Seventh National Conference on Artificial Intelligence*, pages 324–329, 1988.

B. J. Kuipers and C. Chiu. Taming intractable branching in qualitative simulation. In *Proceedings of the Tenth International Joint Conference on Artificial Intelligence (IJCAI-87)*. Los Altos, CA: Morgan Kaufman, 1987.

B. J. Kuipers, C. Chiu, D. Dalle Molle and D. R. Throop. Higher-order derivative constraints in qualitative simulation. *Artificial Intelligence*, 51:343-379, 1991.

W. W. Lee and B. J. Kuipers. Non Intersection of Trajectories in Qualitative Phase Space: A Global Constraint for Qualitative Simulation. In *Proceedings of AAAI 88*, 286-291, 1988.

D. S. Weld and J. de Kleer (Eds.). *Readings in Qualitative Reasoning About Physical Systems.* Los ALtos, CA: William Kaufman.

B. C. Williams. Doing time: Putting qualitative reasoning on firmer ground. In *Proceedings of the American Conference on Artificial Intelligence (AAAI'86)*, pages 105-113, Philadelphia, PA. August 1986.