Sound and Complete Qualitative Simulation Is Impossible

A. C. Cem Say and H. Levent Akın

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

Abstract

State-of-the-art qualitative simulators (for instance, QSIM) are known to be *sound*; no trajectory which is the solution of a concrete equation matching the input can be missing from the output. A simulator which is seen to be *incomplete*, that is, which produces a spurious prediction for a particular input, can usually be augmented with an additional filter which eliminates that particular class of spurious behaviors, and the question of whether a simulator with purely qualitative input which *never* predicts spurious behaviors can ever be achieved by adding new filters in this way has remained unanswered until now. We prove that such a sound and complete qualitative simulation algorithm does not exist.

Introduction

State-of-the-art qualitative simulators (Kuipers 1994) are known to be *sound*; no trajectory which is the solution of a concrete equation matching the input can be missing from the output. A simulator which is seen to be *incomplete*, that is, which produces a spurious prediction for a particular input, can usually be augmented with an additional filter which eliminates that particular class of spurious behaviors, and the question of whether a simulator with purely qualitative input which *never* predicts spurious behaviors can ever be achieved by adding new filters in this way has remained unanswered until now. (Kuipers 2001) In this paper, we prove that such a sound and complete qualitative simulation algorithm does not exist.

The Question

We start by clarifying the nature of the contribution of this paper. The fact that the output set of a qualitative simulator (QSIM) is guaranteed to include the solution of every ODE matching its input (the *soundness* property) was proven in (Kuipers 1986). In the same paper, Kuipers also showed that the version of QSIM described there had the *incompleteness* property, by demonstrating that the qualitative simulation of a frictionless mass-spring oscillator predicts unrealizable (*spurious*) behaviors, where the amplitude decreases in some periods and increases in others.

The latest formulation of the QSIM algorithm (Kuipers 1994) provides a possibility of plugging additional routines (*global filters*) into the simulator in order to eliminate system states which can not be deleted using the basic local constraint filtering method, but which can still be shown to be inconsistent with the mathematical model and the behavior prefix that has been created up to that point. For each different class of spurious predictions that has been identified until now, it has been possible to specify a global filter which would eliminate that behavior family from the algorithm's output when incorporated to the software. This process of improvement has had the following structure: In order to be able to say that a particular predicted behavior is spurious, and therefore suitable for elimination from the simulator output without forsaking the soundness property, one first proves that that behavior is mathematically inconsistent with the simulated model and starting state. For instance, the aforementioned spurious oscillations of the frictionless mass-spring system can be shown to violate a conservation constraint that follows directly from the structure of the input equations. But this proof can itself be seen as the specification of a filter routine which would eliminate exactly the set of behaviors that violate the "law" that it establishes. The kinetic energy constraint (Fouché and Kuipers 1992) is a global filter which has been developed in this fashion to eliminate the class of spurious predictions exemplified by the ones about the mass-spring system. See (Struss 1988), (Struss 1990), (Say and Kuru 1993), (Kuipers 1994), (Say 1998), (Könik and Say 1998), and (Say 2001) for discussions on the causes of some specific types of spurious solutions, and other examples of spurious behavior classes being discovered simultaneously with their "cures."

Since there is no limit to the number of global filters that can be added to the algorithm, and to the mathematical sophistication that can be involved in their derivation, one can legitimately ask the following questions:

- Is the *present* version of the algorithm (i.e. the one obtained by the incorporation of all the global filters specified until now) complete or incomplete?
- Is it possible to obtain a sound and complete version of QSIM by adding a finite number of additional global filters to the present version?
- Regardless of its internal details, does there exist a sound and complete qualitative simulator whose input and

output vocabularies are identical to those of the "pure" QSIM algorithm?

This paper provides a negative answer to these questions.

Preliminaries

Our proof makes use of the previously established fact (Say 1997) that all members of a very rich set of numbers (the radical extension of a superset of the rational numbers) can be represented exactly using the input vocabulary of a pure qualitative simulator.

QSIM-Representable Numbers

A real number r is said to be *QSIM-representable* if there exists a set of QSIM variable quantity spaces and constraints, from which r's equality to a particular landmark symbol p in that set can be unambiguously deduced. 0 is QSIM-representable because of the existence of the standard landmark 0. The first five rows of Table 1 demonstrate the facts that the numbers 1, -1, 2, and 1/2 are QSIM-representable because we can deduce that the landmarks b_1, c_1, e_1, and g_1 respectively equal these numbers by using the information in that table. (All user-defined landmarks except c_1 are positive in this table.)

CONSTRAINTS	CORRESPONDENCES
$A(t) = a_1$	
$A(t) * B(t) = A(t)$	$a_1 * b_1 = a_1$
$B(t) + C(t) = D(t)$	$b_1 + c_1 = 0$
$B(t) + B(t) = E(t)$	$b_1 + b_1 = e_1$
$E(t) * G(t) = B(t)$	$e_1 * g_1 = b_1$
$\frac{d}{dt}H(t) = J(t)$	
$\frac{d}{dt}K(t) = L(t)$	
$K(t) = f(H(t)),\ f \in M^+$	$f(0) = 0,\ f(\infty) = k_1$
$H(t) * H(t) = N(t)$	
$B(t) = b_1$	
$N(t) + B(t) = P(t)$	
$P(t) * L(t) = J(t)$	
$E(t) * K(t) = Q(t)$	$e_1 * k_1 = q_1$

TABLE 1. QSIM model with exact numerical information

It is easy to see that any rational number is QSIM-representable. Say (1997) showed that infinitely many irrationals, including some transcendentals, are also QSIM-representable. Our proof of qualitative simulation algorithms' intrinsic incompleteness rests on the fact that π can be represented exactly by QSIM landmarks, so we demonstrate the QSIM-representability of π in the last eight rows of Table 1: Rows 9-11 establish the equivalence

$$P = H^2 + 1.$$

The multiplication constraint between the derivatives, when rearranged, simply means that

$$\frac{dK/dt}{dH/dt} = \frac{1}{H^2+1}.$$

Since, by the definition of the M^+ constraint, K is a differentiable function of H, the chain rule for the derivative of a composite function allows us to replace the left hand side by dK/dH:

$$\frac{dK}{dH} = \frac{1}{H^2+1}.$$

Integrating, we get

$$K = f(H) = \arctan H + \mathrm{r},$$

where r is a constant.

The correspondence $f(0) = 0$ enables us to determine that r is zero, so

$$K = f(H) = \arctan H.$$

Using the correspondence $f(\infty) = k_1$, and the fact that $\lim_{x\to\infty} \arctan x = \pi/2$, we conclude that q_1 (which is shown to equal $2 * k_1$ in the last row) is π.

Of course, the indirect way in which we claim to "represent" QSIM-representable numbers is fundamentally inadequate for many purposes. For instance, if one writes a model in which the two landmarks r and s are respectively shown to be equal to the numbers 3 and π, and requires the computation of the sign of the expression $(r - s)$, the present version of the algorithm would not be able to eliminate the (clearly wrong) signs [0] and [+] from consideration. Our certainty about the sign of this expression comes from our *numerical* knowledge, which is exactly the kind of information that *qualitative* simulation programs are supposed to eschew. Unfortunately, the power of their input vocabulary, demonstrated above, means that qualitative simulators must "dirty their hands" with quantitative details to avoid this type of spurious computation.

Computation of Approximations to Irrationals

As part of the buildup for our demonstration of a global state filter that would require unbounded memory for correct execution in the next section, we now present a brief review of elementary facts about the calculation of approximations to irrational numbers by computers. (Press et al. 1988) Since irrational numbers require infinite precision for exact digital representation, even the best numerical estimate for an irrational x produced by the best possible computer in the maximum allowable time is guaranteed to be *not* equal to x. It is important to see that the output of a numerical method for computing an irrational is actually an interval (q_l, q_u) bounded by two rational numbers, which are exactly representable by the computer. (Even if the numerical algorithm outputs a single floating-point number, the lack of information about the values of the uncomputed digits means that there are multiple numbers of higher precision that are "covered" by the computed result.) This interval is guaranteed to contain the irrational under consideration, but it is not possible to

determine the irrational's "position" (e.g. whether it is in the lower or higher half) within the interval without performing further computation.

Spurious States Requiring Unbounded Working Memory for Elimination

We now examine global filtering from a resource requirement perspective. There can be two types of global filters: (Kuipers 1994) *State filters* access only the information in the current simulated state when attempting to find a mathematical inconsistency, whereas *behavior filters* need to have global information about the entire behavior prefix starting from the initial state and terminating at the current state. (We do not consider the "No Change" filter, which requires to see the *previous* simulation state as well as the current one, and which does not eliminate any spurious predictions anyway, to be a filter in this discussion.) All the global filters that we listed earlier are behavior filters, and there are surprisingly few state filters, in fact, the infinity and quiescence checks, and the higher-order derivative constraint are the only ones that we are aware of.

If one considers the realistic case where the memory that a global filter is allowed to use is finite, it is easy to see that behavior filters will run into practical trouble when the lengths of the computed behaviors get large. We will now demonstrate a novel type of spurious behavior, which can theoretically be eliminated by a state filter, *only if the filter has access to unbounded amounts of memory*. In the setup to be described, it will be possible to detect that a member of a particular set of system states computed by QSIM has to be inconsistent, but it will not be possible to determine *which* particular state is the culprit. To preserve the soundness property, no individual state can be deleted until specifically proven to be spurious, so the problematic state (and the behavior postfixes emanating from it) will have to remain in the simulation output.

One builds the qualitative constraint model which will exhibit this type of spurious behavior in the following manner: Pick a QSIM-representable irrational number x and add the combination of variables and constraints that would result in the deduction that landmark p_x of variable P is equal to x to the (initially empty) model. Proceed to employ the highest-precision numerical method that would be allowed by your qualitative simulator's computation budget to calculate the smallest possible interval (q_l, q_u) containing x. Add the constraints and variables which would lead to the deduction that landmark q_m of variable Q is equal to $(q_l + q_u)/2$ to the model. Finally, add the constraint $S(t) + Q(t) = P(t)$, where S is a variable which does not appear in any other constraint, to the model.

The problem occurs when $P(t_i) = p_x$ and $Q(t_i) = q_m$. The simulator has the task of deleting all incorrect sign assignments for the magnitude of $S(t_i)$. Since both P and Q are at point values, it is clear that only one sign assignment for S is supposed to survive. (The difference of two reals is a single real, with an unambiguous sign.) It is also obvious that $S(t_i) \neq 0$, since p_x is an irrational and q_m is a rational. But there is no way for the algorithm to decide whether $S(t_i) > 0$ or $S(t_i) < 0$, as described in the previous section, since we assume that (q_l, q_u) is the narrowest interval within which x can be sandwiched using the available resources. Even if those resources are increased to come up with a narrower interval, the mid-point of that interval would again be a rational number, and a QSIM model exhibiting the same problem could be built with the method described above. Whenever $P(t_i) = p_x$ and $Q(t_i) = q_m$, the behavior tree would have to branch on the choice of sign ([+] or [-]) for $S(t_i)$. One of these subtrees is sure to be spurious, but both of them have to appear in the output, since it is impossible to decide which one to prune.

Discussion

The fact that qualitative simulators usually give multiple behavior predictions in their output is explained by the incompleteness of the information they take as input. In a certain sense, the "incurable" class of spurious behaviors depicted here is also caused by lack of complete information, this time about the full expansion of the irrational under consideration. The difference between a "legitimate" branching in the behavior tree (for instance, the three alternative behaviors produced during the pure qualitative simulation of an initially empty bathtub of finite capacity with constant inflow) and a spurious one is that each non-spurious behavior matches the solution of at least one quantitative equation that matches the simulator input, whereas a spurious behavior does not have this property. In our construction of the previous section, it is evident that the program's input represents a single quantitative equation which has a single solution, and so a simulator with the completeness property would have to output a single behavior.

Although the (clearly artificial) model constructed above has no obvious dynamics, it can of course be "plugged into" any other QDE and would cause the same spurious branchings.

It is well known that certain algebraic or differential equations have no closed form solution. The kind of comparison problem at the heart of the argument of the preceding section can also be posed by taking such an equation known to possess no non-numerical solution, writing it in the QSIM format, and asking for the difference between a QSIM-representable actual root of this equation and its best numerical estimate. (As exemplified in Table 1 and (Say 1997), it is possible to represent arbitrarily complicated differential equations among the dependent variables using the QSIM vocabulary.)

A model that represents rationals of such high precision as the one in our argument is bound to contain huge numbers of constraints and variables. Our argument assumes that the memory required for storing the model itself is unbounded, and the filters do not have write access there. It is possible to "swamp" the higher-order derivative constraint, which has to store lists of names and formulas for variables that can chatter, to run out of memory by

providing a sufficiently large input model as well. (However, it may be remarked that the HOD constraint needs to access qualitative direction information of the previous state, and therefore is not a "single-state" filter.)

The Answer

Of course, in order to claim that no *algorithm* for sound and complete qualitative simulation exists, one should drop the finite resource assumption used in the argument of the previous section. We now proceed with a demonstration which does not rest on this supposition. Our approach is inspired by a remark of Kuipers (1993) to the effect that an eventual qualitative simulation incompleteness theorem might involve showing that the QSIM language is rich enough to include the transcendental functions.

Hilbert's Tenth Problem

In 1900, David Hilbert announced a list of 23 unsolved problems as a challenge to the mathematicians of the 20th century. The tenth problem can be stated as follows:

> "Find an algorithm for deciding whether a given multivariate polynomial with integer coefficients has integer solutions."

In 1970, Yuri Matiyasevich proved that no such algorithm exists. (Matiyasevich 1993)

In the remainder of this paper, we show that a sound and complete qualitative simulator, if it existed, could be used to solve the unsolvable decision problem formulated above. The idea is straightforward: Let $P(x_1, x_2, \ldots, x_n)$ be the given polynomial. As stated in (Moses 1971), $P = 0$ has integer solutions if and only if

$$\sum_{i=1}^{n} \sin^2 \pi.x_i + P^2(x_1, x_2, \ldots, x_n) = 0$$

has real solutions. If we can show that the equation above can be expressed in a collection of QSIM states for any P, the task of determining whether all those states are spurious or not turns out to be equivalent to the task of determining whether P has integer solutions.

One goes about expressing the equation above in QSIM in the following way: Start with an empty model. Each of the variables x_i of P will be represented by a corresponding QSIM variable X_i. Let c_j be the integer coefficients of the terms of P. As mentioned earlier, all integers are QSIM-representable; add the finitely many variables and *add* constraints that are required to equate each c_j to a corresponding landmark l_{j1} of a QSIM variable L_j. Each term $term_j$ of P is a product of integer powers of some of the variables with the coefficient c_j. Use the necessary number of *mult* constraints and intermediate QSIM variables to equate each $term_j$ to a corresponding QSIM variable T_j. P is just the sum of the $term_j$'s; use the necessary number of *add* constraints and intermediate QSIM variables to equate it to a QSIM variable P_Q. Add the single *mult* constraint necessary to equate the QSIM variable *P2* to P^2.

The QSIM-representability of π was demonstrated earlier, and equating each $\pi.x_i$ to a corresponding QSIM variable PX_i is a trivial matter of adding n new *mult* constraints, and the relevant rows of Table 1, to the model.

Clearly, we just need a way of representing the existence of the sine function among two given variables in the QSIM notation to accomplish the task of building the required expression.

The Sine Function Is QSIM-Representable

A function f is *QSIM-representable* if there exists a QSIM model which includes both x and $f(x)$ as variables, and the exact relationship denoted by f can be deduced from the information in the model. (Say 1996) We already saw (Table 1) that the inverse tangent function is QSIM-representable.

In Table 2, the inverse sine function can be deduced to exist between the variables G and H, using the method demonstrated in conjunction with Table 1: When the appropriate substitutions are made, the last row means that

$$\frac{dH}{dG} = \frac{1}{\sqrt{1-G^2}}.$$

Integration, and the substitution of $f(0) = 0$, leads one to conclude that

$$H = f(G) = \arcsin G.$$

CONSTRAINTS	CORRESPONDENCES
$A(t) = a_1$	
$A(t) * B(t) = A(t)$	$a_1 * b_1 = a_1$
$B(t) + C(t) = D(t)$	$b_1 + c_1 = 0$
$C(t) + E(t) = G(t)$	$c_1 + 0 = g_1$
$B(t) + H(t) = G(t)$	$b_1 + 0 = g_2$
$\frac{d}{dt}G(t) = J(t)$	
$\frac{d}{dt}H(t) = K(t)$	
$H(t) = f(G(t)),\ f \in M^+$	$f(0) = 0$
$G(t) * G(t) = L(t)$	
$B(t) = b_1$	
$L(t) + N(t) = B(t)$	
$P(t) * P(t) = N(t)$	$P(t) \geq 0$
$P(t) * K(t) = J(t)$	

TABLE 2. Constraint set of operating region SIN_ORIG

Unlike the inverse tangent, which is defined over the entire real number set, the domain of inverse sine is between the landmarks g_1 and g_2 (shown, respectively, to equal –1 and 1,) of variable G.

When one considers the function from H to G in Table 2, one sees that this is the small portion of $G = \sin H$ in the domain $[-\pi/2, \pi/2]$. We want to construct a model that

represents this relationship over the entire real line, so we build the constraint sets shown in Tables 3 and 4.

CONSTRAINTS	CORRESPONDENCES
$A(t) = a_1$	
$A(t) * B(t) = A(t)$	$a_1 * b_1 = a_1$
$B(t) + C(t) = D(t)$	$b_1 + c_1 = 0$
$C(t) + E(t) = G(t)$	$c_1 + 0 = g_1$
$B(t) + H(t) = G(t)$	$b_1 + 0 = g_2$
$\frac{d}{dt}G(t) = J(t)$	
$\frac{d}{dt}H(t) = K(t)$	
$H(t) = f(G(t)),\ f \in M^-$	
$G(t) * G(t) = L(t)$	
$B(t) = b_1$	
$L(t) + N(t) = B(t)$	
$Q(t) * Q(t) = N(t)$	$Q(t) \leq 0$
$Q(t) * K(t) = J(t)$	

TABLE 3. Constraint set of operating region SIN_MINUS

CONSTRAINTS	CORRESPONDENCES
$A(t) = a_1$	
$A(t) * B(t) = A(t)$	$a_1 * b_1 = a_1$
$B(t) + C(t) = D(t)$	$b_1 + c_1 = 0$
$C(t) + E(t) = G(t)$	$c_1 + 0 = g_1$
$B(t) + H(t) = G(t)$	$b_1 + 0 = g_2$
$\frac{d}{dt}G(t) = J(t)$	
$\frac{d}{dt}H(t) = K(t)$	
$H(t) = f(G(t)),\ f \in M^+$	
$G(t) * G(t) = L(t)$	
$B(t) = b_1$	
$L(t) + N(t) = B(t)$	
$P(t) * P(t) = N(t)$	$P(t) \geq 0$
$P(t) * K(t) = J(t)$	

TABLE 4. Constraint set of operating region SIN_PLUS

Using the same method of "decoding" the constraint set, one sees that the relationship between G and H in Table 3 can be described as

$$H = f(G) = -\arcsin G + r_1,$$

where r_1 is an arbitrary constant. Similarly, Table 4 expresses

$$H = f(G) = \arcsin G + r_2,$$

for an arbitrary constant r_2. Figures 1-3 depict these relationships on the H-G plane. The domain is exactly π units long in all cases. The difference between the models of Figure 1 and Figure 3 is that the latter one does not require H to be zero when $G = 0$. (Although the combination $<G=0, H=0>$ would, of course, be accepted.)

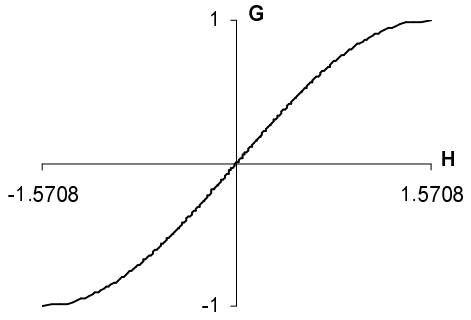

Figure 1. The relationship represented by Table 2

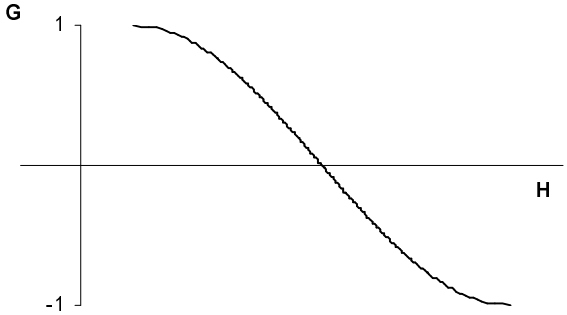

Figure 2. The relationship represented by Table 3

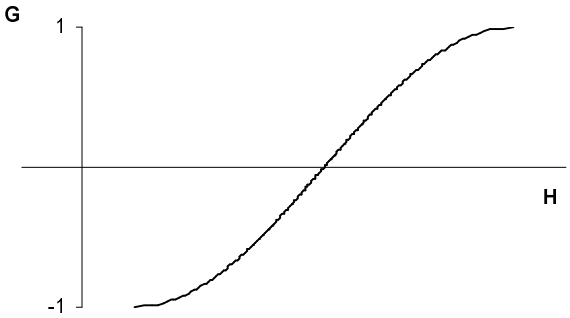

Figure 3. The relationship represented by Table 4

We will now "connect" these three constraint sets to represent sine over the entire real line.

The QSIM input vocabulary enables the user to describe models in terms of several different constraint sets representing different *operating regions* of the system under consideration. The user specifies the boundaries of the applicability ranges of the operating regions in terms of landmarks, which indicate that the simulator should effect a transition to another operating region when they are reached.

For each operating region from which such a transition can occur, one has to specify the following for each possible transition:

- The variable values which will trigger this transition,
- The name of the new operating region,

- The names of variables which will inherit their qualitative magnitudes in the first state after the transition from the last state before the transition,
- The names of variables which will inherit their qualitative directions in the first state after the transition from the last state before the transition,
- Value assignments for any variables which will have explicitly specified values in the first state after the transition.

Consider the QSIM model in Figure 4. What can one say about the relationship between H and G, based on the information contained here? As discussed earlier, the SIN_ORIG region states that G = sin H when $H \in [-\pi/2, \pi/2]$. In this region, when we reach $G = -1$ or $G = 1$, (that is, when H becomes $-\pi/2$ or $\pi/2$,) the model tells us to look at the SIN_MINUS region to see how the variables will behave. (G and H are continuous at this point, since they have been specified to inherit their old values in the transition.) That region states that the function $H = f(G)$ behaves as seen in Figure 2 for a distance of π units in the G-axis. At the other end of this G-domain, we have a continuous transition to a new "segment" of the function which behaves as in Figure 3 for another π units, followed by a segment of the type of Figure 2 again, and so on forever. We deduce that G = sin H for all real H.

Operating Region: SIN_ORIG
Constraint Set: (depicted in Table 2)
Possible Transition:
Trigger: ($G = g_1$) OR ($G = g_2$)
New Operating Region: SIN_MINUS
Variables Inheriting Qmags: *A, B, C, D, E, G, H, J, K, L, N*
Variables Inheriting Qdirs: *A, B, C, D, E, G, H, J, K, L, N*

Operating Region: SIN_MINUS
Constraint Set: (depicted in Table 3)
Possible Transition:
Trigger: ($G = g_1$) OR ($G = g_2$)
New Operating Region: SIN_PLUS
Variables Inheriting Qmags: *A, B, C, D, E, G, H, J, K, L, N*
Variables Inheriting Qdirs: *A, B, C, D, E, G, H, J, K, L, N*

Operating Region: SIN_PLUS
Constraint Set: (depicted in Table 4)
Possible Transition:
Trigger: ($G = g_1$) OR ($G = g_2$)
New Operating Region: SIN_MINUS
Variables Inheriting Qmags: *A, B, C, D, E, G, H, J, K, L, N*
Variables Inheriting Qdirs: *A, B, C, D, E, G, H, J, K, L, N*

Figure 4. QSIM model representing G = sin H

Note that a model that aims to represent n different sine functions with this method will have to include at least 3^n operating regions. This is because at any time, the arguments of these functions can be in any one of the three types of segments of Figure 4. An "n-sine" model like this should be constructed in the following way:

Let CS_{common} be the set of constraints that do not take part in the description of the sine functions in the model. Impose an arbitrary ordering from 1 to n among the sines. The 3^n operating regions will have names of the form $\text{OP\_REG}_{(type\text{-}1,\ type\text{-}2,\ \ldots,\ type\text{-}n)}$, where each *type-i* is one of the symbols *ORIG*, *MINUS*, or *PLUS*, corresponding to the three segment types for the i^{th} sine. The constraint set of a particular region with name $\text{OP\_REG}_{(type\text{-}1,\ type\text{-}2,\ \ldots,\ type\text{-}n)}$ will consist of the union of CS_{common} and the appropriate segment constraint sets (in which the variables necessary for each sine are renamed so that the same name never appears in two or more sets) for each of the sines. Each region will contain descriptions of n possible transitions to "neighboring" regions which can be obtained by changing only one of the *type-i* values (to *PLUS* if it is originally *MINUS*, or to *MINUS* otherwise) in its name.

(Note that the possibility of more than one sine being at their maximums at the same time is handled by the fact that the QSIM output vocabulary treats states before and after a region transition as the same time point, and several such transition-point states can appear as a sequence.)

Return to Hilbert's Tenth Problem

Now that we have shown that it is possible to represent arbitrarily many sine relationships in a QSIM model, it is trivial to finish the construction of the model for the equation

$$\sum_{i=1}^{n} \sin^2 \pi.x_i + P^2(x_1, x_2, \ldots, x_n) = 0,$$

which we started two subsections ago. One creates the 3^n operating regions described in the previous subsection to express that a corresponding variable S_i equals sin PX_i for each of the PX_i. *n mult* constraints equate each variable $S2_i$ to the square of the corresponding S_i, and n more *add* constraints finish the job by expressing the equality of the sum of *P2* and all the $S2_i$ to the variable E, which stands for the left hand side of our equation.

We are now faced with the task of asking the question "Is there a tuple of the x_i's that makes E = 0 ?" in such a way that the answer, if it could be computed, could be read off a QSIM output. The easiest way of doing this is as follows: Run QSIM 2^n times, each time starting the simulation from a different operating region that does not include the segment type *ORIG* in its name, and making sure that each of the QSIM variables representing numerical constants are at their appropriate landmark values at the initial state. All the X_i variables are to be specified to have the qualitative value <?, *std*>, meaning that we do not even know their signs, let alone their numerical values. When its input is specified incompletely, as in this case, QSIM is supposed to create all consistent completions and start simulation from there. If QSIM were able to delete all inconsistent completions and leave all and only the consistent ones in all cases, we would just need to

check whether any complete initial states were created in our 2^n runs to solve Hilbert's tenth problem. All our inputs would be rejected as being inconsistent if and only if P has no solutions in integers. Furthermore, in cases where some initial states survive, meaning that P does have integer roots, we would also be able to see how many of the x_i that form a particular solution are even, and how many are odd, by simply reading the name of the operating region of the surviving state: A zero-crossing of $S_i = \sin PX_i$ indicates an odd value of X_i if and only if the segment is of type *MINUS*.

One may ask why we do not take segments of type *ORIG* into consideration as well in this procedure. The reason is that the *PLUS* segment covers the possibility of crossing the origin anyway. The inclusion of regions with the *ORIG* segments in the overall system model is required for declaring the knowledge that the sinusoidal relationships f_i being described necessarily pass through the origin, that is, to rule out the possibility of $f_i(0) \neq 0$ for any i. (Another possible use of the *ORIG* segment could be for declaring that a particular sine argument is in the interval $[-\pi/2, \pi/2]$, and not anywhere else, at the initial state of a simulation.)

"Non-Deletable" Spurious Behaviors

One may argue that a QSIM prediction B_i starting with initial state $QS(t_0)$ can not be called *spurious* if $QS(t_0)$ itself is mathematically inconsistent. Let us show that the construction of the previous subsection can be modified to allow the solution to the polynomial P to be read off a simulation state computed as a descendant of a consistent initial state.

We change the model of the previous subsection in the following way: Each of the individual terms of P are multiplied by a variable V, resulting in the expression

$$NP = term_1.V + term_2.V + \ldots + term_m.V \ .$$

We then construct the model for

$$\sum_{i=1}^{n} \sin^2 \pi.x_i + NP^2(x_1, x_2, \ldots, x_n) = 0$$

using the techniques described earlier. We run QSIM for a total of 5^n times. The complete initial states for these runs are prepared by the following procedure:

Let the set S equal {<*MINUS*, (−∞, 0)>, <*PLUS*, (−∞, 0)>,
<*MINUS*, (0, ∞)>, <*PLUS*, (0, ∞)>, <*PLUS*, 0>}.
FOR all ways of assigning values from the set S to the variables $A_1, A_2, \ldots, A_n$,
BEGIN
Create an initial state as follows:
The i^{th} subscript of the name of the operating region of this initial state has the value indicated in the first component of A_i;
Variable $X_i(t_0)$ has the value <qmag, *std*>, where qmag is the magnitude indicated in the second component of A_i.
END

The remaining variables are initialized in the following way in all the initial states: All qualitative directions, except that of V, are *std*. V is increasing. All the variables that we created for representing numerical constants are set to their corresponding landmarks. All the S_i variables and V equal zero. Unique assignments are possible for all other variables.

All these initial states are consistent, since the polynomial NP is equal to zero, and the sines being zero just constrain the X_i to be arbitrary integers with the given signs. Consider the simulations starting from these consistent states. In the state for time interval (t_0, t_1), V will be positive, and the determination of whether candidate states with $E = 0$ are spurious or not will have to involve the solution of our unsolvable problem. Any claim that all and only the spurious states will be eliminated by QSIM at this step will be equivalent to saying that QSIM has the capability mentioned in Hilbert's tenth problem.

Since the argument above assumed nothing about the internal workings of the simulator, we can conclude that no qualitative simulator whose input/output vocabulary is identical to that of "pure" QSIM can be both sound and complete.

Conclusion

This paper resolved an open question about whether there exists a qualitative simulation algorithm that predicts all and only qualitative behaviors that correspond to real solutions of ordinary differential equations described by its input model. The answer turned out to be negative; sound qualitative simulation is inherently incomplete. Perhaps counterintuitively, the root cause of the problem is not the perceived weakness, but the actual *power* of the qualitative representation, which enables one to formulate exact numerical equations as input to the simulator. The problem is the inherent incompleteness of mathematics itself, and there is not much that one can do about that.

Of course, the incompleteness results reported here do not diminish the usefulness of qualitative simulators when they are applied to the sort of incompletely specified problems that they were originally designed to deal with. When one already possesses information at the level of precision that the models in this paper contain, one should not employ a *qualitative* reasoner anyway. Together with the hybrid schemes that smoothly integrate signs, reals, (Williams 1991) and intervals of reals, (Berleant and Kuipers 1997) the numerical and purely qualitative methods continue to form the opposite ends of a spectrum of useful tools for scientists and engineers dealing with problems at differing levels of precision.

Acknowledgments

We thank Manuel Bronstein for his helpful answers to our mathematical questions. We are grateful to the three anonymous reviewers for their comments and suggestions.

This work was partially supported by the Boğaziçi University Research Fund. (Grant no: 01A103)

References

Berleant, D., and Kuipers, B. 1997. Qualitative and Quantitative Simulation: Bridging the Gap. *Artificial Intelligence* 95:215-255.

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

Könik, T., and Say, A. C. C. 1998. Extracting and Using Relative Duration Information in Pure Qualitative Simulation. In *Qualitative Reasoning: The Twelfth International Workshop*, AAAI Technical Report WS-98-01, AAAI Press, Menlo Park, California. 155-160.

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

Kuipers, B. J. 1993. Qualitative Simulation: Then and Now. *Artificial Intelligence* 59:133-140.

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

Kuipers, B. 2001. Qualitative Simulation. In Meyers, R. E. ed. *Encyclopedia of Physical Science and Technology, Third Edition.* New York, NY: Academic Press. 287-300.

Matiyasevich, Y. 1993. *Hilbert's Tenth Problem*. Cambridge, Mass.: The MIT Press.

Moses, J. 1971. Algebraic Simplification: A Guide for the Perplexed. *Communications of the ACM* 14:527-537.

Press, W. H.; Flannery, B. P.; Teukolsky, S. A.; and Vetterling, W. T. 1988. *Numerical Recipes in C*. New York, NY: Cambridge University Press.

Say, A. C. C. 1996. Functions Representable in Pure QSIM. In *Proc. Fifth Turkish Symposium on Artificial Intelligence and Neural Networks*, İstanbul, Turkey. 251-255.

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

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. 2001. Improved Reasoning About Infinity Using Qualitative Simulation. *Computing and Informatics* 20(5):487-507.

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.

Struss, P. 1988. Global Filters for Qualitative Behaviors. In *Proc. AAAI-88*, Saint Paul, Minn. 275-279.

Struss, P. 1990. Problems of Interval-Based Qualitative Reasoning. In Weld, D. S., and de Kleer, J. eds. *Readings in Qualitative Reasoning About Physical Systems.* San Mateo, California: Morgan Kaufmann. 288-305.

Williams, B. C. 1991. A Theory of Interactions: Unifying Qualitative and Quantitative Algebraic Reasoning, *Artificial Intelligence* 51:39-94.