Improved Infinity Filtering in Qualitative Simulation

A. C. Cem Say

Department of Computer Engineering
Boğaziçi University
Bebek 80815, İstanbul, Turkey
say@boun.edu.tr

Abstract

We present two modifications to the qualitative simulation algorithm QSIM, which improve its performance in reasoning tasks involving infinite values and infinite time. The first modification corrects an error in the multiplication filter which causes the algorithm to miss certain real solutions of the simulated equation. The soundness property, which is essential for qualitative simulators, is therefore restored to QSIM. The second modification augments the temporal attribute computation routine and results in better identification of infinite time intervals by the algorithm. This, in turn, helps our modified algorithm to successfully eliminate some additional spurious behaviors. Several famous examples from the qualitative reasoning literature are handled with increased predictive accuracy by the new algorithm.

Introduction

The QSIM algorithm (Kuipers 1994) finds solutions to input families of ordinary differential equations by simulating them using a qualitative representation. QSIM has become a "standard" in the qualitative reasoning community, because of the ready availability of its implementations and the strong theoretical background on which it is based. Several researchers, dealing with various aspects of the field, have independently developed new algorithms (see, for instance, (Weld 1988), (Weld 1990), (Say & Kuru 1996), and (Say & Kuru 1997)) based on QSIM, inheriting its appropriate theoretical properties.

We present two modifications to QSIM, which improve its performance in cases when the algorithm's output is supposed to contain infinite time intervals or variables reaching infinite limits. The first modification corrects an error which causes the algorithm to miss certain real solutions of the simulated equation. We show that "standard" QSIM is not sound because of this error, unlike our modified version. The second modification augments the temporal attribute computation routine, and results in better identification of infinite time intervals by the algorithm. This, in turn, helps our modified algorithm to successfully eliminate some additional spurious solutions. We show that several famous qualitative physics problems are handled with increased predictive accuracy by the new algorithm.

This paper is an enhanced version of (Say 1997c). In the following, we use the definitions and other terminological and notational details of (Kuipers 1994), which is the definitive reference for the standard QSIM algorithm.

Conservative Multiplication Filtering

Consider the naive physics problem depicted in Figure 1: We are pulling a very small rock attached to a string upward at constant speed. There is a lamppost of height h on the left, so the rock's shadow on the ground is moving to the right. The position of the shadow (X) and the height of the rock (Y) are variables in our system. The rock's takeoff point from the ground corresponds to 0 in both of these variables' quantity spaces. The foot of the lamppost is d units to the left of the takeoff point. The scene extends infinitely to the right. Light travels infinitely fast.

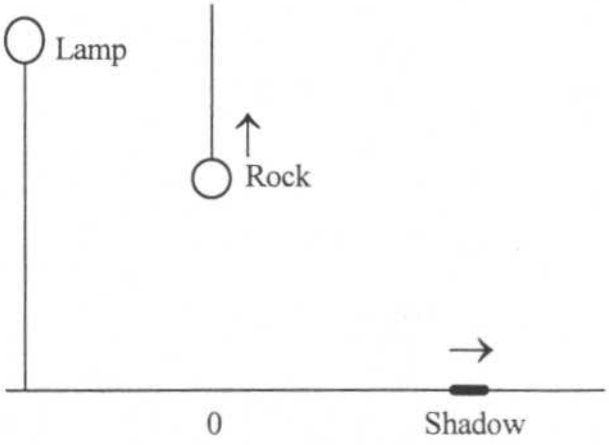

Figure 1: The rock-shadow system

The geometry of the situation imposes the following relation when $Y < h$:

$$\frac{X}{X+d} = \frac{Y}{h}. \qquad (1)$$

Rearranging, we get

$$X = \frac{d * Y}{h - Y}. \qquad (2)$$

The only qualitative behavior starting from Y=0 which is a solution of this model is presented in Table 1. (Note that we use "fixed" variables to represent the constants in the model.) Simulation terminates after t_1, since X is not reasonable over a larger interval.

time	t_0	(t_0, t_1)	$t_1 < \infty$
X	0, inc	$(0,\infty)$, inc	∞, inc
Y	0, inc	(0,h), inc	h, inc
RockSpeed	v, std	v, std	v, std
LampPos	d, std	d, std	d, std
LampHeight	h, std	h, std	h, std
(h–Y)	$(0,\infty)$, dec	$(0,\infty)$, dec	0, dec
(d*Y)	0, inc	$(0,\infty)$, inc	$(0,\infty)$, inc

TABLE 1. Behavior of the Rock-Shadow System

But if we simulated this model with the algorithm as described in (Kuipers 1994) from that initial state, there would be no behaviors in the output. The program would (incorrectly) state that the model and the initial state were inconsistent, and terminate.

The reason for this prediction failure is an error in the qualitative direction consistency check employed for *mult* constraints. QSIM requires ((Kuipers 1994), p. 56) that variables appearing in each constraint of the form `(mult A B C)` satisfy the sign relation

$$[A][B'] + [B][A'] = [C']. \qquad (3)$$

In our case, the constraint `(mult X (h-Y) (d*Y))` requires that

$$[X]\left[(h - Y)'\right] + \left[(h - Y)\right][X'] = \left[(d * Y)'\right]. \qquad (4)$$

Substituting the values at t_1, we obtain

$$[+][-] + [0][+] = [+]. \qquad (5)$$

When QSIM's sign multiplication table is employed, one gets

$$[-] + [0] = [+], \qquad (6)$$

which is not part of the sign addition relation, and the state at t_1 is therefore filtered out. Since the state at (t_0, t_1) has no other successors, it is deemed inconsistent. This information is propagated back to t_0, and therefore the simulation ends with an empty tree.

Clearly, the cause of the problem is the usage of the sign multiplication table, which unambiguously yields zero whenever one of the operands is zero, regardless of the other operand. When the nonzero operand has an infinite value, like the derivative of X in our example, this approach is wrong, and results in prediction failure.

In fact, this need for an exception to the multiplication table has been recognized in the design of the *magnitude* sign consistency check for the *mult* constraint of standard QSIM. That routine correctly allows the magnitude triple for X, (h-Y), and (d*Y) at t_1, even though it is of the form "positive*zero=positive". The problem with the direction check is that, unlike the magnitudes, only sign information is represented for the directions, which loses the important distinction between finiteness and infinity.

To restore the soundness property to QSIM, one can either drop the currently nonconservative direction check for *mult* altogether, or try to modify it so that it works correctly even in the presence of infinite derivatives. Since the former approach would dramatically increase the number of spurious predictions, we adopted the latter one.

In our modified algorithm, the constraint filter for `(mult A B C)` imposes the sign relation depicted in Equation (3) only when all terms have unambiguous sign values according to the magnitude and derivative information available at that stage. A proposed tuple automatically passes the direction consistency test when one of the following conditions is satisfied for at least one of the [P][Q′] terms in the imposed equation:

i. P is infinite and Q′ is zero, or,
ii. P is zero and Q′ is nonzero.

Only the tuples which do not satisfy these conditions are tested using Equation (3). This weaker filter is clearly conservative.

Note that condition (ii) above does not necessarily mean that Q′ is infinite. If we have magnitude information about Q′, we can, in some cases, compute an unambiguous sign for the term in which it appears, and invoke Equation (3) again to eliminate some spurious states. This is achievable if there are some other constraints in the model that allow one to write an expression for the magnitude of Q′, which can be evaluated by plugging in qualitative values of other variables. We used this idea to incorporate a new global filter to QSIM.

The *global multiplication direction filter*, which is applied to candidate time point states, visits every variable triple linked by a `(mult A B C)` constraint in the current operating region. The triples automatically pass the test if the magnitudes of both A and B are nonzero, since such combinations have either been thoroughly tested and approved in the local *mult* constraint, or they contribute at least one term which satisfies condition (i), and are therefore unfilterable. When, say, A, is zero, the algorithm checks if an expression for B's derivative's magnitude in terms of other system variables is available or not. (This expression search need be performed only once for all variables appearing in the first two places of *mult* constraints at the beginning of the simulation.) Some of the rules employed by this routine are presented in Table 2. See (Say 1998) for a more detailed discussion of these rules and the expression search.

Expression	if constraints of this form are in the model
$B' = D$	`(d/dt B D)`
$B' = -E$	`(minus B D)` and `(d/dt D E)`
$B' = R + S$	`(add P Q B)`, `(d/dt P R)`, and `(d/dt Q S)`
$B' = P * S + Q * R$	`(mult P Q B)`, `(d/dt P R)`, and `(d/dt Q S)`
$B' = \frac{dB}{dD} E$	`(M± B D)` and `(d/dt D E)`

TABLE 2. Expression Derivation Rules

If such an expression is available for the derivative's magnitude, and if its evaluation results in an unambiguously finite value, the algorithm inserts zero for the value of [A][B′]. If both terms on the left hand side of Equation (3) can be disambiguated, that equation is imposed on the variable triple to possibly filter it out.

The modified algorithm correctly presents the behavior in Table 1 as its single prediction for the rock-shadow system.

Improved Recognition of Infinite Intervals

The Technique

We will begin our explanation of our second modification to QSIM by presenting an example of one kind of spurious prediction that it eliminates. Consider the model (Fouché & Kuipers 1992) depicted in Table 3: The simulation involves a ball being thrown upward in air from ground level at t_0. The acceleration due to gravity is a negative constant, whereas the acceleration due to air friction is inversely related to the velocity of the ball.

When standard QSIM performs this simulation, it outputs the three behaviors shown in Tables 4-6. In Table 4, the ball has not reached its terminal velocity when it hits the ground. The difference between Tables 5 and 6 is whether it reaches the terminal velocity at or before the moment of hitting the ground. As mentioned in (Fouché & Kuipers 1992), only Table 4 is a real solution of this QDE; the other two behaviors are spurious.

Name	Explanation	
Y	ball height	
V	velocity (upward)	(d/dt Y V)
F	acceleration due to friction	((M- V F) (0 0) (-∞ ∞) (∞ -∞))
G	acceleration due to gravity	(constant G g < 0)
A	acceleration (upward)	(d/dt V A), (add G F A)

TABLE 3. The Ball-Friction Model

time	t_0	(t_0, t_1)	t_1	(t_1, t_2)	$t_2 < \infty$
Y	0, inc	(0,∞),inc	y*, std	(0,y*), dec	0, dec
V	(0,∞), dec	(0,∞), dec	0, dec	(-∞,0), dec	(-∞,0), dec
F	(-∞,0), inc	(-∞,0), inc	0, inc	(0,∞), inc	(0,∞), inc
G	g, std	g, std	g, std	g, std	g, std
A	(-∞,0), inc	(-∞,0), inc	(-∞,0), inc	(-∞,0), inc	(-∞,0), inc

TABLE 4. Prediction #1 for the Ball-Friction Model

time	t_0	(t_0, t_1)	t_1	(t_1, t_2)	$t_2 < \infty$
Y	0, inc	(0,∞),inc	y*, std	(0,y*), dec	0, dec
V	(0,∞), dec	(0,∞), dec	0, dec	(-∞,0), dec	v*, std
F	(-∞,0), inc	(-∞,0), inc	0, inc	(0,∞), inc	f*, std
G	g, std	g, std	g, std	g, std	g, std
A	(-∞,0), inc	(-∞,0), inc	(-∞,0), inc	(-∞,0), inc	0, std

TABLE 5. Prediction #2 (Spurious) for the Ball-Friction Model

time	t_0	(t_0, t_1)	t_1	(t_1, t_2)	t_2	(t_2, t_3)	$t_3 < \infty$
Y	0, inc	$(0,\infty)$,inc	y*, std	(0,y*), dec	(0,y*), dec	(0,y*), dec	0, dec
V	$(0,\infty)$, dec	$(0,\infty)$, dec	0, dec	$(-\infty,0)$, dec	v*, std	v*, std	v*, std
F	$(-\infty,0)$, inc	$(-\infty,0)$, inc	0, inc	$(0,\infty)$, inc	f*, std	f*, std	f*, std
G	g, std	g, std	g, std	g, std	g, std	g, std	g, std
A	$(-\infty,0)$, inc	$(-\infty,0)$, inc	$(-\infty,0)$, inc	$(-\infty,0)$, inc	0, std	0, std	0, std

TABLE 6. Prediction #3 (Spurious) for the Ball-Friction Model

In (Fouché & Kuipers 1992), the authors say that one can use QSIM's optional (and nonconservative) "analytic-function" constraint if one wishes fewer spurious predictions in this case. This constraint (Kuipers 1994) eliminates any behavior in which a variable is constant in at least one interval and has a nonzero derivative sometime else in the same behavior. When this constraint is switched on, the spurious behavior of Table 6 is no longer predicted, but the one in Table 5 is still in the output.

We now present a modification to the "infinity control" component of QSIM which improves the algorithm's ability to recognize infinite intervals. As a side-effect of this modification, both spurious behaviors are eliminated in the ball-friction example without the need for user intervention and at no risk of eliminating genuine behaviors.

Our modification is based on the following idea: Assume that you have a constraint (d/dt P Q) in your model. In the simulation output, you see an interval (t_i, t_{i+1}) in which Q is nearing zero. In the next time point, that is, t_{i+1}, Q arrives at zero. You also happen to know that there is a reasonable function between Q and P for their values in $(t_i, t_{i+1}]$, dQ/dP is negative for the values of P in (t_i, t_{i+1}), and it is finite at $P(t_{i+1})$. With this information, you can safely deduce that $t_{i+1} = \infty$.

The proof is simple: In (t_i, t_{i+1}), the chain rule allows us to write

$$\frac{dQ}{dP}\frac{dP}{dt} = \frac{dQ}{dt}. \qquad (7)$$

Using the d/dt constraint and separating the variables, one obtains

$$\frac{dQ}{dP}dt = \frac{dQ}{Q}, \qquad (8)$$

which, when solved, yields

$$F(t) = \ln|Q(t)| + C, \qquad (9)$$

where F is a function whose derivative is negative all through its domain, with the possible exception of the endpoint $t = t_{i+1}$, and C is an arbitrary constant. Taking the limit, we deduce that

$$\lim_{t \to t_{i+1}} F(t) = -\infty\,. \qquad (10)$$

Equation (9) and information about Q's value can be used to show that the function F is finite in (t_i, t_{i+1}). Using the fact that dQ/dP (that is, F′,) is finite at t_{i+1}, we can safely conclude that $t_{i+1} = \infty$. (This last step is based on the same kind of reasoning as standard QSIM's infinite time detection mechanism; see (Kuipers 1986, Kuipers 1994).) •

To make use of this opportunity of detecting infinite intervals, we have made the following modifications to QSIM:

1. Before the start of simulation, the algorithm checks the constraint model to see if an M- relationship can be deduced (using the simplifying techniques of (Kuipers 1984)) between pairs of variables which occur together in (d/dt P Q) constraints. Since M- functions are not guaranteed to have finite derivatives at the endpoints of their ranges, variable range and corresponding value information about each such derived relationship are additionally checked to see if the point where Q = 0 lies in the interior of the obtained monotonic function's domain. If this restriction is also satisfied, the variable Q is noted for future use.
2. The timepoint labeling routine has been modified so that the temporal attribute $=\infty$ can be asserted for timepoint t_{i+1} if a derivative variable Q that has been noted in part (1) above is zero at t_{i+1} and nonzero in (t_i, t_{i+1}).
3. The temporal attribute consistency filter has been augmented so that, in every proposed system state at timepoint t_{i+1}, the following constraint has to be satisfied for every derivative variable Q noted in part (1):

$$t_{i+1} < \infty \rightarrow [Q(t_{i+1})=0 \rightarrow Q(t_i, t_{i+1})=0]$$

The justification for the last implication in the above constraint is that Q may have been steady at zero since the initial state; there is no inconsistency in such a case.

Returning to the ball-friction example, the algorithm performs the deductions shown in Table 7 at the initial stage. The corresponding value structure of the derived M-relationship between V and A indicates that dA/dV is certain to be finite at the point where A=0, so A is recorded as a potential infinity indicator. Note that generalized CV tuples, possibly including interval as well as point values (Say & Kuru 1993), are derived and used during the identification of the M-.

From the constraints	**with CV tuples**	**Derive the constraint**	**with CV tuples**
(constant G)	G < 0	(M+ F A)	(-∞ -∞) (∞ ∞) ((0 ∞) 0)
(add G F A)			
(M+ F A)	(-∞ -∞) (∞ ∞) ((0 ∞) 0))	(M- V A)	(∞ -∞) (-∞ ∞) ((-∞ 0) 0)
(M- V F)	(0 0) (-∞ ∞) (∞ -∞) ((-∞ 0) (0 ∞)) ((0 ∞) (-∞ 0))		

TABLE 7. Derivation of M-(V,A) in the Ball-Friction Model

When the states at time point t_2 are being generated for the behaviors in Tables 5 and 6, two things may happen:

a. The timepoint labeling routine may check A in the manner described above, and assert $t_2 = \infty$. In this case, the temporal attribute consistency filter will detect that variable Y has a finite value and a nonzero derivative in this state, which is inconsistent if $t = \infty$, and the state will be filtered out.
b. Depending on the order in which the variables appear in the input, the timepoint labeler may assert $t_2 < \infty$, based on the fact that Y is finite and changing. In this case, the temporal attribute consistency filter will delete the state, since A, which is not supposed to reach zero in a finite time, has obtained that value.

In both cases, the modified algorithm eliminates all spurious behaviors successfully.

Further Examples

Bathtub. A liquid tank (or "bathtub") with constant inflow, where the possibility of overflow is deliberately overlooked, can be modeled with the following constraints:

```
(constant Inflow i*)
((M+ Amount Outflow) (0 0) (∞ ∞))
(d/dt Amount Netflow)
(add Netflow Outflow Inflow)
```

If the tank is specified to be empty at the initial state, standard QSIM will compute the behavior shown in Table 8 for this model. As can be seen, no temporal attribute has been assigned to the equilibrium instant t_1. Our modified algorithm would derive the implicit constraint M-(Amount, Netflow), check its corresponding value structure, and correctly label t_1 to be infinite.

time	t_0	(t_0, t_1)	t_1
Amount	0, inc	(0,∞),inc	a*, std
Outflow	0, inc	(0,∞),inc	out*, std
Netflow	(0,∞),dec	(0,∞),dec	0, std
Inflow	i*, std	i*, std	i*, std

TABLE 8. Standard QSIM Prediction for the Bathtub with Nonzero Inflow

This additional precision comes handy in the analysis of the two-tank cascade system, which is obtained by placing another tank below this tank's hole. As explained in (Kuipers 1994), QSIM predicts a spurious behavior in which the upper tank reaches equilibrium before the lower tank during the simulation of this system, unless the analytic function constraint is employed. The modified infinity control routine would, of course, eliminate this problem.

What if there was no inflow and we started with some liquid in the single tank? In this case, we cannot deduce that the tank empties at $t = \infty$. The algorithm still derives an M- constraint between Amount and Netflow, but the corresponding values are now (0 0) and (∞ -∞), and hence there is no guarantee that dNetflow/dAmount would have a finite limit at Netflow = 0. In fact, it is easy to show that this limit is infinite for all real-world tanks. (Only a proper subset of the functions abstracted by the M+(Amount, Outflow) constraint are realizable; see (Say 1997a) for the details.)

Heat Exchanger. The heat exchanger (Figure 2) model to be used in this example is from (Weld 1988). There is cold water in the bath shown as the box in the figure. Hot liquid enters from one end of the pipe and leaves, cooler because of the heat flow, from the other end. The QSIM model is shown in Table 9. The entry end of the pipe corresponds to the negative landmark x^* in the quantity space of variable X. The exit end is 0. The surplus heat Q has a positive value at the start of the simulation, and becomes 0 when thermal equilibrium is reached. (Note that the physics has been somewhat simplified for the sake of brevity.)

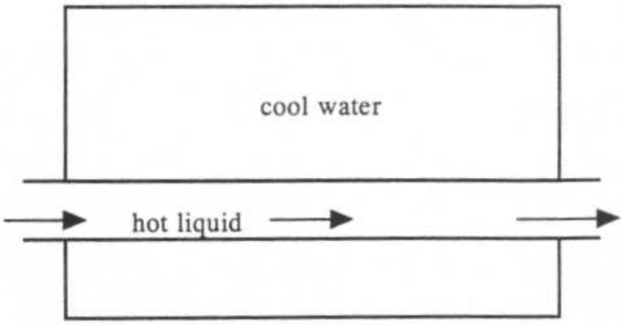

Figure 2: The heat exchanger

Name	Explanation	
X	position of liquid in the pipe	
V	velocity of the flowing liquid	(d/dt X V) (constant V v > 0)
Q	surplus heat of liquid	
K	thermal conductivity	(constant K k < 0)
F	the heat flow in the liquid	(d/dt Q F) (mult Q K F)

TABLE 9. The Heat Exchanger Model

There are three different behaviors, determined by whether the heat flow stops when the unit volume of liquid that we are interested in is in the pipe, and if so, where. (Tables 10-12)

time	t_0	(t_0, t_1)	$t_1 < \infty$
X	x^*, inc	$(x^*, 0)$, inc	0, inc
V	v, std	v, std	v, std
Q	q^*, dec	$(0, q^*)$, dec	0, std
K	k, std	k, std	k, std
F	f^*, inc	$(f^*, 0)$, inc	0, std

TABLE 10. Behavior #1 in heat exchanger simulation

time	t_0	(t_0, t_1)	$t_1 < \infty$
X	x^*, inc	$(x^*, 0)$, inc	0, inc
V	v, std	v, std	v, std
Q	q^*, dec	$(0, q^*)$, dec	$(0, q^*)$, dec
K	k, std	k, std	k, std
F	f^*, inc	$(f^*, 0)$, inc	$(f^*, 0)$, inc

TABLE 11. Behavior #2 in heat exchanger simulation

In this case too, the implicit constraint M-(Q,F) leads to the conclusion that variable F can reach zero only at $t = \infty$, and behaviors #1 and #3 are eliminated by the infinity filter. It turns out that, according to this model, we can never cool our liquid down to the temperature of the coolant.

U-tube. Another famous qualitative physics example is the U-tube, consisting of two tanks at the same level connected by a pipe. (Figure 3) The model is presented in Table 13. Again overlooking the possibility of overflow, if we start with a state in which there is liquid in tank A and tank B is empty, the single QSIM prediction depicts the system reaching equilibrium at a time point for which there is no attribute information.

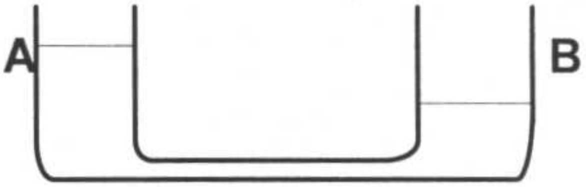

Figure 3: U-tube in operating region NORMAL

time	t_0	(t_0, t_1)	t_1	(t_1, t_2)	$t_2 < \infty$
X	x^*, inc	$(x^*, 0)$, inc	$(x^*, 0)$, inc	$(x^*, 0)$, inc	0, inc
V	v, std	v, std	v, std	v, std	v, std
Q	q^*, dec	$(0, q^*)$, dec	0, std	0, std	0, std
K	k, std	k, std	k, std	k, std	k, std
F	f^*, inc	$(f^*, 0)$, inc	0, std	0, std	0, std

TABLE 12. Behavior #3 in heat exchanger simulation

Name	Explanation	
A	amount of liquid in tank A	
B	amount of liquid in tank B	
total	total amount of liquid	(add A B total) (constant total tot* > 0)
pA	pressure at bottom of tank A	((M+ A pA) (0 0) (∞ ∞))
pB	pressure at bottom of tank B	((M+ B pB) (0 0) (∞ ∞))
pAB	pA - pB	(add pB pAB pA)
fAB	flow from A to B	(d/dt B fAB) ((M+ pAB fAB) (-∞ -∞) (0 0) (∞ ∞))
fBA	flow from B to A	(d/dt A fBA) (minus fAB fBA)

TABLE 13. The U-tube Model

From the constraints	with CV tuples	Derive the constraint	with CV tuples
(constant total)	tot* > 0		
(add A B total)		(M- A B)	(0 (0 ∞)) ((0 ∞) 0)
(M- A B)	(0 (0 ∞)) ((0 ∞) 0)		
(M+ A pA)	(0 0) (∞ ∞)	(M- B pA)	(0 (0 ∞)) ((0 ∞) 0)
(M- B pA)	(0 (0 ∞)) ((0 ∞) 0)		
(add pB pAB pA)			
(M+ B pB)	(0 0) (∞ ∞)	(M- B pAB)	(0 (0 ∞)) ((0 ∞) (-∞ 0))
(M- B pAB)	(0 (0 ∞)) ((0 ∞) (-∞ 0))		
(M+ pAB fAB)	(-∞ -∞) (0 0) (∞ ∞)	(M- B fAB)	(0 (0 ∞)) ((0 ∞) (-∞ 0))

TABLE 14. Derivation of M-(B,fAB) in the U-tube Model

Table 14 shows how our algorithm derives the M-(B, fAB) constraint used to deduce that fAB can reach zero only at $t = \infty$. The accordingly improved behavior prediction is presented in Table 15.

time	t_0	(t_0, t_1)	$t_1 = \infty$
A	(0, ∞), dec	(0, ∞), dec	a*, std
B	0, inc	(0, ∞), inc	b*, std
total	tot*, std	tot*, std	tot*, std
pA	(0, ∞), dec	(0, ∞), dec	pa*, std
pB	0, inc	(0, ∞), inc	pb*, std
pAB	(0, ∞), dec	(0, ∞), dec	0, std
fAB	(0, ∞), dec	(0, ∞), dec	0, std
fBA	(-∞, 0), inc	(-∞, 0), inc	0, std

TABLE 15. Improved Prediction for U-tube System

Summary and Future Work

We presented two modifications to the QSIM algorithm which help it perform better when reasoning about infinite values and times. The first modification rids QSIM of an error in the qualitative direction filter of the *mult* constraint and therefore restores the soundness property to the algorithm. The second modification improves the algorithm's ability to recognize subsystems that can reach quiescence only in infinite time, and hence helps in the elimination of a class of spurious behaviors that are caused when this information is not used.

Both the derivation of expressions for derivative magnitudes mentioned in the section about the new multiplication filter, and the recognition of implicit M-constraints explained in the subsequent section are achieved using a rudimentary algebraic processing component that has been incorporated into the algorithm. Currently, this component is not able to identify M-relationships or derivative expressions "embedded" in models in a relatively less obvious manner. We plan improving the algebraic capabilities of this procedure in the near future. A general-purpose algebraic manipulator that could work on QSIM models would be useful for several other applications as well, see (Fouché & Kuipers 1992) and (Say 1997b) for examples.

Acknowledgments

I thank H. Levent Akın and A. Taylan Cemgil for their help in this work.

References

Fouché. P., and Kuipers, B. J. 1992. Reasoning About Energy in Qualitative Simulation. *IEEE Transactions on Systems, Man, and Cybernetics* 22(1):47-63.

Kuipers, B. J. 1984. Commonsense Reasoning About Causality: Deriving Behavior from Structure. *Artificial Intelligence* 24:169-204.

Kuipers, B. J. 1986. Qualitative Simulation. *Artificial Intelligence* 29:289-338.

Kuipers, B. J. 1994. *Qualitative Reasoning: Modeling and Simulation with Incomplete Knowledge*. Cambridge, Mass.: The MIT Press.

Say, A. C. C. 1997a. Limitations Imposed by the Sign-Equality Assumption in Qualitative Simulation. In *Proc. Eleventh Int. Workshop on Qualitative Reasoning*, Cortona, Italy. 165-173.

Say, A. C. C. 1997b. Numbers Representable in Pure QSIM. In *Proc. Eleventh Int. Workshop on Qualitative Reasoning*, Cortona, Italy. 337-344.

Say, A. C. C. 1997c. Improved Reasoning About Infinity in Qualitative Simulation. In *Proc. Twelfth Int. Symposium on Computer and Information Systems*, Antalya, Turkey. 36-43.

Say, A. C. C. 1998. L'Hôpital's Filter for QSIM. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 20(1):1-8.

Say, A. C. C., and Kuru, S. 1993. Improved Filtering for the QSIM Algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 15(9):967-971.

Say, A. C. C., and Kuru, S. 1996. Qualitative System Identification: Deriving Structure from Behavior. *Artificial Intelligence* 83:75-141.

Say, A. C. C., and Kuru, S. 1997. Postdiction Using Reverse Qualitative Simulation. *IEEE Transactions on Systems, Man, and Cybernetics* 27(1):84-95.

Weld, D. S. 1988. Comparative Analysis. *Artificial Intelligence* 36:333-373.

Weld, D. S. 1990. Exaggeration. *Artificial Intelligence* 43:311-368.