A New Approach of Qualitative Simulation for the Validation of Hybrid Systems

Slim Medimegh, Jean-Yves Pierron, Jean-Pierre Gallois, Frédéric Boulanger

A New Approach of Qualitative Simulation for the Validation of Hybrid Systems

Slim Medimegh[1], Jean-Yves Pierron[1],
Jean-Pierre Gallois[1], and Frédéric Boulanger[2]

[1] CEA, Saclay, France *firstname.lastname*@cea.fr
[2] LRI, CentraleSupélec, Université Paris-Saclay, France frederic.boulanger@lri.fr

Abstract. Hybrid systems are specified in a heterogeneous form, with discrete and continuous parts. Simulating such systems requires precise data and computational power in order to synchronize continuous changes and discrete transitions. However, in the early design stages, the lack of precision about some parameters forbids such simulations. Qualitative simulation consists in computing only some *qualities* of the behavior, without computing the exact values. We present here a new approach for the qualitative simulation of hybrid systems, which relies on an abstract model of the state variables and of the qualitative aspects of their evolution. This model is analyzed by the Diversity symbolic execution engine to build the qualitative behaviors of the system.

Keywords: hybrid systems, qualitative simulation, verification, validation

1 Introduction

Embedded software has become essential in most industrial sectors: energy, transport, telecommunications, healthcare, etc. To ensure a high level of reliability, it is essential to carry out, in the early phases of the development cycle, an analysis of the behavior of the system when it interacts with its environment. This environment usually involves various business knowledge (mechanical, hydraulics, electronics) and it is important to describe each of them in an appropriate formalism. The whole system (the software and its environment) is specified in a heterogeneous manner and contains discrete and continuous parts. The simulation of such hybrid systems requires precise data and computational power in order to detect changes in the continuous values and to synchronize them with discrete transitions. However, during the early stages of the design, the exact value of some parameters is not known yet, while it is already necessary to check the possible behaviors of the system to make some design decisions.

Qualitative simulation can help here. When the behavior of the continuous state variables is described by differential equations, it partitions their domain of variation into discrete domains of qualitative behavior (increasing, decreasing, or constant) according to the sign of their first derivative. Symbolic execution can then produce a tree of abstract behaviors, which can be combined with the

model of the discrete part of the system. When the differential equations are not available, qualitative simulation can be performed using a qualitative model which describes only the causal links between the qualitative behavior of the variables and their derivatives, in the form of a state machine.

In this article, we present a new approach of qualitative simulation, which relies on a qualitative model of the differential equations of the continuous part of the system, taking into account the first and second derivatives of the state variables, and on a model of execution implemented in the Diversity tool to compute the qualitative behavior of the hybrid system.

2 Hybrid Systems

Hybrid systems are dynamic systems that explicitly and simultaneously combine continuous and discrete behaviors. The modeling of hybrid dynamical systems can be made using various formalisms, among which hybrid automata, which are defined by a set of continuous variables and states, and discrete transitions that include guards and assignments to these variables. The evolution of the continuous variables can be described by different kinds of differential equations.

Polyhedral hybrid automata have linear differential equations and guards. *Rectangular* hybrid automata have constant differential equations and guards that are comparisons to constants. *Timed* hybrid automata have differential equations and guards involving time [8], and can be handled by Uppaal [1], Kronos [3] and recently TiAMo [2]. Some tools and methods deal with other types of hybrid systems: Hytech [9] for rectangular hybrid automata with linear guards on the transitions, and CheckMate for non linear continuous evolutions with linear guards [4].

3 Qualitative Simulation

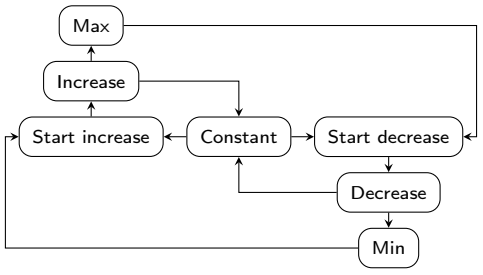

Fig. 1. Qualitative model of the evolution laws of continuous variables

Qualitative simulation is a research field related to artificial intelligence, to automate the reasoning on continuous aspects of systems. It considers only qualitative values, ignoring exact quantities. The purpose of qualitative simulation is to develop a representation that makes it possible to reason about the behavior of the physical system without the quantitative information. Figure 1 shows the kind of qualitative changes that we are interested in.

For hybrid systems, the discrete part of the system is not influenced by the qualitative abstraction process. However, continuous variables are discretized in order to consider only their qualitative changes. Therefore, continuous transitions become discrete transitions, and the resulting system is entirely discrete and can

be treated by traditional methods (property verification, test generation) for the verification of discrete systems.

4 Related Work

Many methods have been developed for qualitative reasoning, for instance:

- The component approach in EnVision by Kleer and Brown, builds a qualitative model of the system from its components. It takes into consideration the qualitative sign of the variables. It determines the qualitative states and the transitions between these states [10].
- The Qualitative Process Theory developed by K. D. Forbus models objects and processes which cause changes to the objects. Qualities of objects are defined by inequalities about quantities associated to these objects, and the qualitative behavior of a system is obtained by reasoning on the processes [6].
- Qualitative Simulation, in which the qualitative model of the system is a differential qualitative equation which is an abstraction of an equivalence class of ordinary differential equations. It relies on the continuity properties of the variables to build a sequence of qualitative states describing the behavior of the system [11]. Many tools have been developed for qualitative analysis: KAM [16] which deals with nonlinear system with two freedom degrees, Maps [17] which deals with second and third order systems, PSX2NL [12] which deals with ordinary differential equations, POINCARE [14] which also deals with ordinary differential equations with one parameter.
- Qualitative simulation for polynomial continuous evolutions developed by Tiwari [15]: the system is modeled as a discrete system whose guards are polynomial, and these guards can be evaluated by the QEPCAD tool.

Our approach is similar to the Tiwari method, but based on qualitative values for avoiding the evaluation of polynomial guards with QEPCAD.

5 Qualitative Simulation with Diversity

We have implemented our approach, which focuses on first and second derivatives, in Diversity, a symbolic execution engine developed at CEA LIST in order to prove safety properties and to automatically generate tests from state machines models of complex systems [7,13]. To analyze complex systems that include continuous parts in hybrid automata, CEA LIST had to extend Diversity with qualitative methods [8]. For this, the state machine of hybrid models has continuous variables, which must be abstracted by a discretization method.

Discretization process

Each continuous variable is represented by a state machine which encodes its qualitative state: increasing, decreasing, constant etc. The guard of the transitions depends on the sign of the derivative of the variable, which is constrained by the differential equations. These equations are analyzed to determine under which conditions some qualitative changes are possible or not. When the change

is not possible, the corresponding execution branch is cut. When the change is possible, the branch is tagged with the condition of the change.

When the structure of the equations is not suitable for processing by a solver, or to avoid the cost of solving the differential equations, it is possible to perform qualitative simulation without differential equations, by using only a qualitative model of these equations. The results are of course less precise, but this approach can be used at very early stages of the design.

From this model, Diversity generates a tree of behaviors, which can be analyzed for reachability properties or cycles. Execution traces can be compared with the result of quantitative simulations, and the compliance of the numerical simulation to the qualitative simulation can be established.

6 A New Approach for Qualitative Simulation

In order to improve the qualitative simulation without differential equations in Diversity, we developed a model of execution which constraints the qualitative evolution of the state variables, and computes their qualitative behavior.

6.1 Model of Execution

In this model of execution, we consider only continuous changes of the state variables and their first derivative. The second derivative, which is the result of input forces, may not be continuous (but this choice is not definitive). For instance, the value of a state variable cannot change from *Negative* to *Positive* without being *Null* in between. Moreover, the value cannot change from *Negative* to *Null* unless the first derivative is *Positive*, and the same is true for the first derivative with regard to the second derivative.

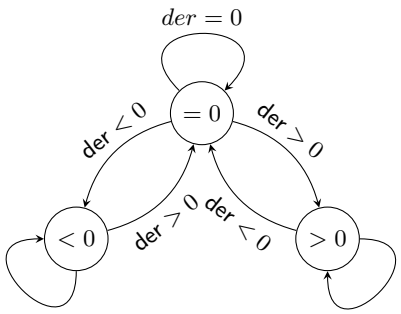

Fig. 2. Qualitative changes

These constraints of continuity and derivability form a model of execution which reflects the physical nature of the phenomenon that we are modeling. It is modeled using state machines as illustrated in Figure 2 for the control of the value of a state variable.

A similar automaton controls the changes of the first derivative with respect to the second derivative. These state machines are nondeterministic. For instance, in the < 0 state, if the first derivative is *Positive*, we can either stay in the current state, or go to the $= 0$ state.

6.2 Computing Qualitative Behaviors

In order to compute the qualitative behavior of a state variable, we observe the changes of the values attached to this variable. For instance, when the second derivative is *Negative* and the first derivative is *Null*, we have a *Maximum*. However, the current value of a state variable and the current value of its first and

second derivatives are not enough to make a difference between a maximum and a start of decrease after a constant stage. We therefore model each continuous state variable in the system by five values: the current (x) and previous (x_{-1}) values of the variable, the current ($\dot{x}$) and previous ($\dot{x}_{-1}$) values of its first derivative, and its second derivative ($\ddot{x}$).

Taking into account only $\dot{x}_{-1}$, $\dot{x}$ and $\ddot{x}$, we identified 13 qualitative states, which cover our analysis goals, without the need for higher order derivatives:

Constant when $\dot{x}_{-1}$, $\dot{x}$ and $\ddot{x}$ are null, **FlexStartIncrease** when $\dot{x}_{-1} = 0$, $\dot{x} = 0$ and $\ddot{x} > 0$. The first derivative is not positive yet, but the dynamics is transitioning toward an increase. **FlexStartDecrease** when $\dot{x}_{-1} = 0$, $\dot{x} = 0$ and $\ddot{x} < 0$. Similar to the previous case when transitioning toward a decrease. **StartIncrease** when $\dot{x}_{-1} = 0$ and $\dot{x} > 0$. The first derivative has just become positive. Notice that there is no condition on the second derivative, which may have come back to 0. **StartDecrease** when $\dot{x}_{-1} = 0$ and $\dot{x} < 0$. **Increase** when $\dot{x}_{-1} > 0$ and $\dot{x} > 0$. The first derivative is positive. **Decrease** when $\dot{x}_{-1} < 0$ and $\dot{x} < 0$. The first derivative is negative. **Maximum** when $\dot{x}_{-1} > 0$, $\dot{x} = 0$ and $\ddot{x} < 0$. The three conditions are necessary to distinguish this case from other qualitative states such as inflection points. **Minimum** when $\dot{x}_{-1} < 0$, $\dot{x} = 0$ and $\ddot{x} > 0$. The three conditions are necessary to differentiate this case from other qualitative states such as inflection points. **FlexIncrease** when $\dot{x}_{-1} > 0$, $\dot{x} = 0$ and $\ddot{x} > 0$. Inflection point during an increase. **FlexDecrease** when $\dot{x}_{-1} < 0$, $\dot{x} = 0$ and $\ddot{x} < 0$. Inflection point during a decrease. **StopIncrease** when $\dot{x}_{-1} > 0$, $\dot{x} = 0$ and $\ddot{x} = 0$. The state variable reaches a plateau at the end of an increase. **StopDecrease** when $\dot{x}_{-1} < 0$, $\dot{x} = 0$ and $\ddot{x} = 0$. The state variable reaches a plateau at the end of a decrease.

By additionally taking into account x_{-1} and x, it is possible to distinguish sub-cases in these qualitative states, for instance "reaching a null maximum" when reaching a maximum with $x_{-1} < 0$ and $x = 0$.

There are impossible cases. Two are due to the continuity of the variable: the conjunction of $x_{-1} < 0$ and $x > 0$, and the conjunction of $x_{-1} > 0$ and $x < 0$ are impossible. Two others are due to the continuity of the first derivative (no angular points). Ten others are due to the fact that each variable change should be consistent with the previous value of the first derivative. For instance, with $x_{-1} = 0$ and $\dot{x}_{-1} = 0$, we necessarily have $x = 0$, because starting from 0 with a null first derivative, the variable cannot change. Other cases are similar.

6.3 Illustrative Example

This technique was applied to the simple example of a bouncing ball. In this example, we have only one state variable, the height of the ball above the ground, noted z. The usual way to model the bouncing ball is to consider that it has only one state, in which it is in free fall, with $\ddot{z} = -g$ and $g = 9.81m.s^{-2}$. The bounce is modeled with a single transition, which is triggered when the ball hits the ground ($z = 0$), and reverses the speed of the ball with a damping factor $0 < c \leq 1$. This model and its qualitative abstraction are shown in Figure 3.

Fig. 3. Hybrid automaton (left) and qualitative (right) automaton of the bouncing ball

The quantitative behavior of this model, as obtained in the Ptolemy II environment [5], is illustrated in Figure 4.

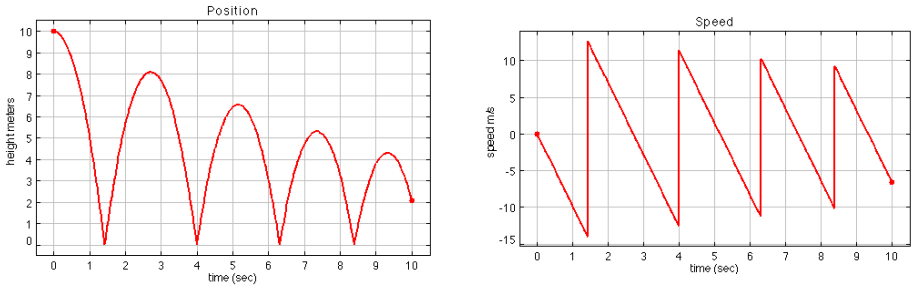

Fig. 4. Quantitative behavior of the bouncing ball

This kind of model is an abstraction of what really happens, because if the ball were really changing its speed instantaneously, there would be an exchange of a finite amount of energy in zero time between the ball and its environment, which corresponds to an infinite power.

6.4 Qualitative Simulation of the Raw Hybrid Model

When performing the qualitative simulation of this unphysical model using our model of execution, Diversity finds a dead-lock:

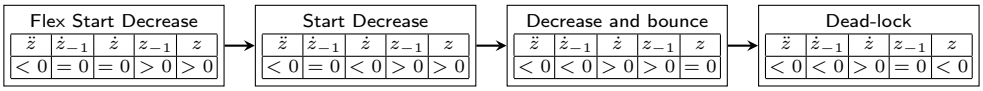

Fig. 5. Qualitative simulation of the raw model

This dead-lock happens because the model of the ball makes the first derivative change from negative to positive without going through zero, locking the state machine which models the first derivative in its "negative" state.

Our model of execution enforces the continuity and derivability of physical states, but the model of the ball violates these properties. However, we have to handle such models because that is the way engineers model systems, and quantitative simulation tools are able to handle them as shown in Figure 4.

6.5 Adjusting the Execution Model

A first solution is to adjust our execution model in order to allow such "unphysical" transitions. With this model of execution, the qualitative state changes to "Increase" after the bounce, then "Maximum", then to "Start Decrease" before cycling through the "Decrease" state, as shown in Figure 6. Therefore, the ball was able to bounce, and the maximum of its height was detected, but not the minimum that was reached during the bounce.

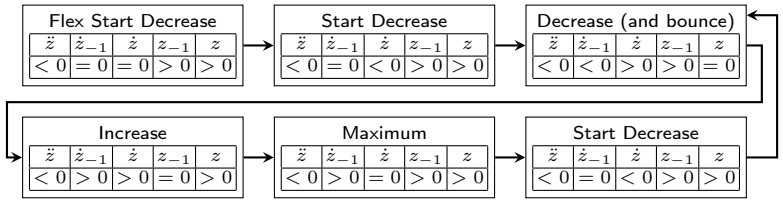

Fig. 6. Qualitative simulation with discontinuous transitions

The issue here is that the detection of the qualitative states relies on the continuity and derivability of the state variable. The discontinuous change of the velocity of the ball prevents the detection of the minimum.

6.6 Adjusting the model

Another solution is to adjust the model of the ball. However, we said that we must be able to handle models as engineers create them. In the case of the bouncing ball, the model can be automatically adjusted by replacing the bouncing transition by a series of transitions. The algorithm for this is as follows:

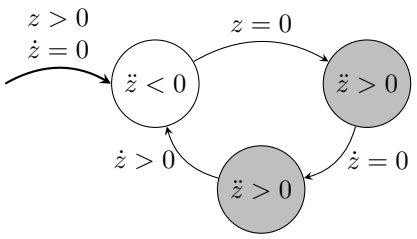

Fig. 7. Adjusted Model for the Bouncing Ball

- changing the derivative from < 0 to > 0 is illegal. The only legal path is to go from < 0 to $= 0$ and then from $= 0$ to > 0, so we replace the illegal transition by the sequence of these two transitions.
- changing the derivative from < 0 to 0 and from 0 to > 0 requires that the second derivative be positive, so we make the second derivative positive during the bounce (the second derivative can be discontinuous).

The resulting state machine is shown in Figure 7, where the additional states have a gray background. The first additional state, reached when $z = 0$, sets the second derivative to "Positive" because this is required to make the first derivative change from "Negative" to "Null". When the first derivative becomes null, the second additional state is reached, and when it becomes positive, as requested by the initial un-physical transition, we can go back to the free fall state of the ball, with a reversed velocity.

With this adjusted model of the bouncing ball and the physical model of qualitative simulation, we obtain the qualitative behavior shown in Figure 8.

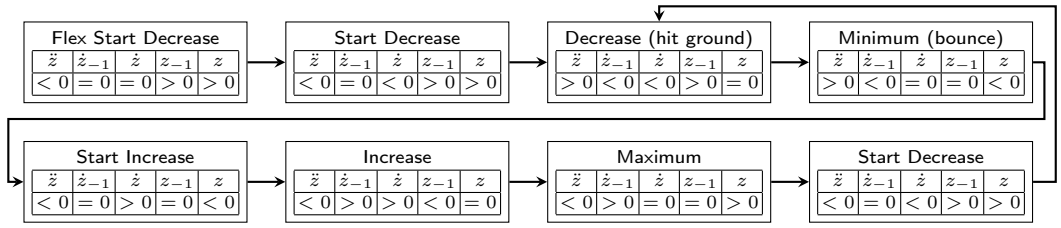

Fig. 8. Qualitative simulation with adjusted model

The qualitative simulation now correctly detects a minimum when the ball hits the ground and bounces, and a maximum when the ball reaches its highest position after having bounced. However, the original model was adjusted to match the model of computation of the qualitative behavior. How is the qualitative behavior we obtained related to the behavior of the original model?

7 Discussion about the Qualitative Model of Execution

In this qualitative simulation, we observe an unexpected artifact: the height of the ball above the ground becomes negative during the bounce. This comes from the fact that in the adjusted model, the first derivative takes some time to become null when the ball hits the ground, so the height goes from null to negative. Since we adjusted the model to make it more physical, we have to check if this artifact has some physical meaning.

The condition $z = 0$ for triggering the bounce corresponds to the surface of the ball hitting the ground. Therefore, the z state variable represents the height of the center of the ball above the ground. z becomes null at the moment when the ground starts pushing the ball upwards. Therefore, the $\ddot{z} > 0$ action we added to adjust the model represents the result of this pushing. This makes the ball slow down and stop ($\dot{z}$ goes from negative to null), but since $\dot{z}$ is still negative, z is still decreasing, so it goes from null to negative. For the physical system, this corresponds to the deformation of the ball, which converts kinetic energy into elastic tension in the ball (and heat if the collision is not elastic). When the ball is deformed against the ground, its center is below the position where it stood when the ball hit the ground, so z is negative.

We consider that the states and transitions added to adjust the model of the ball to the model of computation of the qualitative behavior are only reconstructing the physical behavior of the system that was abstracted in the original model. In the example of the bouncing ball, this reconstruction was possible in a single and systematic manner. However, this preliminary work did not allow us to verify that such adjustments are unique, or that there exists a minimal one for more complex systems.

When illustrating our approach with the example of the bouncing ball, we have only shown one of the qualitative behavior found by the Diversity tool. Among the other behaviors, some are physically possible. For instance, after bouncing on the ground, the ball can go up indefinitely, without reaching a maximum. This is possible if the ball reaches the escape velocity, but since we ignore the exact value of the velocity of the ball, we cannot tell, using qualitative simulation, whether the ball will reach a maximum or fly upwards forever.

Another possible behavior is more troubling: the ball can fall forever towards the ground without reaching it. This is indeed possible in our qualitative model of computation because it is only first order. We qualify only the variations of the state variable, not the variations of its derivative. Therefore, when the ball is falling, we only know that its height is decreasing, not that its velocity is increasing in magnitude. Without this information, it is possible that the ball goes slower and slower towards the ground, never reaching it, and the good thing is that the tool finds this behavior. We therefore plan to build a more complex, second order model of computation which will eliminate such physically impossible behaviors from the qualitative simulation.

8 Conclusion and future work

In this work, we have established a qualitative model of computation based on the first and the second derivative of the state variables to describe the behavior of hybrid systems without differential equations (with only a qualitative model of the equations). Our approach has been applied to the typical hybrid system example of the bouncing ball. We have shown that we can find a qualitative path that allows us to describe the whole behavior of the ball, respecting the continuity and derivability of the first and the second derivative. Using only the first two derivatives is enough to obtain the qualitative behavior of the variables in terms of variation, minimum and maximum. The tree of qualitative behaviors generated by Diversity can be used to prove reachability properties or to detect cycling behaviors.

To continue this work, we plan to enrich the model of computation with second order qualities (qualitative variations of the first derivative) in order to automatically eliminate some unphysical behaviors from the simulation results. Another track to explore is dealing with several bounded state variables, with constraints on their qualitative values or the qualitative value of their derivatives.

The main goal of this approach is to allow the qualitative analysis of hybrid models during the early stages of the design of a system. The abstraction process used to transform an hybrid automata into a qualitative discrete model, and the automatic transformation of this model into one which does obey the constraints of "physicality" encoded in our model of execution, are still very experimental. They can be considered as a kind of abstraction glue to combine the continuous and the discrete models, and its semantics needs a more precise definition to improve the confidence in the results of the qualitative simulation.

References

1. Bengtsson, J., Griffioen, W.D., Kristoffersen, K.J., Larsen, K.G., Larsson, F., Pettersson, P., Yi, W.: Automated verification of an audio-control protocol using uppaal. The Journal of Logic and Algebraic Programming 52, 163 – 181 (2002)
2. Bouyer, P., Colange, M., Markey, N.: Symbolic optimal reachability in weighted timed automata. In: Chaudhuri, S., Farzan, A. (eds.) Computer Aided Verification: 28th International Conference, CAV 2016, Toronto, ON, Canada, July 17-23, 2016, Proceedings, Part I. pp. 513–530. Springer International Publishing (2016)
3. Bozga, M., Daws, C., Maler, O., Olivero, A., Tripakis, S., Yovine, S.: Kronos: A model-checking tool for real-time systems. In: Ravn, A.P., Rischel, H. (eds.) Formal Techniques in Real-Time and Fault-Tolerant Systems: 5th International Symposium, FTRTFT'98 Lyngby, Denmark, September 14–18, 1998 Proceedings. pp. 298–302. Springer Berlin Heidelberg, Berlin, Heidelberg (1998)
4. Chutinan, A., Krogh, B.H.: Computational techniques for hybrid system verification. IEEE Trans. Automat. Contr. 48, 64–75 (2003)
5. Eker, J., Janneck, J.W., Lee, E.A., Liu, J., Liu, X., Ludvig, J., Sachs, S., Xiong, Y., Neuendorffer, S.: Taming heterogeneity - the ptolemy approach. Proceedings of the IEEE 91(1), 127–144 (2003)
6. Forbus, K.D.: Qualitative process theory. Artif. Intell. 24(1-3), 85–168 (Dec 1984)
7. Gallois, J.P., Lanusse, A.: Le test structurel pour la vérification de spécifications de systèmes industriels: L'outil agatha. In: Fiabilité & maintenabilité. Colloque national. pp. 566–574 (1998)
8. Gallois, J.P., Pierron, J.Y.: Qualitative simulation and validation of complex hybrid systems. In: 8th European Congress on Embedded Real Time Software and Systems (ERTS 2016). TOULOUSE, France (Jan 2016)
9. Henzinger, T.A., Ho, P.H., Wong-Toi, H.: Hytech: A model checker for hybrid systems. In: Grumberg, O. (ed.) Computer Aided Verification: 9th International Conference, CAV'97 Haifa, Israel, June 22–25, 1997 Proceedings. pp. 460–463. Springer Berlin Heidelberg, Berlin, Heidelberg (1997)
10. Kleer, J.D., Brown, J.S.: A qualitative physics based on confluences. Artificial Intelligence 24(1), 7 – 83 (1984)
11. Kuipers, B.J.: Qualitative simulation. Artif. Intell. 29(3), 289–338 (Sep 1986)
12. Nishida, T.: Artificial intelligence research in japan grammatical description of behaviors of ordinary differential equations in two-dimensional phase space. Artificial Intelligence 91(1), 3 – 32 (1997)
13. Rapin, N., Gaston, C., Lapitre, A., Gallois, J.P.: Behavioural unfolding of formal specifications based on communicating automata. In: Proceedings of first Workshop on Automated technology for verification and analysis (2003)
14. Sacks, E.P.: Automatic analysis of one-parameter planar ordinary differential equations by intelligent numeric simulation. Artificial Intelligence 48(1), 27 – 56 (1991)
15. Tiwari, A., Khanna, G.: Series of abstractions for hybrid automata. In: Tomlin, C.J., Greenstreet, M.R. (eds.) Hybrid Systems: Computation and Control: 5th International Workshop, HSCC 2002 Stanford, CA, USA, March 25–27, 2002 Proceedings. pp. 465–478. Springer Berlin Heidelberg, Berlin, Heidelberg (2002)
16. Yip, K.M.K.: KAM : a system for intelligently guiding numerical experimentation by computer. Artificial intelligence, MIT press, Cambridge, MA, London (1991)
17. Zhao, F.: Computational dynamics: Modeling and visualizing trajectory flows in phase space. Annals of Mathematics and Artificial Intelligence 8(3), 285–300 (1993)