Proving properties of continuous systems: qualitative simulation and temporal logic*

Benjamin Shults
Department of Mathematics (bshults@math.utexas.edu)
University of Texas at Austin, Austin, TX 78712 USA

Benjamin J. Kuipers
Computer Science Department (kuipers@cs.utexas.edu)
University of Texas at Austin, Austin, TX 78712 USA

To appear, *Artificial Intelligence Journal*, 1997.

Abstract

We demonstrate an automated method for proving temporal logic statements about solutions to ordinary differential equations (ODEs), even in the face of an incomplete specification of the ODE. The method combines an implemented, on-the-fly, model-checking algorithm for statements in the temporal logic CTL* [3, 7, 8] with the output of the qualitative simulation algorithm QSIM [13, 16]. Based on the QSIM Guaranteed Coverage Theorem, we prove that for certain CTL* statements, Φ, if Φ is true for the temporal structure produced by QSIM, then a corresponding temporal statement, Φ', holds for the solution of any ODE consistent with the qualitative differential equation (QDE) that QSIM used to generate the temporal structure.

1 Introduction

The world is continuous and dynamic, but we want to use discrete symbolic means to reason reliably about it. We demonstrate a method for doing this for a significant range of cases by using qualitative simulation to generate a finite structure guaranteed to describe the behaviors of the continuous system, then interpreting that structure as a model to check the validity of statements in temporal logic.

*This work has been supported in part by the National Science Foundation (grant IRI-9216584) and by the Electric Power Research Institute. A preliminary report on this work appeared as [17].

The main theorem of this paper, stated informally, says the following: Suppose M is a QSIM behavior tree generated from the qualitative differential equation C. If M is a model for a temporal logic formula, then the formula describes every solution to every ODE which abstracts to C. Of course, we will formalize all of these relationships carefully in this paper.

In many applications in which ordinary differential equations are used, information about initial conditions or the specific relationship between a pair of quantities is not completely known. In some cases constants are only known to lie in a certain range or the relationship between quantities is only known to be monotonic. Qualitative reasoning allows this information to be used to generate descriptions of solutions to any ODE which abstracts to the known information. We call such an abstract ODE a qualitative differential equation (or QDE). In many such applications we want to draw conclusions about the solution to *any* ODE consistent with the limited information we have about a system.

Furthermore, there are a number of applications of model-based reasoning that can profit from reliable inference about time-ordered events over the set of possible behaviors of a continuous system. Since applications such as control, monitoring, diagnosis and design must often cope with conditions of incomplete knowledge, the ability to do temporal reasoning over the possible behaviors of a system described by a qualitative or semi-quantitative model is particularly valuable. Our program, TL, makes a formal connection between solutions to real differential equations and temporal-logic model checking.

A qualitative simulator, such as QSIM, constructs a tree-like structure whose branches represent the possible behaviors consistent with the qualitative differential equation and initial state input to the QSIM algorithm [13, 16]. This set of behaviors is expressed as a finite structure of qualitative state descriptions. In the case of QSIM, this structure is guaranteed to contain a branch which describes any "reasonable" extended real valued function which is a solution of an ordinary differential equation which abstracts to the QDE under circumstances to be described. We call this property the "soundness" of QSIM, and this property is the content of the Guaranteed Coverage Theorem.

Since the output of the QSIM algorithm is a structure whose paths describe reasonable, extended real valued functions, we would like to be able to formulate temporal questions about the system it describes and have those questions answered. This is accomplished using temporal-logic model-checking. A model-checking algorithm takes as input a temporal-logic formula and a tree-like structure and determines whether the structure is a model (in the logical sense of the word) for the formula. Temporal logic augments propositional logic with temporal operators on time-varying truth-values, such as *always*, *eventually*, and *until*. Modal logic adds operators for truth-values in alternate possible worlds (i.e., alternate behaviors or paths), such as *necessarily* and *possibly*.

We have chosen to use the branching-time temporal logic CTL* which is described by Emerson and Clarke [7, 8].

Because QSIM is sound, for any CTL* statement Φ which is "universal" in a

sense we will define, if Φ is modeled by the structure produced by QSIM, then a corresponding theorem holds for the solution of any ordinary differential equation consistent with the QDE that generated the QSIM structure. Therefore, at least for universals, statements in temporal logic about continuous systems can be proved by qualitative simulation. This allows a hybrid reasoning system to prove common-sense statements and to do expert reasoning about dynamical systems.

We also provide a limited completeness result: in case all paths in the structure output by QSIM describe reasonable, extended real valued functions which are solutions to differential equations consistent with the QDE input to QSIM, then even CTL* formulas which are not universal may be used to prove properties of the system.

The propositional part of the temporal language includes propositions which allow the construction of formulas containing numerical information. This can be used in conjunction with the numerical extensions to QSIM—Q2 [14], Q3 [2] and NSIM [12]—in order to prove numerical properties of physical systems.

In Section 2 we describe and define the temporal-logic language CTL* and present some basic definitions and facts which will be needed in our main theorem. The reader already familiar with CTL* may want to read only Section 2.1 to learn about our notation conventions and Section 2.4 to see the standard results from the literature which we will be using.

In Section 3 we describe the QSIM framework and prove the Guaranteed Coverage Theorem. Even readers familiar with QSIM should read most of Section 3 since we use an updated formalization and add some new terminology.

Section 4 begins to show how the QSIM framework and the underlying differential equations are related to the theory of temporal logic and CTL* formulas. There we explain how the output of the QSIM algorithm is used as a structure over which formulas in CTL* can be interpreted. We also show how CTL* formulas describe reasonable real-valued functions.

In Section 5 we introduce the last hypothesis to the main theorem and prove the main theorem. We also prove some useful special cases and a completeness result. Section 5 also discusses some issues concerning the implementation.

In Section 6 we describe some applications of the combination of temporal-logic model-checking with qualitative simulation.

Sections 2–4 lay the groundwork for the statement of the main theorem. We will be stating the main theorem in increasing degrees of formality as we develop the terminology.

2 CTL*

Computational Tree Logic (CTL and its extension CTL*) is a branching-time temporal logic. The theory of branching-time temporal logics is summarized by Emerson in *The Handbook of Theoretical Computer Science* [8]. We will

customize CTL* slightly in order to allow states with no successors because in continuous systems a state may have no successor (e.g. if time reaches infinity or if the value of some variable crosses a boundary of its range). In this section, we define the syntax and semantics of the CTL* language and, in Section 2.4, give some basic results and definitions which will be used by our main theorems. The presentation of CTL* here does not differ significantly from the presentation of the language in [8] except in the notation we use. We use this notation as a convenience for our implementation.

A model-checking algorithm examines a temporal structure and a temporal-logic formula and determines whether the structure is a model (in the logical sense of the word) for the formula.

Our implementation (TL) of a model-checking algorithm for CTL* is an "on-the-fly" model-checker based on the algorithm of Bhat, Cleaveland and Grumberg [3]. On-the-fly algorithms have the advantage over the more common "global" algorithms of being able to terminate with the correct result before constructing the entire exponentially-large structure. If the formula happens to be in the sublanguage CTL of CTL* then the complexity of this on-the-fly algorithm is the same as the best known algorithms for CTL model-checking. Our implementation is customized for expressing statements about continuous systems (see Section 4).

2.1 Terminology and Notation

We interpret a CTL* formula over a temporal structure $M = \langle S, X, L \rangle$ where

- S is a set of states,
- X is a set of fullpaths,
- $L : S \times AP \to \{T, F\}$ is an interpretation which takes a state $s \in S$ and an atomic proposition $\phi \in AP$ and assigns a Boolean truth value.

Here AP is the set of atomic propositions. A *fullpath* is a path which is either infinite or terminates with a state which has no successor.

We use the notation $\langle s_0, s_1, s_2, \ldots \rangle$ to denote an infinite or finite totally-ordered set. We let $\Lambda(x)$ denote the cardinality of a finite, totally-ordered set x. If x is an infinite, totally-ordered set, then by $i < \Lambda(x)$ we mean i is any nonnegative integer. Here we use totally ordered sets to represent paths and fullpaths. Notice that the last state in a finite fullpath $x = \langle s_0, s_1, s_2, \ldots \rangle$ is $s_{\Lambda(x)-1}$.

We now describe the path quantifiers and the basic temporal operators on propositions. The names we use for path quantifiers and temporal operators are equivalent to the more concise names used in the temporal logic research community:

`A` $\equiv$ `necessarily`	`G` $\equiv$ `always`	`X` $\equiv$ `next`
`E` $\equiv$ `possibly`	`F` $\equiv$ `eventually`	`U` $\equiv$ `until`

We prefer to give a rough description before the formal syntax and semantics are defined. Suppose some state s and path x starting at s are given and that p is a formula. The two path quantifiers are

`(necessarily` p`)`, which is true at s if p is true of *every* fullpath starting with s, and

`(possibly` p`)`, which is true at s if p is true of *some* fullpath starting at s.

The elementary temporal operators are `(next` p`)` and `(until` p q`)` where p and q are formulas.

`(next` p`)` is true of the path x if $\Lambda(x) = 1$ or p is true of the path obtained from x by deleting its first state, and

`(until` p q`)` is true of x if q is true of some state in x and p is true of every state preceding the first state in which q is true. This operator is sometimes called `strong-until`, to distinguish it from `weak-until` to be defined below.

The precise syntax and semantics of `until` and `next` will be defined in the following sections. We will use the following abbreviations to define other operators in terms of `until` and `next`:

`(releases` p q`)`	$\equiv$	`(not (until (not` p`) (not` q`)))`
`(before` p q`)`	$\equiv$	`(not (until (not` p`)` q`))`
`(strong-next` p`)`	$\equiv$	`(not (next (not` p`)))`
`(eventually` p`)`	$\equiv$	`(until true` p`)`
`(always` p`)`	$\equiv$	`(not (eventually (not` p`)))`
`(never` p`)`	$\equiv$	`(always (not` p`))`
`(weak-until` p q`)`	$\equiv$	`(before` q `(and (not` q`) (not` p`)))`
`(infinitely-often` p`)`	$\equiv$	`(always (eventually` p`))`
`(almost-everywhere` p`)`	$\equiv$	`(eventually (always` p`))`

The formula `(releases` p q`)` is true of a path if q is always true or if q is true through the first state in which p is true. The statement `(before` p q`)` is true of a path if p is true in some state previous to the first state in which q is true (though q does not necessarily ever become true). The formula `(weak-until` p q`)` is true of a path if p is true in every state or in every state before the first state in which q is true.

Because we are applying CTL* to structures which may have finite fullpaths, the temporal operator `next` may seem ambiguous. Therefore, we must distinguish between `strong-next` and `weak-next`. The statement `(weak-next` p`)` is

true of a path if the path has no next state or if the path has a second state and p is true of it. The statement `(strong-next` p`)` is true of a path if the path has a second state and p is true of that state. In our discussion, we consider `next` alone to mean `weak-next`.

In the following two subsections we give the formal definitions for the temporal operators and path quantifiers of CTL*.

2.2 Syntax

A state formula is a formula which is interpreted over a state and a path formula is a formula which is interpreted over a path. State formulas in CTL* are generated by rules (S1–S3) below. The path formulas in CTL* are generated by rules (B1–B3) below. Although the semantics of `releases`, `strong-next` and `or` can be derived from their definitions as abbreviations, we include the definitions here so that the proofs later will be easier to follow.

Definition 1 The syntax of CTL* is defined as follows.

(S1) Each atomic proposition ϕ is a state formula,

(S2) if $p_1, \ldots, p_n$ are state formulas then so are `(and` $p_1 \cdots p_n$`)`, `(or` $p_1 \cdots p_n$`)` and `(not` p_1`)`,

(S3) if p is a path formula then `(possibly` p`)` and `(necessarily` p`)` are state formulas,

(B1) every state formula is a path formula,

(B2) if $p_1, \ldots, p_n$ are path formulas then so are `(and` $p_1 \cdots p_n$`)`, `(or` $p_1 \cdots p_n$`)` and `(not` p_1`)`,

(B3) if p, q are path formulas then so are `(next` p`)`, `(strong-next` p`)`, `(releases` p q`)` and `(until` p q`)`.

We also allow the standard boolean abbreviation for `implies`.

2.3 Semantics

The following notation is needed before the semantics of our logic can be defined. Given a path $x = \langle s_0, s_1, s_2, \ldots \rangle$, for every nonnegative integer $i < \Lambda(x)$ we let x^i denote the path $\langle s_i, s_{i+1}, s_{i+2}, \ldots \rangle$, which is the suffix of x starting at s_i. Thus, for any nonnegative integer $i < \Lambda(x)$, x^i is the path obtained from x by deleting from x the first i states. Notice that if x is finite, then x^i is not defined for $i \geq \Lambda(x)$ and $\Lambda(x^i) = \Lambda(x) - i$.

Now we are ready to give the semantics for the language. We write $M, s_0 \models \Phi$ (respectively $M, x \models \Phi$) to mean that the state formula Φ (respectively path formula Φ) is true in the temporal structure M at the state s_0 (respectively of

the path x). Each item below gives the interpretation of the corresponding item in the syntax above.

Definition 2 If s_0 is a state in M and $x = \langle s_0, s_1, \ldots \rangle$ is a nonempty fullpath in M starting at s_0, then we inductively define $\models$ as follows:

(S1) $M, s_0 \models \phi$ where ϕ is an atomic proposition if and only if $L(s_0, \phi) = \mathrm{T}$,

(S2) $M, s_0 \models$`(and` $p_1 \cdots p_n$`)` if and only if $M, s_0 \models p_i$ for all $i, 1 \leq i \leq n$,
$M, s_0 \models$`(or` $p_1 \cdots p_n$`)` if and only if $M, s_0 \models p_i$ for some $i, 1 \leq i \leq n$,
$M, s_0 \models$`(not` p`)` if and only if it is not the case that $M, s_0 \models p$,

(S3) $M, s_0 \models$`(possibly` p`)` if and only if there is a fullpath y in M starting at s_0, such that $M, y \models p$,
$M, s_0 \models$`(necessarily` p`)` if and only if for every fullpath y in M starting at s_0, $M, y \models p$.

(B1) $M, x \models p$ where p is a state formula if and only if $M, s_0 \models p$,

(B2) $M, x \models$`(and` $p_1 \cdots p_n$`)` if and only if $M, x \models p_i$ for all $i, 1 \leq i \leq n$,
$M, x \models$`(or` $p_1 \cdots p_n$`)` if and only if $M, x \models p_i$ for some $i, 1 \leq i \leq n$,
$M, x \models$`(not` p`)` if and only if it is not the case that $M, x \models p$,

(B3) $M, x \models$`(until` p q`)` if and only if there is a nonnegative integer $i < \Lambda(x)$, such that $M, x^i \models q$ and for every nonnegative integer $j < i$, $M, x^j \models p$,
$M, x \models$`(releases` p q`)` if and only if for every nonnegative integer $i < \Lambda(x)$, $M, x^i \models q$ or there is a nonnegative integer $i < \Lambda(x)$ such that $M, x^i \models p$ and for every $j \leq i$, $M, x^j \models q$,
$M, x \models$`(next` p`)` if and only if $\Lambda(x) = 1$ or $M, x^1 \models p$,
$M, x \models$`(strong-next` p`)` if and only if $\Lambda(x) > 1$ and $M, x^1 \models p$.

2.4 Basic Results

The proofs of our main theorems will use the fact that any formula can be written in the following form.

Definition 3 (Positive Normal Form) A CTL* formula is in *positive normal form* if `until`, `releases`, `next` and `strong-next` are the only temporal operators in the formula and for every `not` in the formula, its scope is an atomic proposition.

Here we require that `implies` first be rewritten in terms of `not` and `and` or `or`. Every CTL* formula is equivalent to a formula in positive normal form because all temporal operators can be written in terms of those mentioned above and `not`s can be propagated inward to propositions [3].

Definition 4 (Universal Formula) A CTL* expression Φ is said to be *universal* if, when the formula is written in positive normal form, there are no occurrences of the path quantifier `possibly`.

We call a path formula a *perfect path formula* if it contains no path quantifiers. These are exactly the formulas which correspond to formulas in Propositional Linear Time Logic (PLTL). If Φ is a formula in CTL*, then Φ' denotes the perfect path formula obtained from Φ by deleting all occurrences of the path quantifiers. For example, if p and q are propositions and

$\Phi =$ `(necessarily (until` p `(necessarily` q`)))`,

then

$\Phi' =$ `(until` p q`)`.

We call Φ' the *perfection* of Φ.

The following lemma is needed in the proof of Lemma 2 which is used in the proof of one of the main theorems.

Lemma 1 If Φ is a universal formula and x is a fullpath in M such that $M, x \models \Phi$, then $M, x \models \Phi'$.

The proof of this is complex and not enlightening. Therefore, it has been put in Appendix C.

Lemma 2 For every universal CTL* state formula Φ, and every temporal structure M and state s in M, if $M, s \models \Phi$ then for every fullpath x in M starting at s, $M, x \models \Phi'$.

bf Proof: The proof follows easily by induction on the length of Φ by using Lemma 1.

3 QSIM

In Section 3.1 we briefly describe the QSIM framework. We refer the reader to Kuipers' full description of the QSIM framework [16] and to Appendix B for details on the new definition of a reasonable function. Other reformalizations of concepts related to the Guaranteed Coverage Theorem are described in the present section.

The QSIM algorithm takes as input the user's qualitative or semi-quantitative description of a physical system. This input is called a qualitative differential equation. This description is formally related to some class of ODEs as we explain below. The output from the QSIM algorithm is a tree whose nodes are states describing the values of the variables in the input QDE.

The main theorem of this section (the Guaranteed Coverage Theorem) stated informally says that every solution to any ODE related to the QDE is represented in the tree output by QSIM. We give the formal statement of the Guaranteed Coverage Theorem below.

Sections 3.1 and 3.2 explain some of the basic terminology used in the statement of the Guaranteed Coverage Theorem. These sections also explain why the hypotheses of the theorem are necessary. Those sections are designed so that the basic ideas are easy to find. A casual reader should be able to understand the statement of the Guaranteed Coverage Theorem without reading all of the details in Sections 3.1 and 3.2.

Section 3.1 among other things, formalizes the relationship between QDEs and ODEs and the relationship between the finite output of QSIM and the generally infinite structure which it represents. Section 3.2 formalizes the relationship between fullpaths in QSIM structures and continuous functions.

Now we give the formal statement of the Guaranteed Coverage Theorem. All unfamiliar terms used in this statement (e.g., specification, splitting, closed, abstraction) are defined in Sections 3.1 and 3.2.

Suppose M is a closed tree generated from the QDE and initial state $\langle C, I\rangle$. Suppose the ODE, F, abstracts to C and that the structural abstraction, F', of F has solution set U. The QSIM algorithm is carefully crafted to guarantee that the qualitative structure of U is described by some rooted fullpath in the represented structure $\widehat{M}$:

Theorem 1 (Guaranteed Coverage) Under the conditions above, there is a rooted fullpath x in $\widehat{M}$ and a specification $\langle x, c\rangle$ of x such that $\langle x, c\rangle$ qualitatively describes some splitting $\langle \{t_i\}, U\rangle$ of U.

The proof is given by Kuipers [13, 16]. Because most of the QSIM framework—the algorithm itself, for example—is beyond the scope of this paper, we will not detail the proof here.

3.1 The QSIM Framework

A *qualitative differential equation* consists of a finite set of *variables* (each of which is associated with a *quantity space* which is a totally-ordered set of *landmarks*), and a set of constraints on the values of the variables. A QDE is a structural abstraction of a class of ordinary differential equations. The QDE codifies the QSIM user's incomplete knowledge of a physical system.

Starting with a QDE, C, and an initial state, I, qualitative simulation with QSIM produces a finite tree, $M = \langle S, R, B\rangle$, of *qualitative states*, linked by the QSIM *successor* relation, R. The finite tree which it produces is called a *QSIM behavior tree* in the literature. We let $M = \langle S, R, B\rangle$ represent the behavior tree where S is the set of states, R is the successor relation and B is the set of fullpaths starting at the initial state. Each behavior is represented as a finite,

totally-ordered set beginning at the initial state and terminating at a state with no R-successor. The set B is completely determined by the relation R and the initial state I. Because of the significance of the initial state, we call fullpaths whose first state is I *rooted fullpaths*.

We will say that this QSIM behavior tree was *generated by* the pair $\langle C, I\rangle$ of the QDE and the initial state. A *QSIM behavior* is a path in the behavior tree, starting at the root and terminating at a leaf of the tree, i.e. B is the set of QSIM behaviors in M. Each state describes the *qualitative value* of each *variable* appearing in the QDE model. Each variable will represent a function of time. The qualitative value of a variable v over a state s is of the form $\langle qmag, qdir\rangle$, where *qmag* describes the magnitude of v as equal to a landmark or in an open interval defined by two landmarks, and *qdir* is the sign of the derivative of v. By considering the qualitative values of the variables at a state, and the constraints in the QDE, QSIM is able to derive a number of properties of the states and behaviors, including quiescence, stability and cycles.

A QSIM state is called a *transition state* if it has no R-successors due to the fact that the value of one of its variables crosses a boundary of the QDE description. QSIM allows the user to produce *transition relations* between a transition state in one tree and the root of a tree generated by another QDE. This allows the user to produce a tree which has different models for its behavior in different ranges. The theorems here could be extended to take transition relations into account, however, the extension is tedious and unenlightening so, in the theorems in this paper, we assume that transition states have no successors.

Structural Abstraction The class of ODEs related to a given QDE is that class of ODEs which structurally abstract to the QDE. The concept of structural abstraction is best understood by example.

Example 1 Given an ODE, F, there is associated with it a set, F', of simultaneous equations which is derived from F. We will call F' the *structural abstraction* of F. For example, consider an equation for simple harmonic motion:

$$\frac{d^2x}{dt} = -x$$

We structurally abstract this equation in several steps. First we introduce a variable v so that $v = \frac{dx}{dt}$ and again we let $a = \frac{dv}{dt}$. Finally, we write $a = -x$ so that F' is a set of three equations in three variables (not including time). At each step, the equation is broken down into its components until each equation is simple enough to be abstracted to a QSIM constraint. This set of three equations is called the structural abstraction of the original equation for harmonic motion and is denoted F'. From the structural abstraction, it is easy to create a QDE. See Section 3.3.1 of Kuipers' book *Qualitative Reasoning* [16] for more details on the structural abstraction of an ODE.

The structural abstraction, F', is useful because it can easily be abstracted into a QDE. Following our example, we obtain the following QDE from F':

```
(d/dt X V)
(d/dt V A)
(M- a X)
```

¿From this QDE, QSIM will produce a temporal structure. Since the equation describing simple harmonic motion abstracts to this QDE, we would like to know that the solutions to this ODE are described by some fullpath in the temporal structure generated by QSIM. The Guaranteed Coverage Theorem says just that.

A solution, U, of F' is a set of functions of time which simultaneously make each of the equations in F' true. Since U is a set of functions we need a way of relating the functions in U to the variables in F' for which they are supposed to be substituted. We will use the symbol ψ to represent this bijection from the set of functions in U to the set of variables in F'. Similarly, if M is a QSIM tree produced by the QDE, C, abstracted from F' and U is a solution to F', we let $\psi_{U,M}$ be the bijection from U to the set of variables in C. The bijection is simply the relationship between the names of the variables in F' and the names of the variables in C.

It should be clear that a solution, U, to F' can be converted into a solution to F by going through this transformation in the other direction. For example, since $U = \{\sin, \cos, -\sin\}$ is a solution to F', where $\psi(\sin) = x, \psi(\cos) = v$ and $\psi(-\sin) = a$, we can conclude that the sine function is a solution to the original equation for simple harmonic motion.

The Represented QSIM Structure Here we make the important distinction between the finite QSIM tree, M, and the corresponding infinite structure $\widehat{M}$. Essentially, $\widehat{M}$ is obtained from M by following cycles states through the states which they match. However, we have to be careful to do this in a sensible way when the strong-match criterion is used.

QSIM may use various matching criteria when it detects cycles. The *strong-match* criterion requires that the value of each variable in the states to be matched is a landmark (rather than an interval) and that those landmarks match the values of the variables in the previously-existing state (the qualitative derivatives have to match regardless of the match criterion). The *weak-match* criterion allows a match when the values are either intervals or landmarks. The QSIM user may also dictate whether cycles are detected across QSIM behaviors (cross-edge cycles) or only on the same QSIM behavior.

The type of cycle detection chosen makes a difference in the interpretation of the tree. If strong-matching is used, then a match represents a real cycle in the system. That is, the system has returned to a previous state and therefore, by the uniqueness theorem for differential equations, it must continue from that

point exactly as it did before. Therefore, such a cycle *behavior* represents a single *fullpath* in the infinite structure. If weak-matching is used, then a match does not necessarily represent precisely the same state and hence, the system may continue from the cycle along a different path than the one it has already followed.

In the former case, there will be a one-to-one correspondence between the behaviors in M and the rooted fullpaths in $\widehat{M}$. In the latter case, we may end up with infinitely many rooted fullpaths in $\widehat{M}$.

We want to do temporal reasoning about paths which pass through these cycle states. Therefore we will define what we call the *QSIM structure,* $\widehat{M} = \langle S, X \rangle$, *represented by* $M = \langle S, R, B \rangle$. Here X is the set of fullpaths *represented* in the behavior tree M. To construct X, we first define the set X_r of *rooted fullpaths*. A rooted fullpath in $\widehat{M}$ is a path starting at the root of the QSIM tree and continuing through cycle states in a semantically sensible way. That is to say, we only add fullpaths which satisfy the restrictions mentioned above related to the type of cycle matching used. If the strong-match criterion was used with no cross-edge cycles allowed, then for each cycle behavior, we add a single infinite fullpath which follows that behavior then passes through the same cycle infinitely many times. If another kind of matching was used, then we simply add the cycle pairs to the relation R to obtain the relation $\widehat{R}$ and X_r is the set of rooted fullpaths generated by $\widehat{R}$. Finally, we define X to be the suffix closure of X_r.[1]

In the case of strong-matching with no cross edge cycles, it will be useful to go into more detail. Since, in this case, each behavior becomes associated with a single rooted fullpath we can define the natural bijection, z_0, from the set of rooted fullpaths, X_r, to the set of behaviors, B. Since, in this case, any fullpath in X is a suffix of a unique rooted fullpath, we can extend the bijection z_0 to a function $z : X \to B$ so that for any $x \in X, z(x) = z(x_r)$ where x_r is the rooted fullpath of which x is a suffix. (Thus, z is not generally a bijection.) We think of z as mapping a fullpath to its associated QSIM behavior. This assignment will be useful when we prove properties of systems about which we have some quantitative information.

Closed Trees Ideally, given a QDE, the QSIM algorithm will terminate, not because it runs out of memory or other resources, but because it has finished simulating all possible behaviors. When the QSIM algorithm terminates in this "natural" way, we call the tree it produces *closed*. In this case, every behavior in the behavior tree returned by QSIM terminates with a state which is a transition state, a cycle state or a quiescent state. There are cases, however, in which QSIM does not return a closed tree regardless of how long it is allowed to run.

[1]Notice that, in the former case, X is not necessarily fusion closed and hence not R-generable. A set X is fusion closed if, whenever $x_1 s y_1, x_2 s y_2 \in X$, then $x_1 s y_2 \in X$ for any states x_1, x_2 and path s, y_1, y_2. A set X is R-generable if it is naturally generated by some relation [8].

In cases where QSIM returns a tree which is not closed, the hypotheses of the Guaranteed Coverage Theorem do not hold. If the behavior tree M is not closed then it is possible that an actual behavior of the system is not represented by any rooted fullpath in the represented structure $\widehat{M}$.

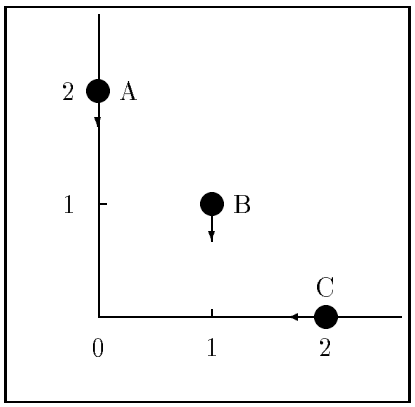
The normal QSIM simulation style creates new landmarks for critical values, applies a strong cycle-match criterion (all variables must have identical landmark values), and does not allow *cross-edge* cycles (i.e. considers cycle matches only within the same QSIM behavior.) Under this simulation style, certain systems such as the damped spring never close. However, by applying the *envisionment* simulation style (no new landmarks, weak cycle-match criterion, and cycle matches anywhere in the behavior tree), *every* qualitative model has a finite closed behavior tree. (See Chapter 5 of Kuipers' *Qualitative Reasoning* [16].)

Quantitative Information When strong-cycle matching is used and cross-edge cycles are not allowed, then the QDE and the initial state may be augmented with quantitative information such as numerical interval bounds on the real values denoted by landmarks and other symbolic terms in the behavior prediction [2, 12, 14, 16]. In this case, QSIM propagates this quantitative information and uses it to prune branches of the tree which are inconsistent with the information. The most important quantitative information for the purposes of this paper is the information which QSIM derives about the landmarks.

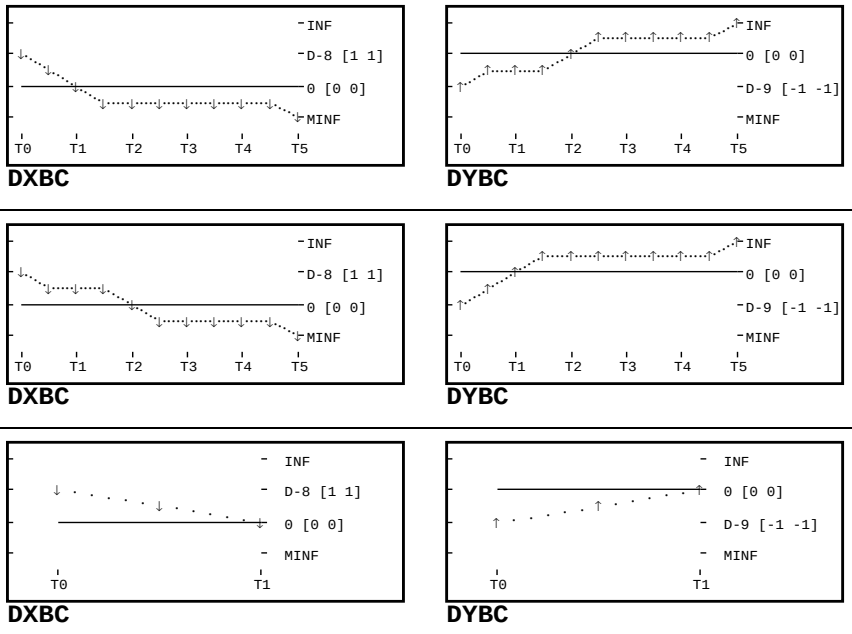
A landmark in a quantity space of a QSIM variable is intended to name some real number. The quantitative extensions to QSIM are able to restrict the possible values of a landmark to some closed, extended-real interval. This quantitative information may be different on each QSIM behavior. Thus, the user cannot simply ask for the range of the possible values of a landmark. The user must ask for the range of the possible values of a landmark *in a given behavior*. So in this case, we will use the function z to determine which behavior a given fullpath is related to. If weak-cycle matching is used or cross-edge cycles are detected, then the numeric information loses its sense.

Example 2 In order to illustrate the fact that quantitative information is stored on QSIM behaviors rather than on states, we will construct a simple example with numeric information. Three billiard balls [20] start to move with constant velocities and initial positions shown in Figure 1(a). The QSIM QDE model for this scenario provides quantity spaces for position, velocity, and acceleration in the x and y directions, and constraints for constant-velocity motion. Collisions are detected when the differences in x and y positions of two balls are simultaneously zero.

When there is partial quantitative information about the speeds of the balls—`A` and `B` have velocity -2 units/second, and the velocity, `Cxp`, of `C` is some constant within the interval $[-3.5, -1.5]$ in units/second—QSIM predicts three possible behaviors, corresponding to `C` passing ahead of `B`, passing behind



(a) Three balls on a billiard table, with initial positions and velocities.



(b) Given incomplete knowledge of the speed of ball C, QSIM predicts that C may pass ahead of B, behind B, or collide with B at $t = 0.5$.

Figure 1: Predicting behaviors of a real-time system.

B, and colliding with B (Figure 1(b)). In case C collides with B, the collision takes place at $t = 0.5$ seconds. There is no possibility of C colliding with A.

We chose to deal with this amount of information because it illustrated our point without much complexity. Naturally, if the user had more or less knowledge about the conditions on the system, another QDE and initial state could be constructed.

Now, consider the value of horizontal velocity, Cxp, of C in the first state. We know that it is a real number between -3.5 and -1.5 and that it is constant. In the third state of the third behavior, we know that Cxp is equal to -2. In the third state of the second behavior, we know that Cxp is greater than -2. In the third state of the first behavior, we know that Cxp is less than -2. But since Cxp is a constant, its value over each behavior does not vary, therefore, its value at the first state (which all three behaviors share) depends on which behavior we are in. QSIM must store the quantitative information about the values of landmarks not at each state but at each behavior.

We shall return to this example in section 6.1.

3.2 Qualitative Description

In this section, we give a formal meaning to the following informal phrase: "the QSIM fullpath x describes the set of real valued functions U." This is what we want to say when U is the solution to an ODE which abstracts to the QDE used to generate the behavior x.

In order to do this, we will first partition the domain of the functions in U in such a way that the partition corresponds to the value of the time variable in the states in x. That will be called a *splitting* of U. Second, we will assign specific real numbers to the landmarks of the variables in x. This will be called a *specification* of x. From there, it will be relatively easy to define what it means for the fullpath x to describe the set of functions U. At the end of this section, we give a detailed example showing how the function sine is described by the rooted fullpath generated by QSIM, given the simple harmonic motion QDE.

Splittings We are given a set of reasonable, extended real valued functions $U = \{u_i : 1 \leq i \leq n\}$. Since we are thinking of U as a solution to a set of simultaneous equations which was derived from an ODE, we will assume that each of the functions in U shares the domain, A, some interval (of time) in the extended reals. In order to say that U is described by a QSIM fullpath, we need a way of partitioning the domain of the functions in U that will be consistent with the values of the time variable in the fullpath.

We define a *splitting* of U as follows. Let $\{t_i\}$ be a strictly increasing sequence of points (indexed from 0) in A satisfying the following conditions: (1) if t is a critical point of some u_k, then $t \in \{t_i\}$, (2) $\{t_i\}$ has no finite limit point and (3) $\{t_i\}$ converges to ∞ only if $\infty \notin A$. According to the definition of a reasonable function (Appendix B), such a set exists, and may be infinite only

if ∞ is the supremum of A and A is open on the right. We will call the pair $\langle \{t_i\}, U \rangle$ a *splitting* of U. Since the critical points of the functions all must be in $\{t_i\}$ (condition 1 above), one splitting is distinguished from another splitting by the choice of non-critical points in $\{t_i\}$. If the set $\{t_i\}$ is finite, then we will let t_J be the greatest element of $\{t_i\}$. Consequently, $J+1$ is the cardinality of $\{t_i\}$.

Associated with any splitting $\langle \{t_i\}, U \rangle$ there is a natural partition $\{D_k\}$ of the interval A. Each D_k is either a singleton containing one of the points in $\{t_i\}$ or an open interval whose endpoints are two consecutive points in $\{t_i\}$. The indexes on the sets in the partition follow the order of the indexes in $\{t_i\}$. That is, if A is closed on the left then $D_0 = \{t_0\}, D_1 = (t_0, t_1)$, and so on. If A is open on the left then $D_0 = (a, t_0), D_1 = \{t_0\}, D_2 = (t_0, t_1)$, and so on, where a is the infimum of A. If A is closed on the right then $D_{2J} = \{t_J\}$ (or $D_{2J+1} = \{t_J\}$ if A is open on the left). To avoid the problem of going to this much trouble to figure out which t_i is the left endpoint of D_k, we will let d_k denote the index such that t_{d_k} is the left endpoint of D_k. If A is open on the right, then as we have said, there may be infinitely many sets in $\{D_k\}$.

When dealing with a splitting, $\langle \{t_i\}, U \rangle$, of U, we will use the following abbreviations:

$$\begin{aligned} U|_{_l} &= \{u_i|_{_{D_l}} : 1 \leq i \leq n\} \\ U|_{_{l+}} &= \{u_i|_{\bigcup_{l \leq k} \{D_k\}} : 1 \leq i \leq n\} \end{aligned}$$

That is to say that $U|_{_l}$ is the set of functions in U each restricted to the domain D_l and $U|_{_{l+}}$ is the set of functions in U each restricted to the domain $\bigcup_{l \leq k} \{D_k\}$.

Specification of a fullpath In order to define the notion of a splitting being *described* by a fullpath, we will need to relate the landmarks in the fullpath to specific extended real numbers. We call a mapping of landmarks to extended real numbers satisfying certain sensible conditions, a *specification* of the landmarks. We want the specification to be order-preserving. Also if quantitative information is assigned on a fullpath, we want the mapping to be consistent with that information.

Suppose c is a function with domain, some partially ordered set V of landmarks, and range, the set R^* of extended real numbers. The function c is called a *specification* of V if c preserves the partial order and $c(\texttt{minf}) = -\infty, c(\texttt{inf}) = \infty$ and $c(0) = 0$. The reason the order on V is only partial is that the landmarks come from different variables. The landmarks of any single variable are totally ordered in V.

Given a fullpath x and a specification, c, of the landmarks of the variables in x under the partial order determined by the quantity spaces of the variables in x, we call the pair $\langle x, c \rangle$ a *specification of* x if c is also consistent with any numeric information which might be associated with $z(x)$. (Recall, $z(x)$ is the

QSIM behavior associated with x when such can be determined uniquely.) In particular, if, on the behavior $z(x)$, the qualitative landmark, `X1`, has been determined, by a quantitative extension to QSIM, to refer to a number in the numeric range $[n_1, n_2]$, then $c(\texttt{X1}) \in [n_1, n_2]$.

Definition of Qualitatively Describes Now we can state the phrase "the QSIM fullpath x describes the set of real valued functions U" formally. Given a specification $\langle x, c\rangle$ of a fullpath $x = \langle s_0, s_1, \ldots\rangle$ in a QSIM directed graph $\widehat{M}$ and a splitting $\langle\{t_i\}, U\rangle$ of the set of reasonable, extended real valued functions $U = \{u_i : 1 \leq i \leq n\}$ with common domain, A, we say that $\langle x, c\rangle$ *qualitatively describes* the splitting if the specification corresponds to the splitting as described in detail in the remainder of this subsection.

The intention is that the partition, $\{D_k\}$, of the domain, A, determined by the splitting will correspond to the range of the time variable in the fullpath x in such a way that each element, D_k, of the partition will correspond to the value of the time variable in the state s_k and thus the values of the functions in $U|_k$ will correspond to the values of the QSIM variables in the state s_k.

The function $\psi_{U,M}$ referred to below is the bijection described in Section 3.1 which relates the variables in M with the variables in U.

The cardinality of $\{t_i\}$ must equal the number of time-point states in x. Therefore, since the states in x alternate between time-point states and time-interval states, $\langle x, c\rangle$ qualitatively describes $\langle\{t_i\}, U\rangle$ if and only if there is a correspondence between x and U in which s_k corresponds to $U|_k$ as follows:

- Given any landmark `X1` of a QSIM variable `X`$= \psi_{U,M}(u_i)$ and any nonnegative integer k, the qualitative value of `X` at the state s_k is `X1` if and only if $u_i|_{D_k} = c(\texttt{X1})$.
- $u_i'|_{D_k} > 0$ if and only if the qualitative derivative of the variable $\psi_{U,M}(u_i)$ in s_k is `inc`.
- $u_i'|_{D_k} = 0$ if and only if the qualitative derivative of the variable $\psi_{U,M}(u_i)$ in s_k is `std`.
- $u_i'|_{D_k} < 0$ if and only if the qualitative derivative of the variable $\psi_{U,M}(u_i)$ in s_k is `dec`.
- If the fullpath x is finite and QSIM has labeled the last state in x with `t=inf` then $t_J = \infty$.
- If the fullpath x is finite and QSIM has labeled the last state in x with `t<inf` then $t_J \neq \infty$.

Lemma 3 The specification $\langle x, c\rangle$ of x qualitatively describes $\langle\{t_i\}, U\rangle$ if and only if $\langle x^h, c\rangle$ qualitatively describes $\langle\{t_i : d_h \leq i\}, U|_{h+}\rangle$ for every nonnegative integer $h < \Lambda(x)$.

bf Proof: The proof comes straight from the definition of what it means for a specification of a fullpath to describe qualitatively a splitting of a set of functions. This definition involves a correspondence between s_h with D_h for every nonnegative integer $h < \Lambda(x)$.

It follows from this definition and the definition of the QSIM algorithm that if $\langle x, c\rangle$ qualitatively describes a splitting $\langle \{t_i\}, U\rangle$ of a set of reasonable, extended real valued functions, then given any pair of landmarks, `X0` and `X1`, of a QSIM variable `X`$= \psi_{U,M}(u_i)$ and any nonnegative integer k, if the qualitative value of `X` at the state s_k is the interval `(X0 X1)` then $u_i(D_k) \subset (c(\texttt{X0}), c(\texttt{X1}))$. It also follows that the state s_k has been determined by QSIM to be quiescent if and only if $u_i'|_{D_k} = 0$ for each $1 \leq i \leq n$.

Example 3 Let us consider our simple example again. We will show how the Guaranteed Coverage Theorem is satisfied in this example. Recall that we have the solution $U = \{\sin, \cos, -\sin\}$ to the structural abstraction of the equation

$$\frac{d^2x}{dt} = -x$$

where $\psi(\sin) = x, \psi(\cos) = v$ and $\psi(-\sin) = a$. We will show that some rooted fullpath in the structure represented by the output of the QSIM algorithm describes this set of functions.

We translated the structural abstraction into the QDE:

```
(d/dt X V)
(d/dt V A)
(M- A X).
```

If M is the tree produced by QSIM from this input, then $\psi_{U,M}(\sin) =$ `X`, $\psi_{U,M}(\cos) =$ `V` and $\psi_{U,M}(-\sin) =$ `A`.

Given the abstraction of the simple harmonic motion ODE, we complete the input to QSIM by describing the initial state, and giving instructions to QSIM such as "do not create new landmarks." (See Figure 2.) In order to make the output more interesting, we add a single additional landmark `X*` to the quantity space of the position variable. From this, QSIM generates the tree, M, shown in Figure 2.

It is a consequence of the Guaranteed Coverage Theorem that *some* rooted fullpath in $\widehat{M}$ has a specification which qualitatively describes some splitting of the sine function. Since we used weak matching, $\widehat{M}$ has infinitely many rooted fullpaths and not all of those have specifications which qualitatively describe a splitting of the sine function. If we had used strong matching, then $\widehat{M}$ would have exactly three rooted fullpaths, all of which are infinite and have specifications that qualitatively describe a splitting of the sine function.

To demonstrate the Guaranteed Coverage Theorem in this example, let's consider the rooted fullpath in $x \in \widehat{M}$ which cycles through the second behavior

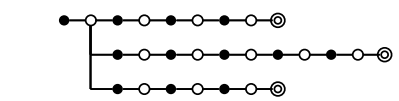
Figure 2: QSIM input and output for undamped spring

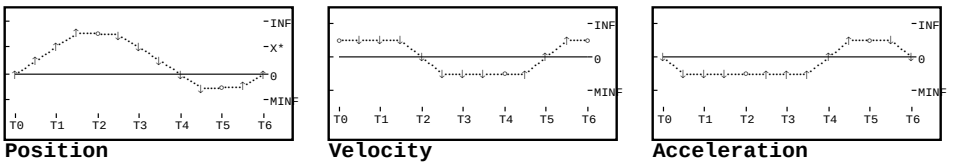
$$\ddot{x} + f(x) = 0$$

where $f \in M^+(x)$.

```
(define-QDE Spring-for-TL
  (quantity-spaces
   (X      (minf 0 X* inf)  "Position")
   (V      (minf 0 inf)     "Velocity")
   (A      (minf 0 inf)     "Acceleration"))
  (constraints
   ((d/dt X V))
   ((d/dt V A))
   ((M- A X)         (0 0) (minf inf) (inf minf))))

(defun spring-envisionment ()
  (setq SS (make-new-state :from-qde Spring-for-TL
                           :sim (make-sim :no-new-landmarks '(X V A)
                                          :cycle-detection :weak)
                           :assert-values '((X (0 nil))
                                            (V ((0 inf) nil)))))
  (qsim SS)
  (qsim-display SS))
```





Each path in the behavior tree ends in a cycle state. Each cycle state matches the initial state. The three behaviors are distinguished by their amplitude. The second behavior is shown.

(illustrated in Figure 2) infinitely many times. We must find a specification of x and a splitting of the sine function so that the two match as described in the definition of qualitative description. For the specification of x, we only need to find a function c which maps `X*` to some number between 0 and 1. Let's say, $c(\texttt{X*}) = 1/2$.

Let us now select a splitting of the sine function. We describe the set $\{t_i\}$ as the union of the following sets ordered by $<$:

$$\{\tfrac{\pi}{6} + 2k\pi | k \text{ is a positive integer}\},$$
$$\{\tfrac{k\pi}{2} | k \text{ is a positive integer}\} \text{ and}$$
$$\{\tfrac{11\pi}{6} + 2k\pi | k \text{ is a positive integer}\}.$$

Since $\lim_{x\to\infty} \sin x$ does not exist in R^*, the domain of our sine function is open on the right and so we are allowed to have an infinite set $\{t_i\}$ as a splitting.

Now it is easy to show that $\langle x, c\rangle$ qualitatively describes $\langle\{t_i\}, U\rangle$. The partition $\{D_k\}$ of the domain of sine determined by $\{t_i\}$ corresponds nicely with the domain of the time variable in x. Consider the landmark `X*` of `X`$=\psi_{U,M}(\sin)$. In the third state of x, `X` has the value `X*`. Notice that $D_2 = \{\frac{\pi}{6}\}$ and $\sin(\frac{\pi}{6}) = 1/2 = c(\texttt{X*})$ as required by the definition of qualitatively describes. It is easy to check that the other requirements are also satisfied.

4 QSIM and the Logic

The main theorem of this paper, stated informally, says the following: Suppose $\widehat{M}$ is a QSIM structure generated from the QDE C. If $\widehat{M}$ is a model for a CTL* formula, then the CTL* formula describes every solution to every ODE which abstracts to C. In order to state and prove this formally, we need two things. First, we need to explain what it means for a CTL* formula to describe a real-valued function. Second, we need to explain how temporal-logic propositions are checked in QSIM structures. We do this in reverse order since the latter is helpful in understanding the former.

In Section 4.1, we give the details of how model checking is applied to QSIM structures. In Section 4.2, we formalize the relationship between CTL* formulas and continuous functions and give an example.

4.1 QSIM Structures for CTL*

Given the structure $\widehat{M}$, the only thing needed to have a temporal structure as defined in Section 2.1 is an interpretation of propositions.

The *temporal structure* $\widehat{M_{TL}} = \langle S, X, L\rangle$, *represented by* a QSIM behavior tree $M = \langle S, R, B\rangle$, is obtained from $\widehat{M} = \langle S, X\rangle$ by the interpretation L of the propositions given in Figure 3 in which s represents the state over which the propositions are being interpreted.

This temporal structure $\widehat{M_{TL}}$ is the structure over which we will interpret CTL* formulas.

Figure 3: The propositional level of the language

`(qval` v `(`$qmag$ $qdir$`))` Suppose v is a variable of the state s, $qmag$ is a landmark or open interval defined by a pair of landmarks in the quantity space associated with v, and $qdir$ is one of {`inc`, `std`, `dec`}. This proposition is true when the qualitative derivative of v in s is $qdir$ and the qualitative magnitude of v in s is equal to or a subset of $qmag$.

`(status quiescent)` is true exactly when the qualitative derivative of each variable in the state is `std`.

`t=inf` is true at a state if QSIM was able to determine that the time variable in this state must be infinite.

`t<inf` is true at a state if QSIM was able to determine that the time variable in this state must be finite.

`(in-range` v `(`n_1 n_2`))` Suppose v is a variable in the state s and n_1 and n_2 are extended real numbers. If the value of var in s is a landmark then this proposition is true if and only if the number represented by that landmark in s is known to lie in an interval which is a subset of $[n_1, n_2]$. If the value of var in s is an interval `(X1 X2)`, then this proposition is true if and only if the interval $[n_1, n_2]$ contains both of the intervals in which QSIM has determined the numbers named by `X1` and `X2` to lie.

Our implementation, TL, of a model checking algorithm over QSIM structures, includes propositions in the language which are not mentioned in Figure 3 but are useful in practice. Since they add clutter to the statements of definitions and theorems in this paper, we will describe some of these operators in Appendix A and explain what adjustments need to be made to definitions and proofs in order to retain our theorems.

The propositions `t=inf` and `t<inf` allow the user to express the difference between, for example, "eventually in a possibly asymptotic sense" and "eventually in finite time". Alone, `eventually` really means "eventually in a possibly asymptotic sense". In order to express "eventually in finite time", use the propositions `t=inf` and `t<inf`. For example, we may say `(eventually (and` p `t<inf))` to mean that p becomes true in finite time.

The proposition `in-range` is sensible only in the states of behavior trees generated from a QDE containing some quantitative information. Simulation with quantitative information is handled by extensions to QSIM such as Q2 [14], Q3 [2] and NSIM [12]. The numbers referred to in these expressions are

extended real numbers: they may be `-inf` or `+inf` as well as real values. The use of the numeric propositions and quantitative information derived by QSIM from the numeric information given in the QDE, allows TL to prove time-related properties of physical systems.

The expressiveness of the application of CTL* to QSIM can easily be increased without adding to the complexity of model-checking by augmenting the propositional part of the language. See Appendix A for some such extensions.

4.2 Temporal Description

Here we define what it means for a CTL* perfect path formula Φ to describe a set of functions and we give an example of a simple CTL* formula and show that it describes the sine function.

If c is a specification of the landmarks mentioned in a perfect path formula Φ then we will call $\langle \Phi, c \rangle$ a *specification* of Φ. Let $\langle \{t_i\}, U \rangle$ be a splitting of a set of reasonable, extended real valued functions $U = \{u_i : 1 \leq i \leq n\}$ on a common domain A. Let $\{D_k\}$ denote the partition of A associated with $\{t_i\}$. Let ψ be a bijection from some subset T of U to the set of variables mentioned in the formula Φ. We recursively define what it means to say that $\langle \Phi, c \rangle$ *temporally describes* the splitting $\langle \{t_i\}, U \rangle$ *via* ψ. We assume that Φ is in positive normal form and so we make the definition according to the form of Φ as follows.

- If Φ is a proposition, then it must correspond to the splitting according to the following cases:
  - $\Phi =$ `(qval` $\psi(u_i)$ `(`*qmag qdir*`))` if and only if
    - if *qmag* is the landmark value `X0` of $\psi(u_i)$ then $u_i|_{D_0} = c($`X0`$)$,
    - if *qmag* is an interval `(X1 X2)` in the quantity space of $\psi(u_i)$ whose endpoints are landmark values of $\psi(u_i)$ then $u_i(D_0) \subset (c($`X1`$), c($`X2`$))$,
    - *qdir* = `inc` if and only if $u_i'|_{D_0} > 0$,
    - *qdir* = `std` if and only if $u_i'|_{D_0} = 0$ and
    - *qdir* = `dec` if and only if $u_i'|_{D_0} < 0$,
  - $\Phi =$ `(status quiescent)` if and only if $u_i'|_{D_0} = 0$ for each $1 \leq i \leq n$,
  - $\Phi =$ `t=inf` if and only if $D_0 = \{\infty\}$,
  - $\Phi =$ `t<inf` if and only if every element of D_0 is a real number,
  - $\Phi =$ `(in-range` $\psi(u_i)$ `(`n_1 n_2`))` if and only if $u_i(D_0) \subset [n_1, n_2]$,
- $\Phi =$ `(and` $p_1 \cdots p_m$`)` if and only if $\langle p_k, c \rangle$ temporally describes $\langle \{t_i\}, U \rangle$ via ψ for each $1 \leq k \leq m$,
- $\Phi =$ `(or` $p_1 \cdots p_m$`)` if and only if $\langle p_k, c \rangle$ temporally describes $\langle \{t_i\}, U \rangle$ via ψ for some $1 \leq k \leq m$,

- Φ = `(not` p`)` if and only if $\langle p, c\rangle$ does not temporally describe $\langle \{t_i\}, U\rangle$ via ψ,
- Φ = `(until` p q`)` if and only if for some nonnegative integer h, $\langle q, c\rangle$ temporally describes $\langle \{t_i : d_h \leq i\}, U|_{_{h+}}\rangle$ via ψ and for every nonnegative integer $l < h$, $\langle p, c\rangle$ temporally describes $\langle \{t_i : d_l \leq i\}, U|_{_{l+}}\rangle$ via ψ,
- Φ = `(releases` p q`)` if and only if for every nonnegative integer h such that $\langle q, c\rangle$ does not temporally describe $\langle \{t_i : d_h \leq i\}, U|_{_{h+}}\rangle$ via ψ, there is a nonnegative integer $l < h$ such that $\langle p, c\rangle$ temporally describes $\langle \{t_i : d_l \leq i\}, U|_{_{l+}}\rangle$ via ψ,
- Φ = `(next` p`)` if and only if $A = D_0$ or $\langle p, c\rangle$ temporally describes $\langle \{t_i : d_1 \leq i\}, U|_{_{1+}}\rangle$ via ψ and
- Φ = `(strong-next` p`)` if and only if $A \neq D_0$ and $\langle p, c\rangle$ temporally describes $\langle \{t_i : d_1 \leq i\}, U|_{_{1+}}\rangle$ via ψ.

Example 4 As an example, let us convince ourselves that the formula

```
(infinitely-often
   (before (qval X (0 dec))
           (qval X (0 inc))))))
```

temporally describes the sine function restricted to $[0, \infty)$. Let us call the formula under consideration Φ. We let the set U contain only the restricted sine function. The specification of Φ, in this case, is trivial: $c(0) = 0$. We will use the bijection ψ : $\sin(t) \mapsto$ `X`. The splitting for sine will be $\{t_i\}$ = $\{i\pi |$ where i is a nonnegative integer$\}$. Let $\{D_k\}$ denote the partition of $[0, \infty)$ associated with $\{t_i\}$. We want to convince ourselves that $\langle \Phi, c\rangle$ temporally describes $\langle \{t_i\}, U\rangle$ via ψ.

This amounts to proving that there are infinitely many nonnegative integers k satisfying the following conditions: (1) $\sin|_{_{D_k}} = 0$, (2) $\cos|_{_{D_k}} < 0$ and (3) there is some $l > k$ such that $\sin|_{_{D_l}} = 0$ and $\cos|_{_{D_l}} > 0$. Every positive odd integer satisfies these conditions so we are done.

5 The Main Results

The main theorem of this paper, which we can now almost state formally, says the following: Suppose M is a closed QSIM tree generated from the qualitative differential equation C. If $\widehat{M}_{TL}$ is a model for a universal CTL* formula `(necessarily` Φ`)`, then for every solution, U, to every ODE which abstracts to C, there is some splitting $\langle \{t_i\}, U\rangle$ of U and some specification $\langle \Phi', c\rangle$ of the perfection, Φ', of Φ such that $\langle \Phi', c\rangle$ temporally describes $\langle \{t_i\}, U\rangle$ via $\psi_{_{U,M}}$. Therefore, TL is sound. In this section, we prove this theorem, discuss some corollaries and also prove a more limited completeness result.

There is one more hypothesis which needs to be present in the main theorem. This hypothesis is usually satisfied by QSIM structures but still must be mentioned. It is possible, for some propositions, that QSIM may not determine all of the information needed to use that proposition with confidence. Section 5.1 explains this notion and contains a theorem that relates the qualitative and temporal descriptions of a set of functions as defined in Sections 3.2 and 4.2, respectively. The remainder of this section contains the theorems which are the most important for applications.

5.1 Determined QSIM Trees

Suppose that x is a fullpath in a closed QSIM structure $\widehat{M}$ and further that $\widehat{M}_{TL}, x \models \Phi$ where Φ is a perfect path formula. Suppose that $\langle x, c\rangle$ is a specification of x which qualitatively describes $\langle \{t_i\}, U\rangle$. In this section, we will prove that $\langle \Phi, c\rangle$ temporally describes $\langle \{t_i\}, U\rangle$ via $\psi_{U,M}$.

In order to prove this, we need to be certain that QSIM determines the information in propositions completely and correctly. Otherwise, the induction step in the proof of Theorem 2 does not work. Formally, we need to know that if s is a state in a QSIM tree M and $\widehat{M}_{TL}, s \models \phi$ where ϕ is an *atomic proposition* then for every fullpath, y, starting at s, $\langle y, c\rangle$ qualitatively describes $\langle \{t_i\}, U\rangle$ *if and only if* $\langle \phi, c\rangle$ temporally describes $\langle \{t_i\}, U\rangle$ via $\psi_{U,M}$. This is the conclusion of Lemma 4. In an arbitrary QSIM tree, this may not be true, although exceptions are not common. The only case that arises in the language of the body of this paper occurs in a transition state at which it is impossible to determine whether `t=inf` or `t<inf`. For example, consider the QDE $x' = f(x)$, where $f \in M_0^+$ (that is, f is a monotonically increasing function with $f(0) = 0$). With an initial state $x(t_0) > 0$ the behavior diverges, terminating at a qualitative state where $qmag(x)$ is $\langle \texttt{inf}, \texttt{inc}\rangle$, which is a transition state. However, some choices of f (e.g., $f(x) = x^2$) imply that $x(t)$ becomes infinite at finite time, while others (e.g., $f(x) = x$) imply that $x(t)$ becomes infinite only at infinite time, so the time label for the transition state is undetermined.

Therefore, we define a QSIM state to be *determined* with respect to the propositions `t=inf` and `t<inf` if QSIM has determined one of `t=inf` or `t<inf`. With respect to the other propositions we have defined, all QSIM trees are determined. However, when we define new propositions, this issue needs to be addressed. That is to say, when one defines a new proposition, one needs to define what it means to be determined with respect to that proposition in such a way that the proof of Lemma 4 goes through as well as the induction step in Theorem 2. This can be a subtle point as you can see in Appendix A.

The TL program can warn the user about any state which is not determined with respect to an atomic proposition being queried on that state. When a state is not determined, the TL program still operates but the hypotheses of the theorems relating the operation of TL with the reasonable, extended real valued functions are no longer satisfied.

Lemma 4 If s is a state in a QSIM tree M which is determined with respect to the proposition ϕ and $\widehat{M}_{TL}, s \models \phi$, then for every fullpath, x, starting at s, if $\langle x, c\rangle$ is a specification of x, then $\langle x, c\rangle$ qualitatively describes $\langle \{t_i\}, U\rangle$ if and only if $\langle \phi, c\rangle$ temporally describes $\langle \{t_i\}, U\rangle$ via $\psi_{U,M}$.

bf Proof: The result follows directly from the definition of the semantics of the proposition (given in Figure 3), the definition of the meaning of a specification of a formula temporally describing a splitting of a set of functions, and the definition of the meaning of a specification of a fullpath qualitatively describing a splitting of a set of functions. Notice that the determinedness hypothesis is needed in the part of the proof involving the propositions `t=inf` and `t<inf` because if QSIM does not determine this information, the proof fails.

We say that a QSIM structure $\widehat{M}$ is determined with respect to a proposition if every state in $\widehat{M}$ is determined with respect to the proposition.

Theorem 2 relates the two ways of describing a set of reasonable, extended real valued functions and will be used in the proofs of the main theorems of this paper.

Theorem 2 Suppose x is a fullpath in a QSIM structure $\widehat{M}$ which is determined with respect to all of the propositions in Φ, a perfect path formula, and $\widehat{M}_{TL}, x \models \Phi$. If the specification $\langle x, c\rangle$ of x qualitatively describes $\langle \{t_i\}, U\rangle$ then $\langle \Phi, c\rangle$ temporally describes $\langle \{t_i\}, U\rangle$ via $\psi_{U,M}$.

The proof is complex and not enlightening. Therefore, it has been put into Appendix C.

There are two reasons we did not simply make this theorem the definition of a temporal description. First, when we say that a formula describes a function, we want to be talking about a formula and a function without an intervening QSIM structure. Second, the subtleties involved in the definition of determined which are brought to light when one tries to prove Lemma 4 and Theorem 2 might be missed if the definition of temporally describes were given at such a high level. See Appendix A for an example of this.

5.2 Main Theorems for Universal Formulas

This section contains the main results of this paper. As a consequence of the main theorems, the user of the TL and QSIM systems may prove temporal statements about dynamical systems as follows. First, the user constructs a QDE, C, and uses QSIM to generate a closed tree, M. Then the user may use TL to check if a universal formula, Φ, is modeled by $\widehat{M}_{TL}$. If it is, then the user has proved that the perfection, Φ', of Φ describes the solution to any differential equation which abstracts to C.

Theorem 3 Let U be a solution to the structural abstraction of any ODE which abstracts to the QDE, C. Suppose QSIM generates the closed tree M from $\langle C, I\rangle$. Let Φ be a universal formula in CTL*. If $\widehat{M}_{TL}, I \models$ `(necessarily` Φ`)`, then there is a specification c of the landmarks mentioned in Φ' such that $\langle \Phi, c\rangle$ temporally describes some splitting $\langle \{t_i\}, U\rangle$ of U via $\psi_{U,M}$.

bf Proof: Let Φ be a universal path formula and M a closed QSIM behavior tree which is determined with respect to the propositions in Φ. Suppose $\widehat{M}_{TL}, I \models$ `(necessarily` Φ`)`. Let U be as in the hypotheses. By the Guaranteed Coverage Theorem, we know that there is a rooted fullpath y_U in $\widehat{M}_{TL}$ and a specification, $\langle y_U, c\rangle$, of y_U such that $\langle y_U, c\rangle$ qualitatively describes some splitting $\langle \{t_i\}, U\rangle$ of U. By Lemma 2, $\widehat{M}_{TL}, y_U \models \Phi'$. Since M is determined with respect to the propositions in Φ, and $\widehat{M}_{TL}, y_U \models \Phi'$, we use Theorem 2 to conclude that $\langle \Phi', c\rangle$ temporally describes $\langle \{t_i\}, U\rangle$ via $\psi_{U,M}$.

The following corollary follows from the proof of Theorem 3.

Corollary 1 Let U be a solution to the structural abstraction of any ODE which abstracts to the QDE, C. Suppose QSIM generates the closed tree M from $\langle C, I\rangle$. Let Φ be a universal formula in CTL*. If $\widehat{M}_{TL}, I \models$ `(necessarily` Φ`)`, then there is a rooted fullpath $x \in X_r$ and a specification, $\langle x, c\rangle$, of x consistent with the information derived by QSIM on the fullpath x, such that $\langle \Phi, c\rangle$ temporally describes some splitting $\langle \{t_i\}, U\rangle$ of U via $\psi_{U,M}$.

We include this corollary in the discussion because it provides more information about the specification c of the landmarks mentioned in the formula. This tells us that the specification must be the specification of some fullpath in $\widehat{M}$ which qualitatively describes the splitting of U.

The conclusion of the theorem states that the *perfect* path formula related to the universal formula describes the solutions to the equations. Quantifier nesting is irrelevant, as far as the conclusions of this theorem are concerned. While there are situations in which nested quantifiers are useful, such as gaining insight into some detail of the QSIM structure (see Section 6.2), these applications do not rely on the main point of our theorems, i.e. the relation between the QSIM prediction and the underlying dynamical system.

Therefore, if the user is using TL only for the purpose of proving that a formula temporally describes the solutions to an ODE, then he or she may as well enter a formula of the form `(necessarily` Φ`)` where Φ is a perfect path formula.

5.3 Numeric Queries

The previous discussion is particularly relevant to queries involving numeric information. It has been mentioned that the numeric information which QSIM

derives about landmarks may vary across behaviors. QSIM keeps track of numeric information with respect to QSIM behaviors, not with respect to states. This fact makes a more specific form of Theorem 3 desirable.

To check a proposition involving numeric information (such as `in-range`) we must know which fullpath the state being checked is in. Furthermore, that fullpath must be associated with a particular QSIM behavior in M so that numeric information can be retrieved with respect to that QSIM behavior. This problem is solved by using the function z defined in Section 3.1.

The following corollary is simply a special case of Corollary 1 in which we can also specify that the specification is consistent with the numeric information on some behavior of the QSIM tree.

Corollary 2 Let U be a solution to the structural abstraction of any ODE which abstracts to the QDE, C. Suppose QSIM generates the closed tree M from $\langle C, I\rangle$ using strong-match and no cross-edge cycle detection. Let Φ be a universal formula in CTL*. If $\widehat{M}_{TL}, I \models$ `(necessarily` Φ`)`, then there is a rooted fullpath $x \in X_r$ and a specification, $\langle x, c\rangle$, of x consistent with the information derived by QSIM on the behavior $z(x)$, such that $\langle \Phi, c\rangle$ temporally describes some splitting $\langle \{t_i\}, U\rangle$ of U via $\psi_{U,M}$.

The proof of this corollary is exactly the same as the proof of Theorem 3. The difference is that since we specified the type of cycle detection, we know that the function z is defined and can use it to obtain numeric information.

5.4 Completeness Results

Suppose the user has generated a QSIM tree, M, from $\langle C, I\rangle$ and has an interesting CTL* perfect path formula Φ. We know that if $\widehat{M}_{TL}, I \models$ `(necessarily` Φ`)`, then the solution to any ODE which abstracts to C is described by Φ. But suppose $\widehat{M}_{TL}, I \models$ `(necessarily` Φ`)` is false but the user wants to know if there is *some* solution, U, to *some* ODE which abstracts to C, such that Φ describes U. The user might test the formula `(possibly` Φ`)`. If this formula is modeled by the QSIM temporal structure then the user still cannot, in general, conclude that there is a solution, U, to an ODE which abstracts to C, such that Φ describes U. This is so becuase the QSIM temporal structure may have a rooted fullpath which is "spurious", i.e. a fullpath which does not describe any solution to any ODE which abstracts to C.

In this section, we provide some circumstances under which the user may draw positive conclusions from a formula of the form `(possibly` Φ`)` where Φ is a perfect path formula.

Suppose $\widehat{M}$ is a closed QSIM structure generated from a QDE and initial state $\langle C, I\rangle$. We sometimes would like to know whether there is any ODE, F, which abstracts to C whose solution is described by some given perfect path formula. In order to do this, the QSIM tree must be closed, determined with respect to the propositions in Φ and satisfy the following completeness condition.

Definition 5 We call a closed QSIM behavior tree, M, *complete* if for every rooted fullpath, x in $\widehat{M}$ there is an ODE with structural abstraction, F', which abstracts to the input QDE and a splitting of the solution to F' which is qualitatively described by some specification of x.

In other words, a closed tree is complete if every rooted fullpath in $\widehat{M}$ describes some solution to an ODE which abstracts to C.

One way to check for the completeness of a tree is to prove, either mathematically or by numeric simulation, that there is a reasonable, extended real valued solution corresponding to each fullpath in the structure represented by the tree.

Under these conditions, the user is able to draw sound conclusions about the solution so *some* (but not *every*) ODE which abstracts to the QDE as in the scenario described above. Theorem 4 details this result.

Theorem 4 Suppose Φ is a perfect path formula in CTL*. Suppose M is a closed, complete QSIM behavior tree generated from the QDE and initial state $\langle C, I\rangle$ and determined with respect to the propositions in Φ. If $\widehat{M}_{TL}, s \models$ `(possibly Φ)`, then there is an ODE, F, whose structural abstraction, F', has solution U and abstracts to C and there is a specification of Φ which temporally describes some splitting of U via $\psi_{U,M}$.

bf Proof: Let Φ be a perfect path formula in CTL*, M a closed, complete QSIM behavior tree generated by the QDE and initial state $\langle C, I\rangle$. Suppose that M is determined with respect to the propositions in Φ and $\widehat{M}_{TL}, s \models$ `(possibly Φ)`.

We want to show that there is an ODE, F, whose structural abstraction, F', has solution U and abstracts to C and there is a specification $\langle \Phi, c\rangle$ which temporally describes a splitting of U via $\psi_{U,M}$.

We know from the semantics of CTL* that there is a fullpath x in $\widehat{M}_{TL}$ such that $\widehat{M}_{TL}, x \models \Phi$. Because M is complete, we know that there is a set of reasonable, extended real valued functions $U = \{u_i : 1 \leq i \leq n\}$ such that U is a solution to the structural abstraction of some ODE which abstracts to C and $\langle x, c\rangle$ qualitatively describes a splitting of U for some specification $\langle x, c\rangle$ of x.

Therefore, $\langle \Phi, c\rangle$ temporally describes this fixed splitting of U via $\psi_{U,M}$ by Theorem 2.

6 Applications of CTL* and QSIM

TL is the name of a CTL* model-checker customized for use with QSIM. The current implementation replaces the experimental versions described and used in previous publications [15, 17]. The underlying model-checking algorithm is that of Bhat, Cleaveland and Grumberg [3]. Bhat, Cleaveland and Grumberg

prove that this algorithm has the same complexity as the best known global algorithms for both CTL* and CTL. Their algorithm has the added advantage of being "on-the-fly" rather than "global"; i.e. it is possible for the algorithm to halt with the correct answer without constructing the entire exponentially-large structure required to check some formulas in CTL*.

Temporal reasoning may be useful any time QSIM is used. QSIM has been used to simulate controllers, human organs and disease, abstract and real physical systems, electrical circuits, population dynamics, chemical reactions, etc. [16]

TL can be used to prove that a QSIM tree is closed with the following query

```
(TL R (necessarily
        (eventually (or (status quiescent)
                        (status cycle)
                        (status transition)))))
```

where `R` is the root of the tree. (See Appendix A for an explanation of arguments to the `status` proposition other than `quiescent`.)

TL automatically reports when an atomic proposition is checked on a state in which that proposition is not determined.

6.1 Examples

First, we demonstrate the use of TL to ask and answer questions about some simple models: the undamped oscillator, whose behavior tree (Figure 2) is rooted in the initial state `SS`; and the damped oscillator, whose behavior tree (Figure 4) is rooted in the state `DS`.

Example 5 (Undamped Oscillator) The simple spring conserves energy, so all behaviors end in cycles, as shown by the behavior tree in Figure 2. Therefore, the closedness query would return T. The three behaviors differ according to whether the amplitude of the oscillation passes a predefined landmark value, `X*`. The queries shown demonstrate that the solution to any ODE consistent with the QDE in Figure 2 never becomes quiescent, always reaches a cycle state, and necessarily has an infinite sequence of events crossing $x = 0$ in opposite directions. (Since the variable `X` can have only one qualitative value in a state, the last two formulas below are equivalent.)

```
(TL SS (necessarily
         (always (not (status quiescent)))))
  => T

(TL SS (necessarily (eventually (status cycle))))
  => T

(TL SS (necessarily (and (infinitely-often (qval X (0 inc)))
                         (infinitely-often (qval X (0 dec))))))
```

```
  => T

(TL SS (necessarily
         (infinitely-often
            (before (qval X (0 dec))
                    (qval X (0 inc))))))
  => T
```

Since the simple spring tree is closed and determined, we have shown that every reasonable solution to an ODE which abstracts to the QDE in Figure 2 has a splitting which is temporally described by a specification of the perfect path formula associated with each of the formulas above.

The predicted tree is not complete, since behaviors that cycle through different branches are not possible. We could rewrite the QDE in various ways to make the tree complete. Simply removing the extraneous landmark in `X` would suffice. This would produce a single behavior. Using the strong-match cycle criterion would also produce a complete tree in this case. The next example produces a complete tree.

Example 6 (Damped Oscillator) The damped spring loses energy. The first behavior in the behavior tree in Figure 4 ends in a cycle representing a decreasing oscillation. The second two are partial cycles followed by "nodal" (i.e. over- or critically-damped) convergence to a quiescent state at the origin. These qualitative behaviors have specifications which qualitatively describe real trajectories of nonlinear instances of the QDE. Since weak-match cycles were detected, this finite behavior tree represents a structure with infinitely many rooted fullpaths, oscillating a finite number of half-cycles around the origin before "nodal" convergence and a single rooted fullpath which never becomes quiescent. TL determines that each of the universal questions asked about the simple spring behavior tree above is false of the damped spring, but the corresponding existential statements are true.

```
(TL DS (possibly (always (not (status quiescent)))))
  => T

(TL DS (possibly (eventually (status cycle))))
  => T

(TL DS (possibly (eventually (status quiescent))))
  => T

(TL DS (possibly (and (infinitely-often (qval x (0 inc)))
                      (infinitely-often (qval x (0 dec))))))
  => T

(TL DS (possibly
```

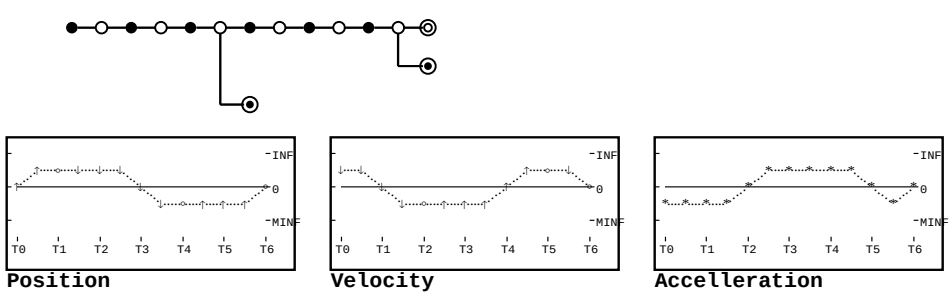
Figure 4: QSIM input and output for damped spring

$$\ddot{x} + g(\dot{x}) + f(x) = 0$$

where f and g are in $M^+(x)$.

```
(define-QDE DSpring-for-TL
  (quantity-spaces
   (x     (minf 0 inf)   "Position")
   (v     (minf 0 inf)   "Velocity")
   (a     (minf 0 inf)   "Acceleration")
   (ff    (minf 0 inf)   "Fluid friction")
   (fs    (minf 0 inf)   "Spring force"))
  (constraints
   ((d/dt x v))
   ((d/dt v a))
   ((m- x fs)      (0 0)   (minf inf)  (inf minf))
   ((m- v ff)      (0 0)   (minf inf)  (inf minf))
   ((add fs ff a))))

(defun dspring-envisionment ()
  (setq DS (make-new-state
               :from-qde DSpring-for-TL
               :sim (make-sim :no-new-landmarks '(x v a ff fs)
                              :ignore-qdirs '(a)
                              :cycle-detection :weak
                              :state-limit 200)
               :assert-values '((x (0 nil)) (v ((0 inf) nil)))))
  (qsim DS)
  (qsim-display DS))
```



The first behavior in the tree ends in a cycle state which matches the root. The difference between the second two states is the direction from which they approach quiescence. The second behavior is shown.

```
        (infinitely-often
          (before (qval x (0 dec))
                  (qval x (0 inc))))))
  => T

(TL DS (necessarily
        (always (possibly (eventually (status quiescent))))))
  => T

(TL DS (necessarily
        (always (implies (not (status quiescent))
                  (possibly (always (not (status quiescent))))))))
  => T
```

The damped spring structure is complete, since there are nonlinear choices for the two monotonic functions in the model that give "spiral in" behavior away from the origin, followed by "nodal" behavior close to the origin. If both monotonic functions are linear, of course, the only possibilities are pure "nodal" and pure "spiral in" behavior. Therefore, we have proved that for each of the first five formulas above, there is a set of functions which is a solution to an ODE which abstracts to the QDE given in Figure 4 and is temporally described via $\psi_{U,M}$ by a specification of the perfect path formula corresponding to the CTL* formula.

The last two formulas say that, no matter how many oscillations you've seen so far, it is always possible that (a) the behavior could terminate with nodal convergence to a quiescent state, and (b) the behavior could go on oscillating forever. Since the last two formulas are not universal, Theorem 3 gives no information. These two formulas are used for the purpose of discovering features of the QSIM structure, $\widehat{M}$, and not for proving properties of dynamical systems.

Example 7 (A Quantitative Example) Now we reconsider the billiards example in order to show the use of quantitative information in QSIM QDEs and the resulting proofs that TL provides for time-critical systems. Refer to Figure 1 and the description of the system given in Section 3.1.

With only qualitative information about the balls' positions and speeds, QSIM gives a closed tree with 55 different behaviors, representing the different orders in which balls can collide, pass each other's positions, or reach infinity. With complete quantitative information, specifying identical speeds of 2 position units per second (in the indicated directions), QSIM predicts a single behavior in which balls `B` and `C` collide at $t = 0.5$ seconds. With the information provided, QSIM produces a tree with three behaviors.

The queries given below were checked on the tree with the information described in Section 5.3—`A` and `B` have velocity -2 units/second, and the velocity, `Cxp`, of `C` is some constant within the interval $[-3.5, -1.5]$ in units/second.

Because the predicted tree of behaviors is complete, we can draw conclusions from the answers to each of the following queries.

```
(TL SS (necessarily
         (always (implies (and (in-range dxBC (0 0))
                               (in-range dyBC (0 0)))
                   (in-range time (.5 .5))))))
  => T

(TL SS (possibly
         (eventually (and (in-range dxBC (0 0))
                          (in-range dyBC (0 0))
                          (in-range time (.5 .5))))))
  => T

(TL SS (necessarily
         (always (not (and (in-range dxAC (0 0))
                           (in-range dyAC (0 0)))))))
  => T
```

The first TL query proves that in the solution to any ODE which abstracts to the QDE if `B` and `C` collide then it happens at time 0.5. The second query proves (by Theorem 4) that there is a solution to an ODE consistent with the given QDE in which `B` and `C` do collide at time 0.5. The third query shows that `A` and `C` cannot collide in any ODE consistent with the given QDE. This is so because `B` blocks `C` in the behavior in which `C` might hit `A`.

6.2 TL as a Debugging Tool for QSIM Models

Because QSIM is not complete in general, a QSIM behavior tree may contain paths which do not correspond to real behaviors. Therefore, the truth of certain CTL* statements (e.g. those beginning with the quantifier `possibly`), do not imply the truth of the corresponding statement in an actual behavior. This provides an opportunity for a tool such as TL to be used to find such paths. If the QSIM user knows that a certain sequence of events cannot occur in a real behavior, he can use TL to find out if that sequence of events occurs in any of the paths in the QSIM behavior tree. The user can have TL print information which will isolate the path on which the spurious behavior occurs. Also, as in the damped spring example, nested quantifiers can be used to gain insight into some interesting structures of the represented QSIM structure.

The program can be and has been used on terminals which do not support the graphical display of QSIM behavior trees. In these circumstances, the user can learn everything he or she may need to know about a QSIM behavior tree by evaluating a few carefully chosen CTL* statements.

6.3 Proving Properties of Controllers

Kuipers & Åström [15] have used TL and QSIM to prove properties of heterogeneous control laws. A heterogeneous controller is a nonlinear controller created

by the composition of local control laws appropriate to different, possibly overlapping, operating regions. Such a controller can be created in the presence of incomplete knowledge of the structure of the system, the boundaries of the operating regions, or even the control action to take. A heterogeneous control law can be analyzed, even in the presence of incomplete knowledge, by representing it as a qualitative differential equation and using qualitative simulation to predict the set of possible behaviors of the system. By expressing the desired guarantee as a statement in CTL*, the validity of the guarantee can be automatically checked against the set of possible behaviors. Kuipers & Åström [15] demonstrate the design of heterogeneous controllers, and prove certain useful properties, first for a simple level controller for a water tank, and second for a highly nonlinear chemical reactor.

Evi Gazi and Lyle Ungar also use TL to prove properties of models of chemical reaction controllers [10, 9].

There are three programs—Q2 [14], Q3 [2] and NSIM [12]—which extend QSIM to take advantage of numeric information, to prune spurious behaviors and to derive numeric bounds on landmark values and time-points. The program TL is easily applied to the behavior trees output by these QSIM extensions which use quantitative bounding information and produce quantitative bounds on the predictions. For these applications we use the propositional part of the language with the numeric propositions to include numerical information in the state propositions. These propositions allow TL to prove time-critical properties of models of a system, even in the face of incomplete knowledge.

6.4 TeQSIM: Temporal Constraints on Simulation

In this paper, we use temporal logic formulas to check the *output* of QSIM. Brajnik and Clancy [5, 6, 4] extend the interaction between qualitative simulation and model-checking to treat temporal logic statements as an *input*. TeQSIM (pronounced *tek'sim*) interleaves model-checking with QSIM's simulation agenda, allowing simulation only of branches that can satisfy the given temporal logic formula. This makes it possible to focus simulation on a particular portion of the state space, which is useful for large, complex models that might not otherwise be tractable. It also allows the user to specify exogenous inputs, discontinuous changes, the results of observations, and various other types of boundary conditions. One can use temporally guided simulation to explore critical portions of a large state space to discover, for example, constraints on an exogenous variable required for a plan to succeed, followed by unguided simulation of a model incorporating the new constraints to derive a performance guarantee. Brajnik and Clancy [5, 6] demonstrate TeQSIM on a realistic control and planning problem from the domain of water supply management.

7 Relation to other Work

The results described in this paper are related to other work done in the fields of temporal-logic model-checking and simulation and control.

Probably the most work in temporal-logic model-checking has been done in applications of CTL and CTL* to computer processes such as parallel computing [8, 18]. More closely related work has been done by Moon, *et al* [19] who checked statements in CTL against state transition graphs in discrete-time systems generated from programmable logic controller ladder diagrams. Their specific application was to chemical process control. TL makes it possible to apply a more complex temporal logic (CTL*) to continuous-time control systems, and indeed to dynamical systems in general.

Alur and Henzinger [1] use a logic called Metric Temporal Logic (MTL) to check properties of discrete event systems. Metric Temporal Logic is, strictly speaking, not as expressive as CTL*. However, it integrates time information at a higher level of the language, therefore it is easy to express some statements in MTL which are difficult to express in CTL*.

Jahanian [11] modeled real time systems in the Modechart language. Statements in Real Time Logic were checked against a Modechart model. Real Time Logic is undecidable in general but certain classes of statements are shown to be decidable. Model-checking CTL* is decidable [8]. However, Real Time Logic is especially suited for expressing statements which are useful in time-critical systems, whereas some such statements are more difficult to make in CTL*.

Other systems exist which allow temporal-logic sentences to be checked against a structure representing discrete event systems. TL makes a formal connection between continuous dynamical systems and time-critical temporal-logic model-checking.

8 Conclusion

TL implements a method for using modal- and temporal-logic formulas to prove properties of the behavior of a continuous physical system even with an incomplete, qualitative or semi-quantitative description. If the user can describe a physical system in terms of a set of qualitative constraints, then by using QSIM and TL, he or she can prove theorems about the behavior of any reasonable, extended real valued function consistent with those constraints. This applies even to systems with time-critical requirements. This provides a meaningful and sound interpretation for the phrase, "proof by simulation."

This link between logic-based and simulation-based inference methods will support a variety of hybrid reasoning techniques that could be of substantial value for the design and validation of continuous and piecewise-continuous systems.

A Extensions to the Propositional Language

The implementation, TL, of the language includes other propositions, some of which we describe in this appendix. In most cases, the added propositions are useful only to describe the predicted QSIM structure, and not to prove theorems about the underlying dynamical systems. Some of them can be included in the proof of the theorems but this inclusion would require distracting special treatment. This appendix discusses issues involved in adding new propositions to the language.

First we mention the "proposition" of the form `(funcall` f`)` where f is a lisp function. This returns the value returned by the lisp function called with the state as its single argument. This is used mainly for side effects such as printing information about a state. Note that the proposition `funcall` cannot be considered as part of the logic when we talk about complexity, soundness or the main theorems. It is added for user-extensions and convenience and should be used with care.

The next proposition we mention illustrates the major issues involved in adding a proposition to the language. The syntax is:

`(contains-range` v `(`n_1 n_2`))`

where v is a variable name in the state s and n_1 and n_2 are extended real numbers. This proposition is true when the numeric range, in which the number named by v in the current behavior has been determined to lie, contains the interval $[n_1, n_2]$ as a subset.

If the theorems in the paper are going to be applied to a new proposition, then we must be able to include it in the proof of Lemma 4 in such a way that the induction step in the proof of Theorem 2 can be performed. Therefore, we must define what it means for a specification of this proposition to *describe temporally* a splitting of a set of reasonable, extended real valued functions and determine what is required for a state to be *determined* with respect to the proposition. Once Lemma 4 and Theorem 2 are proved for the proposition, the rest of the theorems will follow.

We say that a specification of the proposition

`(contains-range` $\psi(u_i)$ `(`n_1 n_2`))`

temporally describes $\langle \{t_i\}, \{u_i : 1 \leq i \leq n\} \rangle$ via ψ if and only if $[n_1, n_2] \subset u_i(D_0)$ where $\{D_k\}$ is the partition of the domain corresponding to the splitting.

The reader might want to try to prove Lemma 4 and Theorem 2 at this point in order to see the problem which now arises. Without a strict definition of what it means for a state to be determined with respect to this new proposition, the proof does not go through. In fact, Theorem 2 is false without such a definition. Since the set U of functions is fixed, we cannot prove that the

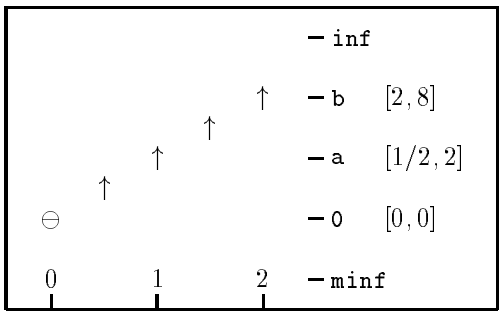


Figure 5: Qualitative behavior with quantitative landmarks bounds.

subrange $[n_1, n_2]$ specified in the proposition contains the value of the specific function u_i. I.e. it is possible that a fullpath x describes a set of functions and a path formula containing the `contains-range` proposition be modeled by x but the path formula may not describe the set of functions.

Consider the following as a counterexample. Suppose that the variable `V` in the QDE has real-valued function solutions $u(x) = rx^2$ for $r \in [1/2, 2]$ due to the constraints and numeric information provided by the user. Further, suppose that `V` has the quantity space `(minf 0 a b inf)` where the real value named by the landmark `a` is known to fall within the interval $[1/2, 2]$ and the real value named by the landmark `b` is known to fall in the interval $[2, 8]$ on the behavior in Figure 5. Suppose that, in a certain path (Figure 5), at time $t = 0$ we have `(qval V (0 std))`, for time $t \in (0, 1)$ we have `(qval V ((0 a) inc))`, at time $t = 1$ we have `(qval V (a inc))`, for time $t \in (1, 2)$ we have `(qval V ((a b) inc))` and at $t = 2$ we have `(qval V (b inc))`. The following formulas will be true on this fullpath yet they describe functions which are not real solutions to the QDE:

```
(eventually
  (and (contains-range V (1/2 1/2))
       (strong-next (strong-next (contains-range V (8 8))))))

(eventually
  (and (contains-range V (2 2))
       (strong-next (strong-next (contains-range V (2 2))))))
```

These path formulas describe real-valued functions which cut across the ranges in a way which an actual solution to the QDE could not do. In this case, the use of nontrivial range information and the `contains-range` proposition, can combine to describe a function which is spurious.

In order for Lemma 4 and Theorem 2 to hold in general, for a QSIM state

to be *determined* with respect to (`contains-range` v (n_1 n_2)), the numeric range associated with a landmark of the variable v in the state must be trivial (i.e. contain a single point).

Other propositions such as `intersects-range` might be useful to the TL user. However, because of the strict condition required for determinedness, such propositions are intended to be used more for gaining information about the QSIM prediction than proving theorems about continuous systems. The exception, of course, is in the case that there is such complete information that programs such as Q2 are able to narrow the possible values associated with a landmark to a single real number. In this case, `contains-range` and other similar propositions may be used to prove theorems about continuous systems but they become equivalent to the `in-range` proposition.

Finally, we mention that the `status` proposition can take any of the arguments {`quiescent`, `stable`, `unstable`, `transition`, `cycle`}. The proposition will be true when QSIM has determined the state to have the named property. A state has the `stable` property if it is quiescent and in stable equilibrium. A `transition` state is a terminal state in a path in which the value of one of the variables crosses a boundary of its range. A `cycle` state is a state which matches a previously generated state and whose successors are already represented in the tree. Other than `quiescent, stable` and `unstable` these properties have more to do with the QSIM interface than with the underlying functions being described. That is the reason we did not include a discussion of these properties in the discussion of the logic.

We define a QSIM state to be *determined* with respect to the propositions `(status stable)` and `(status unstable)` if it is not quiescent or if it has been determined to be stable. This is because QSIM may be incorrect when it determines a quiescent state to be `unstable`.

Additional propositional operators can also be added to allow the user to gain other information about QSIM states. For example, the newest release of QSIM produces "chatter-sink" states in order to express more succinctly the fact that certain variables may chatter indefinitely or, at some point, stop chattering. So, we could add the proposition `chatter-sink-p` to the language such that it is true of a state if and only if the state is a chatter-sink state. In this case, the proposition has no real meaning when translated to the domain of real-valued functions. Therefore, once again, it is used mainly to draw information about QSIM's output.

B Refined Definition of a Reasonable Function

In this appendix, we use R to denote the reals and R^* to denote the extended reals.

The traditional definition of a reasonable function [16] is too restrictive for our current purposes. Giving a satisfactory definition of reasonable is not simple.

We would like to let functions such as sine on $[a, \infty)$ and tangent on $[-\pi/2, \pi/2]$ to be reasonable. On the other hand, we do not want to allow functions to be reasonable which cannot be simulated by QSIM. Finding a balance between including functions which QSIM does simulate and excluding functions which make simulation impossible is an area open to further investigation. The definition must be such that the QSIM algorithm simulates every reasonable solution to any ODE which abstracts to the input. However, we want it to be inclusive enough to cover interesting functions.

The following questions come up in this context. Should we allow infinite derivatives at points in R? Should we allow the limits of f' not to exist at $\pm\infty$? Should we allow infinitely many critical points in R? Is there a concise way of expressing the definition which gives us the best of both worlds?

The definition we offer here is adequate for the purposes of this paper. This definition is more inclusive than the traditional one [16], but more inclusive definitions are possible.

Definition 6 Suppose A is an interval in R^* with supremum b and infimum a. $f : A \to R^*$ is a reasonable function over A if

1. f is continuous on A,
2. f is continuously differentiable on (a, b) with derivative f',
3. f has only finitely many critical points in any bounded interval of $R \cap A$ and
4. if $a \in R$ then $\lim_{t \to a+} f'(t)$ exists in R^* and if $b \in R$ then $\lim_{t \to b-} f'(t)$ exists in R^*.

According to this definition, sine is reasonable on any interval $[a, \infty)$ for $a \in R$, but not over $[a, \infty]$. Tangent is reasonable over $[-\pi/2, +\pi/2]$, but $\sin(1/x)$ and $x \sin(1/x)$ are not reasonable over $(0, a)$ for $0 < a < \infty$.

C Proofs

Proof of Lemma 1. We want to show that if Φ is a universal formula and x is a fullpath in M such that $M, x \models \Phi$, then $M, x \models \Phi'$.

Without loss of generality, we assume that Φ is in positive normal form. The proof of the lemma is by induction on the length of Φ.

Since atomic propositions are perfect, $\Phi = \Phi'$ and hence $M, x \models \Phi$ if and only if $M, x \models \Phi'$. This justifies the base case.

Suppose that every universal formula of length less than k makes the theorem true and that Φ has length k.

If $\Phi =$ `(necessarily` p`)`, then for every fullpath y starting at the first state in x, $M, y \models p$. Therefore, by induction, for every fullpath y starting at the first state in x, $M, y \models p'$. In particular, $M, x \models p'$ and hence $M, x \models \Phi'$.

If $\Phi =$ `(and` $p_1 \cdots p_n$`)`, then $M, x \models p_i$ for each $i, 1 \leq i \leq n$. We need to show that, $M, x \models p_i'$ for each i. This follows by induction. Thus, $M, x \models \Phi'$.

If $\Phi =$ `(or` $p_1 \cdots p_n$`)`, then $M, s \models p_i$ for some $i, 1 \leq i \leq n$. We need to show that, $M, x \models p_i'$ for some i. This follows by induction. Therefore, $M, x \models \Phi'$.

If $\Phi =$ `(not` p`)`, then Φ is an atomic proposition since Φ is in positive normal form and so $p = p'$.

If $\Phi =$ `(until` p q`)`, then there is a nonnegative integer $i < \Lambda(x)$ (we choose the smallest) such that $M, x^i \models q$ and for every nonnegative integer $j < i$, $M, x^j \models p$. Therefore, $M, x^i \models q'$ and for every nonnegative integer $j < i$, $M, x^j \models p'$ by induction. Therefore, $M, x \models$ `(until` p' q'`)`.

If $\Phi =$ `(releases` p q`)`, then for every nonnegative integer $i < \Lambda(x)$ such that $M, x^i \not\models q$ there is a nonnegative integer $j < i$ such that $M, x^j \models p$.

First suppose that for every nonnegative integer $i < \Lambda(x)$, $M, x^i \models q$. Then $M, x^i \models q'$ for every nonnegative integer $i < \Lambda(x)$ by induction. Therefore, $M, x \models \Phi'$.

Now suppose that there is a nonnegative integer $i < \Lambda(x)$ such that $M, x \not\models q$. We select i to be the smallest such nonnegative integer. Thus, there is a nonnegative integer $j < i$ such that $M, x^j \models p$. For all such j we also have $M, x^j \models p'$ by induction. For every nonnegative integer $k < i$, we have $M, x^k \models q$ by the choice of i and so $M, x^k \models q'$ by induction. Therefore, we get that for every nonnegative integer $l < \Lambda(x)$ such that $M, x^l \not\models q'$ there is a nonnegative integer $j < l$ such that $M, x^j \models p'$. That is to say, $M, x \models$ `(releases` p' q'`)`.

If $\Phi =$ `(next` p`)`, then if $\Lambda(x) > 0$ then $M, x^1 \models p$ and we are done by induction. If $\Lambda(x) = 0$ then $M, x \models$ `(next` p'`)` as well.

If $\Phi =$ `(strong-next` p`)`, then $\Lambda(x) > 0$ and $M, x^1 \models p$ so we are done by induction.

Proof of Theorem 2. Suppose x is a fullpath in a QSIM structure $\widehat{M}$ which is determined with respect to all of the propositions in Φ, a perfect path formula, and $\widehat{M}_{TL}, x \models \Phi$. If the specification $\langle x, c \rangle$ of x qualitatively describes $\langle \{t_i\}, U \rangle$ then we show that $\langle \Phi, c \rangle$ temporally describes $\langle \{t_i\}, U \rangle$ via $\psi_{_{U,M}}$.

Without loss of generality, we assume that Φ is in positive normal form. Suppose that x is a fullpath in a QSIM structure $\widehat{M}$ which is determined with respect to all of the propositions in Φ, a perfect path formula in positive normal form, and $\widehat{M}_{TL}, x \models \Phi$. We will apply induction on the length, k, of Φ. In the base case, Φ is an atomic proposition. This case follows from Lemma 4.

Our induction hypothesis says that for any fullpath y in a QSIM structure such that $\widehat{M}_{TL}, y \models \Psi$ where Ψ is a perfect path formula in positive normal form of length less than k and M is determined with respect to every proposition in Ψ, then if $\langle y, c \rangle$ qualitatively describes a splitting $\langle \{r_i\}, V \rangle$ then $\langle \Psi, c \rangle$ temporally describes $\langle \{r_i\}, V \rangle$ via $\psi_{_{U,M}}$.

If $\Phi =$ `(and` $p_1 \cdots p_m$`)`, then we must show that if $\langle p_j, c \rangle$ temporally describes $\langle \{t_i\}, U \rangle$ via $\psi_{_{U,M}}$ for each $j, 1 \leq j \leq m$ then $\widehat{M}_{TL}, x \models p_j$ for each

$j, 1 \leq j \leq m$. This follows by the induction hypothesis.

If $\Phi = (\texttt{or}\ p_1 \cdots p_m)$, then we must show that if $\langle p_j, c\rangle$ temporally describes $\langle \{t_i\}, U\rangle$ via $\psi_{U,M}$ for some $j, 1 \leq j \leq m$ then $\widehat{M}_{TL}, x \models p_j$ for some $j, 1 \leq j \leq m$. Again, this follows by induction.

If $\Phi = (\texttt{not}\ p)$, then we must show that if $\langle p, c\rangle$ does not temporally describe $\langle \{t_i\}, U\rangle$ via $\psi_{U,M}$ then $\widehat{M}_{TL}, x \not\models p$. Since, by the definition of positive normal form, p must be an atomic proposition, this follows from the induction hypothesis and Lemma 4.

If $\Phi = (\texttt{until}\ p\ q)$, then we must show that if $\langle x, c\rangle$ qualitatively describes $\langle \{t_i\}, U\rangle$, then $\langle \Phi, c\rangle$ temporally describes $\langle \{t_i\}, U\rangle$ via $\psi_{U,M}$. By the semantics of `until` we know that there is a nonnegative integer $h < \Lambda(x)$ such that $\widehat{M}_{TL}, x^h \models q$ and for every nonnegative integer $l < h, \widehat{M}_{TL}, x^l \models p$. Since $\langle x, c\rangle$ qualitatively describes $\langle \{t_i\}, U\rangle$, $\langle x^h, c\rangle$ qualitatively describes $\langle \{t_i : d_h \leq i\}, U|_{h+}\rangle$ by Lemma 3. Thus $\langle q, c\rangle$ temporally describes $\langle \{t_i : d_h \leq i\}, U|_{h+}\rangle$ via $\psi_{U,M}$ by induction. For every nonnegative integer $l < h$, $\langle x^l, c\rangle$ qualitatively describes $\langle \{t_i : d_l \leq i\}, U|_{l+}\rangle$ by Lemma 3. Thus $\langle p, c\rangle$ temporally describes $\langle \{t_i : d_l \leq i\}, U|_{l+}\rangle$ via $\psi_{U,M}$ by induction. So $\langle \Phi, c\rangle$ temporally describes $\langle \{t_i\}, U\rangle$ via $\psi_{U,M}$.

If $\Phi = (\texttt{releases}\ p\ q)$, then we must show that if $\langle x, c\rangle$ qualitatively describes $\langle \{t_i\}, U\rangle$ then $\langle \Phi, c\rangle$ temporally describes $\langle \{t_i\}, U\rangle$ via $\psi_{U,M}$. Since $\widehat{M}_{TL}, x \models \Phi$ we know that for every nonnegative integer $h < \Lambda(x)$ such that $\widehat{M}_{TL}, x^h \not\models q$ there is a nonnegative integer $l < h$ such that $\widehat{M}_{TL}, x^l \models p$. First suppose that $\widehat{M}_{TL}, x^h \models q$ for every nonnegative integer $h < \Lambda(x)$. We know that $\langle x^h, c\rangle$ qualitatively describes $\langle \{t_i : d_h \leq i\}, U|_{h+}\rangle$ for every nonnegative integer $h < \Lambda(x)$ by Lemma 3. Therefore, $\langle q, c\rangle$ temporally describes $\langle \{t_i : d_h \leq i\}, U|_{h+}\rangle$ via $\psi_{U,M}$ for every nonnegative integer $h < \Lambda(x)$ by induction and so $\langle x, c\rangle$ qualitatively describes $\langle \{t_i\}, U\rangle$.

Now suppose that there is a nonnegative integer $h < \Lambda(x)$ such that $\widehat{M}_{TL}, x^h \not\models q$ and it is the smallest such h. There is a nonnegative integer $l < h$ such that $\widehat{M}_{TL}, x^l \models p$. For every nonnegative integer $j < h, \widehat{M}_{TL}, x^j \models q$ and since $\langle x^j, c\rangle$ qualitatively describes $\langle \{t_i : d_j \leq i\}, U|_{j+}\rangle$, $\langle q, c\rangle$ temporally describes $\langle \{t_i : d_j \leq i\}, U|_{j+}\rangle$ via $\psi_{U,M}$ by induction. For one of these $j, \widehat{M}_{TL}, x^j \models p$ and hence $\langle p, c\rangle$ temporally describes $\langle \{t_i : d_j \leq i\}, U|_{j+}\rangle$ via $\psi_{U,M}$ by induction. So we have that if there is a nonnegative integer $h < \Lambda(x)$ such that $\widehat{M}_{TL}, x^h \not\models q$, there is a nonnegative integer $j < h$ such that $\langle p, c\rangle$ temporally describes $\langle \{t_i : d_j \leq i\}, U|_{j+}\rangle$ via $\psi_{U,M}$ and for every nonnegative integer $l \leq j, \langle q, c\rangle$ temporally describes $\langle \{t_i : d_l \leq i\}, U|_{l+}\rangle$ via $\psi_{U,M}$. Therefore, $\langle \Phi, c\rangle$ temporally describes $\langle \{t_i\}, U\rangle$ via $\psi_{U,M}$.

If $\Phi = (\texttt{next}\ p)$, then we must show that if $0 = 2J$ or $\langle p, c\rangle$ temporally describes $\langle \{t_i : d_1 \leq i\}, U|_{1+}\rangle$ via $\psi_{U,M}$ then $\Lambda(x) = 1$ or $\langle x^1, c\rangle$ qualitatively describes $\langle \{t_i : d_1 \leq i\}, U|_{1+}\rangle$. This follows by induction and the use of Lemma 3.

If $\Phi = (\texttt{strong-next}\ p)$, then we must show that if $0 < 2J$ and $\langle p, c\rangle$

temporally describes $\langle \{t_i : d_1 \leq i\}, U|_{1+} \rangle$ via $\psi_{U,M}$ then $\Lambda(x) > 1$ and $\langle x^1, c \rangle$ qualitatively describes $\langle \{t_i : d_1 \leq i\}, U|_{1+} \rangle$. Once again, this follows from the induction hypothesis and Lemma 3.

Acknowledgments

QSIM and TL are available for anonymous `ftp` at `ftp.cs.utexas.edu` in the directory `/pub/qsim`. These and other results of the Qualitative Reasoning Group are accessible by World-Wide Web via `http://www.cs.utexas.edu/users/qr`.

This report was prepared by the University of Texas at Austin as an account of work sponsored in part by the National Science Foundation (grant IRI-9216584) and by the Electric Power Research Institute, Inc. (EPRI) (grant RP8030-21). Neither EPRI, members of EPRI, the University of Texas at Austin, nor any person acting on their behalf: (a) makes any warranty, express or implied, with respect to the use of any information, apparatus, method, or process disclosed in this report or that such use may not infringe privately owned rights; or (b) assumes any liabilities with respect to the use of, or for damages resulting from the use of, any information, apparatus, method, or process disclosed in this report.

We would like to thank Bhat, Grumberg and Cleaveland for their excellent article [3] and Rance Cleaveland for answering our questions about their algorithm. A difficult task in writing a paper which is intended for audiences in two somewhat disjoint disciplines, such as Qualitative Reasoning and Temporal Logic, is to use language which is comfortable to readers in both fields. We are grateful for helpful comments and suggestions from Markus Kaltenbach, Richard Trefler, Dan Clancy, Bert Kay, Giorgio Brajnik, Michael Hofbaur, and two anonymous reviewers.

References

[1] R. Alur and T. Henzinger. Real-time logics: conplexity and expressiveness. *Information and Computation*, 104(1):35–77, 1993.

[2] Daniel Berleant and Benjamin Kuipers. Combined qualitative and numerical simulation with Q3. In Boi Faltings and Peter Struss, editors, *Recent Advances in Qualitative Physics*. MIT Press, Cambridge, MA, 1992.

[3] Girish Bhat, Rance Cleaveland, and Orna Grumberg. Efficient on-the-fly model checking for CTL*. In *Proc. Conf. on Logic in Computer Science (LICS-95)*, 1995.

[4] Giorgio Brajnik and Daniel J. Clancy. Guiding and refining simulation using temporal logic. In *Third International Workshop on Temporal Representation and Reasoning (TIME'96)*, Florida, 1996.

[5] Giorgio Brajnik and Daniel J. Clancy. Temporal constraints on trajectories in qualitative simulation. In *Proc. 10th Int. Workshop on Qualitative Reasoning About Physical Systems*, Fallen Leaf Lake, California, USA, 1996.

[6] Giorgio Brajnik and Daniel J. Clancy. Temporal constraints on trajectories in qualitative simulation. In *Proc. 13th National Conf. on Artificial Intelligence (AAAI-96)*, Cambridge, MA, 1996. AAAI Press/The MIT Press.

[7] E. M. Clarke, E. A. Emerson, and A. P. Sistla. Automatic verification of finite-state concurrent systems using temporal logic specifications. *ACM Transactions on Programming Languages and Systems*, 8(2):244–263, August 1986.

[8] E. Allen Emerson. Temporal and modal logic. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, pages 995–1072. Elsevier Science Pub. B. V./MIT Press, 1990.

[9] E. Gazi, L. H. Ungar, and B. J. Kuipers. Temporal logic for summarizing monte-carlo simulation: an application to controller verification. In R. Shoureshi, editor, *Intelligent Control*. IEEE, 1996. (submitted).

[10] E. Gazi, L. H. Ungar, W. D. Seider, and B.J. Kuipers. Automatic analysis of monte-carlo simulations of dynamic chemical plants. In *Proceedings of the European Symposium on Computer Aided Process Engineering (ESCAPE 6)*. Pergamon Press, 1996. (in press).

[11] Farnam Jahanian and Douglas A. Stewart. A method for verifying properties of Modechart specifications. In *Proceedings of the Real-time Systems Symposium*, Huntsville, AL, December 1988.

[12] Herbert Kay and Benjamin Kuipers. Numerical behavior envelopes for qualitative models. In *Proc. 11th National Conf. on Artificial Intelligence*, pages 606–613, Cambridge, MA, 1993. AAAI/MIT Press.

[13] B. Kuipers. Qualitative simulation. *Artificial Intelligence*, 29:289–338, 1986.

[14] B. Kuipers and D. Berleant. Using incomplete quantitative knowledge in qualitative reasoning. *Proc. 7th National Conf. on Artificial Intelligence (AAAI-88)*, 1988.

[15] B. J. Kuipers and K. Åström. The composition and validation of heterogeneous control laws. *Automatica*, 30(2):233–249, 1994.

[16] Benjamin Kuipers. *Qualitative Reasoning: Modeling and Simulation with Incomplete Knowledge*. MIT Press, Cambridge, MA, 1994.

[17] Benjamin Kuipers and Benjamin Shults. Reasoning in logic about continuous systems. In J. Doyle, E. Sandewall, and P. Torasso, editors, *Principles of Knowledge Representation and Reasoning: Proceedings of the Fourth International Conference (KR-94)*, San Mateo, CA, 1994. Morgan Kaufmann.

[18] O. Lichtenstein and A. Pnueli. Checking that finite state concurrent programs satisfy thir linear specifications. In *Twelfth Annual ACM Symposium on Principles of Programming Languages*, pages 97–107, 1984.

[19] I. Moon, G. J. Powers, J. R. Burch, and E. M. Clarke. Automatic verification of sequential control systems using temporal logic. *AIChE Journal*, 38(1):67–75, 1992.

[20] Manny Rayner. On the applicability of nonmonotonic logic to formal reasoning in continuous time. *Artificial Intelligence*, 49:345–360, 1991.