



Duration consistency filtering for qualitative simulation

Tolga Könik^a and A.C. Cem Say^b

^a *Artificial Intelligence Lab., University of Michigan, Advanced Tech. Bldg., 1101 Beal Ave., Ann Arbor, MI 48109-2110, USA*

E-mail: konik@umich.edu

^b *Department of Computer Engineering, Boğaziçi University, Bebek 34342, İstanbul, Turkey*

E-mail: say@boun.edu.tr

We present two new qualitative reasoning formalisms, and use them in the construction of a new type of filtering mechanism for qualitative simulators. Our new sign algebra, SR1*, facilitates reasoning about relationships among the signs of collections of real numbers. The *comparison calculus*, built on top of SR1*, is a general framework that can be used to qualitatively compare the behaviors of two dynamic systems or two excerpts of the behavior of a single dynamic system at different situations. These tools enable us to improve the predictive performance of qualitative simulation algorithms. We show that qualitative simulators can make better use of their input to deduce significant amounts of qualitative information about the relative lengths of the time intervals in their output behavior predictions. Simple techniques employing concepts like symmetry, periodicity, and comparison of the circumstances during multiple traversals of the same region can be used to build a list of facts representing the deduced information about relative durations. The *duration consistency filter* eliminates spurious behaviors leading to inconsistent combinations of these facts. Surviving behaviors are annotated with richer qualitative descriptions. Used in conjunction with other spurious behavior elimination methods, this approach would increase the ability of qualitative simulators to handle more complex systems.

Keywords: qualitative reasoning, qualitative simulation, spurious behavior filtering, sign algebra, qualitative comparison

AMS subject classification: 34Cxx

1. Introduction

The AI methodology of qualitative reasoning [17] is based on the principle of using “low-resolution” representations in models of systems whose component quantities and relationships are only incompletely known. This ability to concisely represent and use incomplete knowledge enables qualitative reasoners to solve entire families of equations “in one stroke”, and renders them useful for proving certain behavioral properties collectively for large classes of systems with models sharing the same basic structure.

In this paper, we present two new qualitative reasoning formalisms, and use them in the construction of a new type of filtering mechanism for qualitative simulators. The manuscript is structured in two parts. The first part, comprising sections 2 and 3, describes the mathematical tools we developed for comparing functions represented in

the qualitative format. We start by defining a new algebra named $SR1^*$, whose domain includes not only signs and real numbers, but arbitrary nonempty sets of reals. $SR1^*$ enables us to reason about the relationships among the signs of collections of real numbers. After that, we present a new comparison formalism, the *comparison calculus*, which is built on top of $SR1^*$. The comparison calculus is a general framework that can be used to qualitatively compare the behaviors of two dynamic systems or two excerpts of the behavior of a single dynamic system at different times.

In the second part of the paper, starting with section 4, we demonstrate a novel technique for improving the predictive performance of qualitative simulators. Qualitative simulation algorithms symbolically solve families of ordinary differential equations, predicting a set of trajectory descriptions, such that any actual solution of the equations in the input set is guaranteed to match one of these predictions in the output. Along with qualitative descriptions of all possible behaviors that can be exhibited by systems in the input model, qualitative simulators may produce *spurious predictions*, behavior descriptions which no such system would exhibit. These spurious predictions limit the usefulness of qualitative reasoners in applications like design and diagnosis [10]. There is on-going research [10–13] that aims to improve qualitative simulation for reducing the number of spurious behaviors in the algorithms' output.

We show that qualitative simulation algorithms can make better use of their input to deduce significant amounts of information about the relative lengths of the time intervals in their output behavior predictions. Simple techniques employing concepts like symmetry, periodicity, and comparison of the circumstances during multiple traversals of the same interval can enable the reasoner to build a list of facts representing the deduced information about duration comparisons. These facts are used by the new *duration consistency filter*, which eliminates proposed spurious behaviors leading to inconsistent duration data. Surviving behaviors are annotated with richer descriptions of the qualitative properties of system variables, in addition to the extracted duration comparison information. We describe the incorporation of this approach to the “standard” qualitative simulation algorithm QSIM [10]. The correctness guarantees of some of the duration comparison fact extraction methods that we employ are based on properties of the $SR1^*$ algebra and the comparison calculus proven in the first part of the paper. Examples of the utility of the improved algorithm and an evaluation are presented.

PART I. MATHEMATICAL FOUNDATIONS OF FUNCTION COMPARISON

2. Sign algebras

Qualitative reasoning algorithms make heavy use of the sign representation for quantities. In this section, a brief review of two sign algebras from the qualitative reasoning literature will be followed by a presentation of our extension, $SR1^*$. This forms the basis of the comparison calculus to be introduced in the next section.

Table 1
Operator tables of multiplication and addition in S1.

·	[+]	[-]	[0]	[?]	+	[+]	[-]	[0]	[?]
[+]	[+]	[-]	[0]	[?]	[+]	[+]	[?]	[+]	[?]
[-]	[-]	[+]	[0]	[?]	[-]	[?]	[-]	[-]	[?]
[0]	[0]	[0]	[0]	[0]	[0]	[+]	[-]	[0]	[?]
[?]	[?]	[?]	[0]	[?]	[?]	[?]	[?]	[?]	[?]

Definition 2.1 (Domain of signs). A *sign* is one of the following four subsets of \mathcal{R} , the set of real numbers:

$$[+] =_{\text{def}} (0, +\infty), \quad [-] =_{\text{def}} (-\infty, 0), \quad [?] =_{\text{def}} (-\infty, +\infty), \quad [0] =_{\text{def}} \{0\}.$$

The set $S' = \{[-], [0], [+], [?]\}$ is called the *extended domain of signs*.

Williams [19] defined a basic sign algebra called S1 on the set S' with the sign addition and multiplication operators (table 1). For two signs s_1 and s_2 , “ $s_1 - s_2$ ” is defined to be equivalent to “ $s_1 + ([-] \cdot s_2)$ ”. S1 formalizes commonsense statements such as “*The product of two negative numbers is a positive number*” ($[-] \cdot [-] = [+]$), or “*The sign of the sum of a positive number and a negative number can be anything*” ($[+] + [-] = [?]$).

Here are some simple properties of S1 that we are going to use in the rest of the paper:

Proposition 2.1 (Some simple properties of S1). For the signs $s_1, s_2, s_3 \in S'$

- (i) $s_1 \subseteq s_2 \leftrightarrow s_1 = s_2$ if $s_2 \neq [?]$,
- (ii) $s_1 = s_2 \cdot s_3 \leftrightarrow s_1 \cdot s_2 = s_3$ if $s_2 \neq [?], [0]$,
- (iii) $s_1 \neq [?] \wedge s_2 \neq [?] \leftrightarrow (s_2 + s_1 \neq [?]) \vee (s_2 - s_1 \neq [?])$.

Proof. These statements can be proven by testing them for all sign combinations. \square

Williams’ SR1 [19] is an extension of S1 where real numbers are elements of the domain in addition to signs of S' . For example $[+] + 3 = [+]$ holds in SR1. Building on SR1, we developed a new sign/real hybrid algebra called SR1* that enables us to reason about the relationships among the signs of collections of real numbers. The domain of SR1* is the set of all nonempty subsets of \mathcal{R} . The following set operators are well-defined on SR1*.

Definition 2.2 (Set operators used in SR1*). For A and B , two subsets of \mathcal{R} , the following set operators are used in SR1*:

- (i) $A + B =_{\text{def}} \{a + b: a \in A, b \in B\}$,

- (ii) $-B =_{\text{def}} \{-b: b \in B\}$,
- (iii) $A - B =_{\text{def}} A + (-B)$,
- (iv) $A \cdot B =_{\text{def}} \{a \cdot b: a \in A, b \in B\}$,
- (v) $|A| =_{\text{def}} \{|a|: a \in A\}$ ($|a|$: absolute value of a).

Just like SR1, SR1* contains all signs of S' and all real numbers¹ as elements. Moreover, S' is a subalgebra of SR1* as it is a subalgebra of SR1; that is, the operators in SR1* work exactly as their counterparts in S1 for the signs in S' . Unlike SR1, SR1* contains arbitrary subsets of \mathcal{R} as elements and although the domain of SR1 is a subset of the domain of SR1*, SR1 is not a subalgebra of SR1*. For example, $[+] + (-3)$ is mapped to $(-3, \infty)$ in SR1*, while it is mapped to $[?]$ in SR1, since $(-3, \infty)$ is not an element of SR1. Nevertheless, SR1 operators can easily be defined in SR1* [8].

Proposition 2.2 (A set theoretical property of SR1*). Given $A_1, A_2, \dots, A_n, B_1, B_2, \dots, B_n \in \text{SR1}^*$ such that $A_1 \subseteq B_1, A_2 \subseteq B_2, \dots, A_n \subseteq B_n$, and Φ , a formula written using the binary and unary operators in definition 2.2, the following relationship holds:

$$\Phi(A_1, A_2, \dots, A_n) \subseteq \Phi(B_1, B_2, \dots, B_n).$$

The sign abstraction operator $[.]$ in SR1* is similar to its counterpart in SR1. $[A]$ is the smallest sign in S' that covers A . For example, the expressions $[5] = [+]$, $[(0, +\infty)] = [+]$, and $[\{-1, +1\}] = [?]$ are valid in SR1*.

Definition 2.3 (Sign abstraction operator of SR1*). Given $A \in \text{SR1}^*$, we define $[A] \in S'$ such that:

- (1) $A \subseteq [A]$,
- (2) $\forall s \in S'$ if $A \subseteq s$ then $[A] \subseteq s$.

Definition 2.4 (Sign profile of an SR1* element). For $A \in \text{SR1}^*$, we call $\text{SP}(A)$ the *sign profile of A* , the set of all signs the elements of A have.

For example, if $A = \{-5, +7\}$ then we have $\text{SP}(A) = \{[-], [+]\}$, which we will abbreviate as $\text{SP}(A) = \{-, +\}$. Table 2 depicts all possible sign profiles that may correspond to each sign. We will use the concept of sign profiles in proving the following. Although Williams [19] has given similar statements for the abstraction operator of SR1, in SR1* they deserve a separate proof.

¹ Real numbers are actually not in the domain of SR1*, but all singleton sets of reals are in the domain and the real numbers are isomorphic to them. For example $3 + 2 = 5$ can be considered to be an SR1* expression, because $\{3\} + \{2\} = \{5\}$ is one. Williams' SR1 uses the same trick.

Table 2
Signs versus possible corresponding sign profiles.

$[A]$	$SP(A)$
$[+]$	$\{+\}$
$[-]$	$\{-\}$
$[0]$	$\{0\}$
$[?]$	$\{+, -\}, \{+, 0\}, \{-, 0\}, \{+, -, 0\}$

Proposition 2.3 (Abstraction properties in SR1*). For any $A, B \in SR1^*$,

- (i) $[A + B] \subseteq [A] + [B]$,
- (ii) $[A \cdot B] = [A] \cdot [B]$,
- (iii) $[-A] = -[A]$,
- (iv) $[A - B] \subseteq [A] - [B]$.

Proof. (i)

$$\begin{aligned} A + B &\subseteq [A] + [B] && \text{since } A \subseteq [A], B \subseteq [B] \text{ (proposition 2.2),} \\ [A + B] &\subseteq [[A] + [B]] && \text{since } X \subseteq Y \rightarrow [X] \subseteq [Y], \\ [A + B] &\subseteq [A] + [B] && \text{since } [A] + [B] \in S' \text{ and therefore } [[A] + [B]] = [A] + [B]. \end{aligned}$$

(ii) The equality is proven using table 3. Here, for each value of $[A]$ and $[B]$, we generate all possible values for $SP(A)$, $SP(B)$ and $SP(A \cdot B)$ (in boxes). We observe that in each case, a unique $[A \cdot B]$ corresponds to all possible $SP(A \cdot B)$ values and that $[A] \cdot [B] = [A \cdot B]$.

(iii) Using (ii), by inserting $B = \{-1\}$.

(iv) Similar to (i). □

Theorem 2.1 (General abstraction in SR1*). Given $A_1, A_2, \dots, A_n \in SR1^*$, such that $A_1 \subseteq \Phi(A_2, \dots, A_n)$ where Φ is a formula written using the operators $\{\cdot, -, +\}$, the following relation holds:

$$[A_1] \subseteq \Phi([A_2], \dots, [A_n]).$$

Proof.

$$\begin{aligned} [A_1] &\subseteq [\Phi(A_2, \dots, A_n)] && \text{since } X \subseteq Y \rightarrow [X] \subseteq [Y] \\ &\subseteq \Phi([A_2], \dots, [A_n]) && \text{using proposition 2.3 recursively.} \end{aligned}$$

We now start abstracting functions. In this paper, we assume that the domains of all functions are subsets of \mathcal{R} . □

Definition 2.5 (Abstraction of a function). Given a function f and F , the image of f on a domain \mathbf{I} , we call the sign valued expression $[F]$ the *image abstraction of f on the domain \mathbf{I}* and for a $t \in \mathbf{I}$ we call $[f(t)]$ the *point abstraction of f at t* .

Table 3
Sign multiplication with sign profiles.

		[B]			
		[+]	[-]	[0]	[?]
·		[+] +	[-] -	[0] 0	[?] +,0 -,0 +,- +,-,0
		[+] +	[-] -	[0] 0	[?] +,0 -,0 +,- +,-,0
[A]	[+] +	[+] +	[-] -	[0] 0	[?] +,0 -,0 +,- +,-,0
	[-] -	[-] -	[+] +	[0] 0	[?] -,0 +,0 +,- +,-,0
	[0] 0	[0] 0	[0] 0	[0] 0	[0] 0 0 0 0
	[?] +,0 -,0 +,- +,-,0	[?] +,0 -,0 +,- +,-,0	[?] -,0 +,0 +,- +,-,0	[0] 0 0 0	[?] +,0 -,0 +,-,0 +,-,0 -,0 +,0 +,-,0 +,-,0 +,-,0 +,-,0 +,- +,-,0 +,-,0 +,-,0 +,-,0 +,-,0

Theorem 2.2 (Abstraction of functional relations). Given n functions $f_1, f_2, f_3, \dots, f_n$ and their images $F_1, F_2, F_3, \dots, F_n$ on a domain \mathbf{I} , if it is the case that

$$f_1(t) = \Phi(f_2(t), f_3(t), \dots, f_n(t)) \quad \forall t \in \mathbf{I}$$

such that Φ is a formula written using the operators $\{\cdot, -, +\}$, the following relations hold:

- (i) $F_1 \subseteq \Phi(F_2, F_3, \dots, F_n)$,
- (ii) $[F_1] \subseteq \Phi([F_2], [F_3], \dots, [F_n])$.

Proof. (i) Result of set theory.
(ii) By (i) and theorem 2.1. □

Proposition 2.4 (Abstraction of sign multiplication of functions). Given three functions f, g, h and their images F, G, H on \mathbf{I} such that $\forall t \in \mathbf{I} [f(t)] = [g(t)] \cdot [h(t)]$, the following relation holds:

$$[F] = [G] \cdot [H] \quad \text{if } [G] \neq [?] \vee [H] \neq [?].$$

Proof. Assuming $[G] \neq [?]$, we consider all possible values of $[G]$:

$$[G] = [+] \rightarrow \forall t \in \mathbf{I} [g(t)] = [+] \rightarrow \forall t \in \mathbf{I} [f(t)] = [h(t)] \rightarrow [F] = [H],$$

$$[G] = [-] \rightarrow \forall t \in \mathbf{I} [g(t)] = [-] \rightarrow \forall t \in \mathbf{I} [f(t)] = -[h(t)] \rightarrow [F] = -[H],$$

$$[G] = [0] \rightarrow \forall t \in \mathbf{I} [g(t)] = [0] \rightarrow \forall t \in \mathbf{I} [f(t)] = [0] \rightarrow [F] = [0].$$

In all cases, $[F] = [G] \cdot [H]$ is satisfied. (The proof is similar for $[H] \neq [?]$.) \square

3. The comparison calculus

This section introduces the comparison calculus, a general framework for comparing two incompletely known functions, or two portions of the same function over different parts of its domain.

3.1. Comparison of real variables

Definition 3.1 (Real comparison variables). For two real numbers r_1 and r_2 , we define the *real comparison variables* Δr , Σr , and $\Delta|r|$ such that:

(i) $\Delta r \stackrel{\text{def}}{=} r_2 - r_1$,

(ii) $\Sigma r \stackrel{\text{def}}{=} r_2 + r_1$,

(iii) $\Delta|r| \stackrel{\text{def}}{=} |r_2| - |r_1|$.

Definition 3.2 (Real comparison signs). Given two real numbers r_1 and r_2 , the well-defined sign expressions $[\Delta r]$, $[\Sigma r]$, and $[\Delta|r|]$ are called *the real comparison signs*. Specifically, $[\Delta|r|]$ is called the *real magnitude comparison sign*. Moreover, we call $[r_1]$ and $[r_2]$ the *simple sign constants* and using them we define the *compound sign constants* as: $\Sigma[r] \stackrel{\text{def}}{=} [r_2] + [r_1]$ and $\Delta[r] \stackrel{\text{def}}{=} [r_2] - [r_1]$.

Theorem 3.1 (Real comparison conversion constraints). For two real numbers r_1 and r_2 ,

(i)(a) $[\Delta|r|] = [\Delta r \cdot \Sigma r] = [\Delta r] \cdot [\Sigma r]$,

(i)(b) $[\Sigma r] = [\Delta|r|] \cdot [\Delta r]$ if $[\Delta r] \neq [0]$,

(i)(c) $[\Delta r] = [\Delta|r|] \cdot [\Sigma r]$ if $[\Sigma r] \neq [0]$,

(ii) $[\Sigma r] = \Sigma[r]$ if $\Sigma[r] \neq [?]$,

(iii) $[\Delta r] = \Delta[r]$ if $\Delta[r] \neq [?]$.

Proof. (i)(a)

$$\begin{aligned} [\Delta|r|] &= [|r_2| - |r_1|] = [|r_2|^2 - |r_1|^2] = [r_2^2 - r_1^2] = [(r_2 - r_1) \cdot (r_2 + r_1)] \\ &= [\Delta r \cdot \Sigma r] = [\Delta r] \cdot [\Sigma r] \quad \text{using proposition 2.3(ii)}. \end{aligned}$$

(i)(b) Using (i)(a) and proposition 2.1(ii) since Δr is real and therefore $[\Delta r] \neq [?]$.

(i)(c) Similar to (i)(b).

(ii) $[\Sigma r] = [r_2 + r_1] \subseteq [r_2] + [r_1] = \Sigma[r]$ using proposition 2.3(i). $[\Sigma r] = \Sigma[r]$ since $[\Sigma r] \subseteq \Sigma[r]$, and $\Sigma[r] \neq [?]$ (proposition 2.1(i)).

(iii) Similar to (ii). □

Definition 3.3 (Some real features of functions). For two functions f_1 and f_2 on intervals $\mathbf{I}_1 = (t_{b1}, t_{e1})$ and $\mathbf{I}_2 = (t_{b2}, t_{e2})$, we define the following real valued quantities (for $j = 1, 2$):

- *initial value* of f_j on \mathbf{I}_j : $f_{bj} =_{\text{def}} \lim_{t \rightarrow t_{bj}} f_j(t)$,
- *final value* of f_j on \mathbf{I}_j : $f_{ej} =_{\text{def}} \lim_{t \rightarrow t_{ej}} f_j(t)$,
- *change* of f_j on \mathbf{I}_j : $f_{\Delta j} =_{\text{def}} f_{ej} - f_{bj}$,
- *duration* of \mathbf{I}_j : $t_{\Delta j} =_{\text{def}} t_{ej} - t_{bj}$,
- *average value* of f_j on \mathbf{I}_j :

$$\bar{f}_j =_{\text{def}} \frac{1}{t_{\Delta j}} \cdot \int_{t_{bj}}^{t_{ej}} f_j(t) dt.$$

f_{bj} and f_{ej} will also be called the *end points* of f_j on \mathbf{I}_j .

The quantity pairs defined above can be inserted into definitions 3.1 and 3.2 to obtain expressions that are useful for making comparisons. For example for two time intervals $\mathbf{I}_1, \mathbf{I}_2$ and their durations $t_{\Delta 1}, t_{\Delta 2}$, we get well-defined sign valued expressions like $[\Delta|t_{\Delta}] = [|t_{\Delta 2}| - |t_{\Delta 1}|]$. Such expressions enable us to represent comparison facts using algebraic formulas. For example, $[\Delta|t_{\Delta}] = [-]$ means that the duration of the second interval is smaller than the first one ($|t_{\Delta 2}| < |t_{\Delta 1}|$).

Let us compare the position (x_1, x_2) and velocity (v_1, v_2) graphs of two cars given in figure 1 on the time intervals $\mathbf{I}_1 = (t_{b1}, t_{e1})$ and $\mathbf{I}_2 = (t_{b2}, t_{e2})$. We insert the well-defined real quantities $x_{\Delta 1}, x_{\Delta 2}, \bar{v}_1$, and \bar{v}_2 obtained from these functions (definition 3.3) into definition 3.1 and definition 3.2 to construct the comparison signs $[\Delta|x_{\Delta}] = [|x_{\Delta 2}| - |x_{\Delta 1}|]$ and $[\Delta|\bar{v}] = [|\bar{v}_2| - |\bar{v}_1|]$. In this example, we observe that $[\Delta|x_{\Delta}] = [+]$, that is, the distance traveled by the second car in \mathbf{I}_2 is greater than the distance traveled by the first car in \mathbf{I}_1 ($|x_{\Delta 2}| > |x_{\Delta 1}|$). Similarly, $[\Delta|\bar{v}] = [-]$ means that the average speed of the second car is lower than the average speed of the first car ($|\bar{v}_2| < |\bar{v}_1|$).

Although the values of time points t_{b1}, t_{e1}, t_{b2} , and t_{e2} are not known, applying the common sense rule

“It takes longer to traverse a longer path with a lower speed”

to these observations, we may conclude that the duration of the second interval is longer ($|t_{\Delta 2}| > |t_{\Delta 1}|$), that is, $[\Delta|t_{\Delta}] = [+]$. The next theorem establishes a qualitative relation between the “change-of-position” comparison $[\Delta|x_{\Delta}]$, the average speed comparison $[\Delta|\bar{v}]$, and the duration comparison $[\Delta|t_{\Delta}]$.

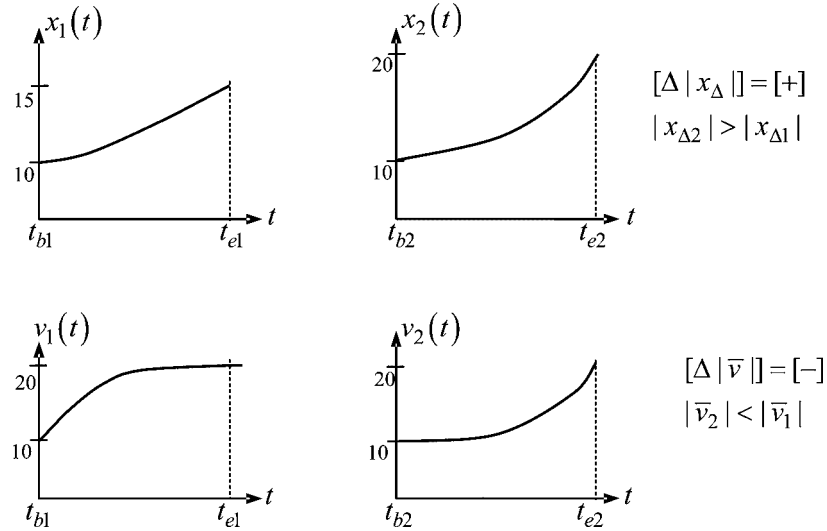


Figure 1. Position and velocity of two cars.

Theorem 3.2 (Qualitative average value constraint). Given a pair of functions x_1, x_2 and their derivatives v_1, v_2 on the intervals $I_1 = (t_{b1}, t_{e1})$ and $I_2 = (t_{b2}, t_{e2})$, the following holds:

$$[\Delta|x_{\Delta}|] \subseteq [\Delta|\bar{v}|] + [\Delta|t_{\Delta}|] \quad \text{if } [\bar{v}_1] \neq [0] \text{ or } [\bar{v}_2] \neq [0].$$

Proof. We assume $[\bar{v}_1] \neq [0]$ (the proof is similar for the case $[\bar{v}_2] \neq [0]$):

$$\begin{aligned} \Delta|x_{\Delta}| &= |x_{\Delta 2}| - |x_{\Delta 1}| = |\bar{v}_2| \cdot |t_{\Delta 2}| - |\bar{v}_1| \cdot |t_{\Delta 1}| \\ &= |\bar{v}_2| \cdot |t_{\Delta 2}| - |\bar{v}_1| \cdot |t_{\Delta 2}| + |\bar{v}_1| \cdot |t_{\Delta 2}| - |\bar{v}_1| \cdot |t_{\Delta 1}| \\ &= (|\bar{v}_2| - |\bar{v}_1|) \cdot |t_{\Delta 2}| + |\bar{v}_1| \cdot (|t_{\Delta 2}| - |t_{\Delta 1}|) \\ &= \Delta|\bar{v}| \cdot |t_{\Delta 2}| + |\bar{v}_1| \cdot \Delta|t_{\Delta}|. \end{aligned}$$

Using $\Delta|x_{\Delta}| = \Delta|\bar{v}| \cdot |t_{\Delta 2}| + |\bar{v}_1| \cdot \Delta|t_{\Delta}|$, we get:

$$[\Delta|x_{\Delta}|] \subseteq [\Delta|\bar{v}|] \cdot [|t_{\Delta 2}|] + [|\bar{v}_1|] \cdot [\Delta|t_{\Delta}|] \quad (\text{theorem 2.1}).$$

We can further simplify this expression using:

$$[|\bar{v}_1|] = [+] \quad \text{since } [\bar{v}_1] \neq [0] \quad \text{and} \quad [|t_{\Delta 2}|] = [t_{\Delta 2}] = [+] \quad \text{since } t_{e2} > t_{b2}$$

and we get:

$$[\Delta|x_{\Delta}|] \subseteq [\Delta|\bar{v}|] + [\Delta|t_{\Delta}|]. \quad \square$$

The precondition of this constraint checks that at least one of the x functions does not return to its initial value at the end of its interval, and therefore has a non-zero average derivative. Since this is clearly satisfied for figure 1, we can insert $[\Delta|x_{\Delta}|] = [+]$ and

$[\Delta|\bar{v}|] = [-]$ in the constraint above and we get $[+] \subseteq [-] + [\Delta|t_\Delta|]$, which is only satisfied for $[\Delta|t_\Delta|] = [+]$, as expected.

It is important to note that we did not need the specific end values of the functions to obtain this results. In section 6.3.1, we show how information extracted from an incompletely described trajectory computed by a qualitative simulator can be used in conjunction with the qualitative average value constraint to compare interval durations.

3.2. Pointwise comparison of functions

The constraints developed so far are useful for comparing real valued variables such as time durations, functions at a single point, or changes of two functions in their intervals. We now describe how entire collections of function values over given intervals can be compared with each other. The discussion starts with two functions on a common interval. This idea is then generalized to functions on different intervals.

3.2.1. Comparison on the same interval

Definition 3.4 (Comparison functions on the same interval). For two functions f_1 and f_2 we construct the comparison functions Δf , Σf , and $\Delta|f|$ on a common interval \mathbf{I} as follows:

- (i) $\Delta f(t) =_{\text{def}} f_2(t) - f_1(t) \ (\forall t \in \mathbf{I})$,
- (ii) $\Sigma f(t) =_{\text{def}} f_2(t) + f_1(t) \ (\forall t \in \mathbf{I})$,
- (iii) $\Delta|f|(t) =_{\text{def}} |f_2(t)| - |f_1(t)| \ (\forall t \in \mathbf{I})$.

Definition 3.5 (Comparison signs of functions on the same interval). Given two functions f_1 , f_2 and their comparison functions Δf , Σf , and $\Delta|f|$ on a common interval \mathbf{I} , if we let F_1 , F_2 , ΔF , ΣF , and $\Delta|F|$ be the images of these functions on \mathbf{I} , the well-defined sign expressions $[\Delta F]$, $[\Sigma F]$, and $[\Delta|F|]$ are called the *pointwise comparison signs*. Specifically, $[\Delta|F|]$ is called the *pointwise magnitude comparison sign*. Moreover, we call $[F_1]$ and $[F_2]$ the *simple sign constants* and using them we define the *compound sign constants* as: $\Sigma[F] =_{\text{def}} [F_2] + [F_1]$ and $\Delta[F] =_{\text{def}} [F_2] - [F_1]$.

Proposition 3.1 (Trivial pointwise comparison constraints). Given two functions f_1 , f_2 compared on a common interval \mathbf{I} , the following constraints hold:

- (i) (a) $[F_1] = [F_2] = [0] \leftrightarrow$
 (b) $\Delta[F] = [0] \leftrightarrow$
 (c) $\Sigma[F] = [0] \leftrightarrow$
 (d) $[\Delta|F|] = [\Delta F] = [\Sigma F] = [0]$.
- (ii) $[\Sigma F] \subseteq \Sigma[F]$.
- (iii) $[\Delta F] \subseteq \Delta[F]$.

$$(iv)(a) [F_1] \neq [?] \wedge [F_2] \neq [?] \leftrightarrow \Delta[F] \neq [?] \vee \Sigma[F] \neq [?].$$

$$(iv)(b) \Delta[F] \neq [?] \vee \Sigma[F] \neq [?] \rightarrow [\Delta F] \neq [?] \vee [\Sigma F] \neq [?].$$

Proof. (i) (a) \leftrightarrow (b) and (a) \leftrightarrow (c) are verified using the operator tables of addition and subtraction on S' ($\forall t \in \mathbf{I}$).

$$(a) \rightarrow f_1(t) = 0, f_2(t) = 0 \rightarrow \Sigma f(t) = \Delta f(t) = \Delta|f|(t) = 0 \rightarrow (d).$$

$$(d) \rightarrow f_2(t) - f_1(t) = f_2(t) + f_1(t) = 0 \rightarrow f_1(t) = f_2(t) = 0 \rightarrow (a).$$

$$(ii) \Sigma f(t) = f_2(t) + f_1(t) \rightarrow [\Sigma F] \subseteq [F_1] + [F_2] \text{ by theorem 2.2(ii).}$$

(iii) Similar to (ii).

(iv)(a) Follows from proposition 2.1(iii).

(iv)(b) Follows from (ii), (iii). \square

As two functions f_1 and f_2 are compared on \mathbf{I} , the sign constants, $[F_1]$, $[F_2]$, and therefore $\Delta[F]$ and $\Sigma[F]$ can be computed by examining each function independently. Specifically, in the qualitative simulation application to be presented in the second part of the paper, these values will be trivially extractable from a partial trajectory. On the other hand, information about the comparison signs is more difficult to obtain. The next theorem establishes the relation between the comparison signs and the sign constants.

Theorem 3.3 (Pointwise comparison conversion constraints). Given two functions f_1, f_2 compared on a common interval \mathbf{I} ,

$$(i) [\Delta|F|] = [\Delta F] \cdot [\Sigma F] \text{ if } \Delta[F] \neq [?] \vee \Sigma[F] \neq [?],$$

$$(ii) [\Sigma F] = \Sigma[F] \text{ if } \Sigma[F] \neq [?],$$

$$(iii) [\Delta F] = \Delta[F] \text{ if } \Delta[F] \neq [?],$$

$$(iv) [\Sigma F] = [\Delta|F|] \cdot \Delta[F] \text{ if } \Delta[F] \neq [?],$$

$$(v) [\Delta F] = [\Delta|F|] \cdot \Sigma[F] \text{ if } \Sigma[F] \neq [?].$$

Proof. (i) For two real numbers r_1 and r_2 , we know that

$$[\Delta|r|] = [\Delta r] \cdot [\Sigma r] \quad (\text{theorem 3.1(i)(a)}).$$

So for f_1 and f_2 , $\forall t \in \mathbf{I}$ letting $r_1 = f_1(t)$ and $r_2 = f_2(t)$ we get:

$$[\Delta|f|(t)] = [\Delta f(t)] \cdot [\Sigma f(t)] \quad (\text{definitions 3.1 and 3.4}).$$

$\Delta[F] \neq [?] \vee \Sigma[F] \neq [?]$ implies that $[\Delta F] \neq [?] \vee [\Sigma F] \neq [?]$ (proposition 3.1(iv)(b)).

Hence by proposition 2.4 we get:

$$[\Delta|F|] = [\Delta F] \cdot [\Sigma F].$$

(ii), (iii) By proposition 3.1(ii), (iii), using proposition 2.1(i).

(iv) Assume $\Delta[F] \neq [?]$ (therefore $[\Delta F] = \Delta[F] \neq [?]$ by (iii)).

Case 1. $[\Delta F] = \Delta[F] = [0]$. By proposition 3.1(i).

Case 2. $[\Delta F] = \Delta[F] \neq [0]$. Since $[\Delta F] = \Delta[F] \neq [0]$, $[?]$, we can convert (i) using proposition 2.1(ii): $[\Delta|F|] \cdot [\Delta F] = [\Sigma F]$ and get: $[\Sigma F] = [\Delta|F|] \cdot \Delta[F]$.

(v) Similar to (iv). \square

If functions f_1 and f_2 are compared on intervals where they possess the same sign uniformly ($[F_1], [F_2] \neq [?]$), then it must be the case that

$$\Delta[F] \neq [?] \quad \text{or} \quad \Sigma[F] \neq [?] \quad (\text{proposition 3.1(iv)(a)}),$$

ensuring that the preconditions of clause (i), and one of the pairs (iii), (iv) or (ii), (v) in theorem 3.3 are always satisfied. On the other hand, if one of the functions does not have a uniform sign ($[F_1] \neq [?] \vee [F_2] \neq [?]$), none of these constraints are satisfied. So we focus on cases where the functions have uniform signs.

Next, we establish how comparison signs propagate over the derivative relation.

Theorem 3.4 (Abstraction of the integral of a function). Given a continuous function h and its integral $h_I(t) = \int_{t_b}^t h(s) ds$ on the interval $[t_b, t_e]$, if we name their images on $\mathbf{I} = (t_b, t_e)$ H and H_I , we get:

$$(i) [h_I(t)] \subseteq [H] \quad \forall t \in (t_b, t_e),$$

$$(ii) [H_I] \subseteq [H].$$

Proof. If $[H] = [?]$, both (i) and (ii) are trivial.

$$\begin{aligned} [H] = [+] &\rightarrow [h(t)] = [+] \quad (\forall t \in \mathbf{I}) \\ &\rightarrow [h_I(t)] = \left[\int_{t_b}^t h(s) ds \right] = [+] \quad (\forall t \in (t_b, t_e)) \end{aligned}$$

because $t > t_b$ and h is continuous, proving (i).

h_I does not change sign on \mathbf{I} , therefore $[H_I] = [+]$, satisfying (ii). The cases for the assignments $[H] = [-]$ and $[H] = [0]$ are similar. \square

Theorem 3.5 (Qualitative fundamental theorem of calculus). For a function h with a continuous derivative h' , if H and H' are their images on the interval $\mathbf{I} = (t_b, t_e)$, the following holds ($h_b = h(t_b)$):

$$[H] \subseteq [h_b] + [H'].$$

Proof. Let us define h'_I on \mathbf{I} such that $h'_I(t) = \int_{t_b}^t h'(s) ds$ and call its image H'_I .

In this notation, the fundamental theorem of calculus is $h(t) = h_b + h'_I(t)$, which yields

$$\begin{aligned} [H] &\subseteq [h_b] + [H'_I] \quad \text{by theorem 2.2(ii)} \\ &\subseteq [h_b] + [H'] \quad \text{since we have } [H'_I] \subseteq [H'] \text{ by theorem 3.4(ii).} \end{aligned} \quad \square$$

Theorem 3.6 (Comparison propagation over derivative). For two functions f_1 and f_2 with continuous derivatives f'_1 and f'_2 on $\mathbf{I} = (t_b, t_e)$, and their initial points $f_{b1} = f_1(t_b)$ and $f_{b2} = f_2(t_b)$ on \mathbf{I} such that using these three pairs, the comparison signs $[\Delta F]$, $[\Sigma F]$, $[\Delta F']$, $[\Sigma F']$, $[\Delta f_b]$, and $[\Sigma f_b]$ are well-defined, the following relations hold:

$$(i) \quad [\Delta F] \subseteq [\Delta f_b] + [\Delta F'],$$

$$(ii) \quad [\Sigma F] \subseteq [\Sigma f_b] + [\Sigma F'].$$

Proof. (i) We let $h(t) = \Delta f(t)$ and insert the following into theorem 3.5:

$$(a) \quad [H] = [\Delta F],$$

$$(b) \quad [h_b] = [\Delta f_b] \quad (h_b = \Delta f(t_b) = f_2(t_b) - f_1(t_b) = f_{b2} - f_{b1} = \Delta f_b),$$

$$(c) \quad [H'] = [\Delta F'] \quad (h'(t) = d\Delta f(t)/dt = f'_2(t) - f'_1(t)).$$

Moreover, h' is continuous since f'_1 and f'_2 are.

(ii) Similar to (i). □

Although the above constraints are not written in terms of magnitude comparison signs, they can be used in conjunction with theorems 3.1 and 3.3 to propagate magnitude comparisons over derivative relations. For example, let us consider the simple case where both the functions and their derivatives are always positive:

$$[F_1] = [F_2] = [f_{b1}] = [f_{b2}] = [F'_1] = [F'_2] = [+].$$

If we also know that the second function has larger initial magnitude, and pointwise larger speed:

$$[\Delta |f_b|] = [+], \quad [\Delta |F'|] = [+],$$

we get:

$$\begin{aligned} [\Delta F'] &= [\Delta |F'|] \cdot \Sigma[F'] = [+] \text{ since } \Sigma[F'] = [+] && \text{(theorem 3.3(v))}, \\ [\Sigma f_b] &= [+] \text{ since } \Sigma[f_b] = [+] && \text{(theorem 3.1(ii))}, \\ [\Delta f_b] &= [\Delta |f_b|] \cdot [\Sigma f_b] = [+] \text{ since } [\Sigma f_b] = [+] && \text{(theorem 3.1(i)(c))}, \\ [\Delta F] &\subseteq [\Delta f_b] + [\Delta F'] = [+] \rightarrow [\Delta F] = [+] && \text{(theorem 3.6(i))}, \\ [\Sigma F] &= \Sigma[F] = [+] \text{ since } \Sigma[F] = [+] && \text{(theorem 3.3(ii))}, \\ [\Delta |F|] &= [\Delta F] \cdot [\Sigma F] = [+] \text{ since } \Sigma[F] = [+] && \text{(theorem 3.3(i))}. \end{aligned}$$

See [8] for the description of further constraints similar to theorem 3.6 that facilitate propagation of comparisons over the other modeling primitives of the qualitative representation; namely, multiplication, addition, and monotonic functional relations. For multiplication and addition, our constraints are provably optimal for the given information.

3.2.2. Comparison on different intervals

Our strategy for comparing two functions on different intervals will be to construct a single comparison interval, and to shift the compared functions so that they are defined on this interval.

Definition 3.6 (Minimum interval). Given two intervals $I_1 = (t_{b1}, t_{e1})$ and $I_2 = (t_{b2}, t_{e2})$, we define the *minimum interval* of I_1 and I_2 as $I = (0, t_\Delta)$, where $t_\Delta = \min(t_{\Delta 1}, t_{\Delta 2})$.

Definition 3.7 (Directional functions). Given a pair of functions f_1, f_2 on the intervals $I_1 = (t_{b1}, t_{e1}), I_2 = (t_{b2}, t_{e2})$, we define, for each f_j ($j = 1, 2$), two functions, the *forward directional function* f_j^\rightarrow , and the *backward directional function* f_j^\leftarrow as follows:

- (i) $f_j^\rightarrow(t) = f_j(t_{bj} + t)$,
- (ii) $f_j^\leftarrow(t) = f_j(t_{ej} - t)$.

For a function f_j , the function f_j^\rightarrow starts at the initial point of f_j and traces its values in the forward direction for the duration of the minimum interval. Similarly, f_j^\leftarrow starts at the end point of f_j and traces it backward. Figure 2 shows two functions f_1 and f_2 with their corresponding directional functions.

By comparing each directional function of f_1 with each directional function of f_2 , we get four different sets of comparison functions and corresponding comparison signs. To distinguish these, we use the directions as superscripts.

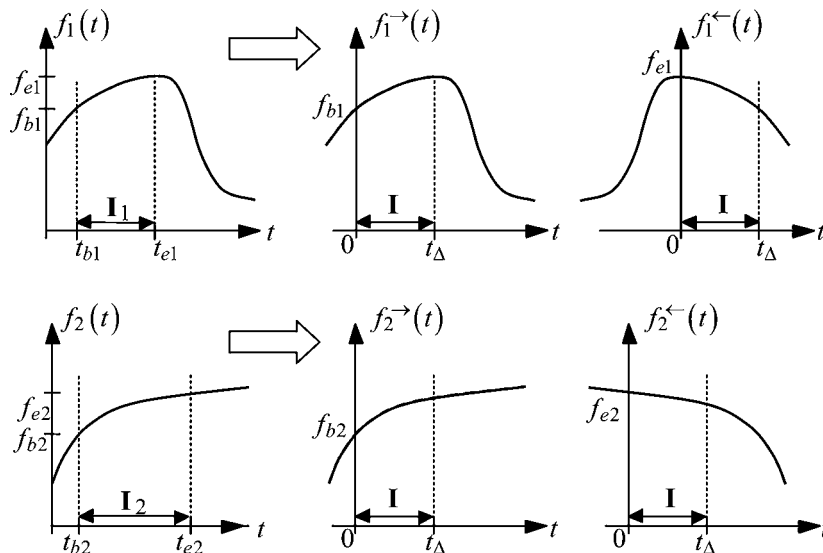


Figure 2. Constructing directional functions.

Definition 3.8 (Directional comparison functions). Given a pair of functions f_1, f_2 well-defined on $\mathbf{I}_1, \mathbf{I}_2$, we define the *directional comparison functions* $\Delta f^{\alpha\beta}(t), \Sigma f^{\alpha\beta}(t)$ and $\Delta|f|^{\alpha\beta}(t)$ on the minimum interval \mathbf{I} to be the comparison functions of the directional functions f_1^α and f_2^β on \mathbf{I} where α and β are two directions from the set $\{\rightarrow, \leftarrow\}$:

- (i) $\Delta f^{\alpha\beta}(t) = f_2^\beta(t) - f_1^\alpha(t),$
- (ii) $\Sigma f^{\alpha\beta}(t) = f_2^\beta(t) + f_1^\alpha(t),$
- (iii) $\Delta|f|^{\alpha\beta}(t) = |f_2^\beta(t)| - |f_1^\alpha(t)|.$

For example, if we compare f_1^\leftarrow , the backward directional function of f_1 , with f_2^\rightarrow , the forward directional function of f_2 , we will get the outward comparison functions $\Delta f^{\leftrightarrow}, \Sigma f^{\leftrightarrow}$, and $\Delta|f|^{\leftrightarrow}$ using which, the outward comparison signs $[\Delta F^{\leftrightarrow}], [\Sigma F^{\leftrightarrow}]$, and $[\Delta|F|^{\leftrightarrow}]$ are constructed. The names and simplified superscripts of all comparison signs are given in table 4. Figure 3 depicts the forward and inward comparison of the functions f_1 and f_2 from figure 2.

The constraints proven earlier (theorems 3.3 and 3.6) for functions on a common interval can help reasoning about different domains if we have a way of mapping directional expressions to the single-domain vocabulary. The next theorem bridges this gap.

Table 4
Directional comparison signs.

Functions	Name	Superscript	Comparison signs
$f_1^\rightarrow, f_2^\rightarrow$	forward	" \rightarrow "	$[\Delta F ^{\rightarrow}], [\Sigma F^{\rightarrow}], [\Delta F^{\rightarrow}]$
$f_1^\leftarrow, f_2^\leftarrow$	backward	" \leftarrow "	$[\Delta F ^{\leftarrow}], [\Sigma F^{\leftarrow}], [\Delta F^{\leftarrow}]$
$f_1^\leftarrow, f_2^\rightarrow$	outward	" \leftrightarrow "	$[\Delta F ^{\leftrightarrow}], [\Sigma F^{\leftrightarrow}], [\Delta F^{\leftrightarrow}]$
$f_1^\rightarrow, f_2^\leftarrow$	inward	" $\rightarrow\leftarrow$ "	$[\Delta F ^{\rightarrow\leftarrow}], [\Sigma F^{\rightarrow\leftarrow}], [\Delta F^{\rightarrow\leftarrow}]$

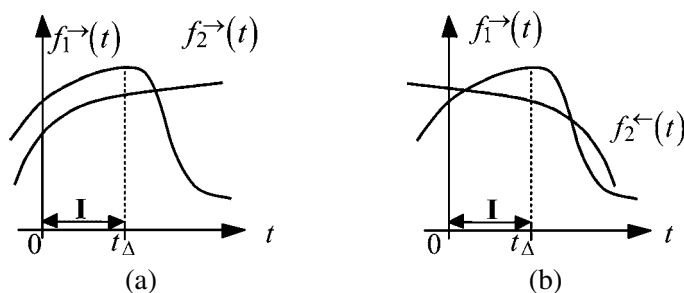


Figure 3. Directional comparison examples of f_1 and f_2 from figure 2: (a) forward comparison, (b) inward comparison. (a) $[\Delta|F|^{\rightarrow}] = [-], [\Delta F^{\rightarrow}] = [-], [\Sigma F^{\rightarrow}] = [+]$, (b) $[\Delta|F|^{\rightarrow\leftarrow}] = [?], [\Delta F^{\rightarrow\leftarrow}] = [?], [\Sigma F^{\rightarrow\leftarrow}] = [+]$.

Theorem 3.7 (Mapping directional functions to comparison constraints). Given two functions v_1 and v_2 , and their derivatives a_1 and a_2 , if the directional functions of v_1 and v_2 on \mathbf{I} in the directions α_1 and α_2 are named f_1 and f_2 , such that $f_1(t) = v_1^{\alpha_1}(t)$, $f_2(t) = v_2^{\alpha_2}(t)$, then the following equalities hold ($j = 1, 2$):

- (i) $f'_j(t) = \begin{cases} a_j^{\rightarrow}(t) & \text{if } \alpha_j = \text{'}\rightarrow\text{'}, \\ -a_j^{\leftarrow}(t) & \text{if } \alpha_j = \text{'}\leftarrow\text{'}. \end{cases}$
- (ii) $f_{bj} = \begin{cases} v_{bj} & \text{if } \alpha_j = \text{'}\rightarrow\text{'}, \\ v_{ej} & \text{if } \alpha_j = \text{'}\leftarrow\text{'}. \end{cases}$
- (iii) $[\Delta|F'|] = [\Delta|A|^{\alpha_1\alpha_2}]$.
- (iv) $[F_j] = [V_j]$ if $[V_j] \neq [?]$.
- (v) $[F'_j] = \begin{cases} [A_j] & \text{if } \alpha_j = \text{'}\rightarrow\text{' and } [A_j] \neq [?], \\ -[A_j] & \text{if } \alpha_j = \text{'}\leftarrow\text{' and } [A_j] \neq [?]. \end{cases}$
- (vi) $[\Delta|V|^{\alpha_1\alpha_2}] = [\Delta|F|]$.

Proof.

- (i) $f'_j(t) = \begin{cases} v'_j(t_{bj} + t) = a_j(t_{bj} + t) = a_j^{\rightarrow}(t) & \text{if } \alpha_j = \text{'}\rightarrow\text{'}, \\ v'_j(t_{ej} - t) = -a_j(t_{ej} - t) = -a_j^{\leftarrow}(t) & \text{if } \alpha_j = \text{'}\leftarrow\text{'}. \end{cases}$
- (ii) $f_{bj} = \begin{cases} v_j^{\rightarrow}(0) = v_j(t_{bj}) = v_{bj} & \text{if } \alpha_j = \text{'}\rightarrow\text{'}, \\ v_j^{\leftarrow}(0) = v_j(t_{ej}) = v_{ej} & \text{if } \alpha_j = \text{'}\leftarrow\text{'}. \end{cases}$

(iii) Since $|f'_j(t)| = |a_j^{\alpha_j}(t)|$ by (i), we can construct the following chain:

$$\Delta|f'| = |f'_2(t)| - |f'_1(t)| = |a_2^{\alpha_2}(t)| - |a_1^{\alpha_1}(t)| = \Delta|a|^{\alpha_1\alpha_2}(t).$$

(iv) We have $F_j = V_j^{\alpha_j} \subseteq V_j$ and use $[F_j] \subseteq [V_j]$ (proposition 2.1(i)).

(v) Using (i) and $[A_j] \neq [?]$ we get:

$$[F'_j] = \begin{cases} [A_j^{\rightarrow}] = [A_j] & \text{if } \alpha_j = \text{'}\rightarrow\text{'}, \\ -[A_j^{\leftarrow}] = -[A_j] & \text{if } \alpha_j = \text{'}\leftarrow\text{'}. \end{cases}$$

(vi) By definition. □

The utility of this result in propagating directional comparisons over derivative relations will be illustrated in section 6.3.2. These comparisons can be propagated over multiplicative, additive, and monotonic functional relationships as well [8].

The main pointwise comparison constraint, which we will present shortly, uses pointwise comparison signs such as $[\Delta|V|^{\rightarrow}]$ in a similar fashion to the way the qualitative average value constraint uses the average value comparison signs such as $[\Delta|\bar{v}|]$.

We first explain the intuitive idea. Let us assume we observe two cars with positions x_1, x_2 and velocities v_1, v_2 on the intervals $\mathbf{I}_1, \mathbf{I}_2$. Assume that we know:

- $[\Delta|V|^{\rightarrow}] = [+]$: the speed of the second car is pointwise higher in the minimum interval.
- $[V_j] \neq [?], [0]$ for $j = 1, 2$: the velocities are either all-positive or all-negative uniformly in their intervals.
- $[\Delta|x_{\Delta}|] = [-]$: the second car has traveled a shorter distance.

In [8], it is shown that the qualitative average value constraint is not sufficient to make a duration comparison given the above information.

Nevertheless, we can use the following argument to compare the durations: The second car has traveled more distance during the minimum interval \mathbf{I} in the first part of the comparison (since $[\Delta|V|^{\rightarrow}] = [+]$). At the end of \mathbf{I} , either \mathbf{I}_1 or \mathbf{I}_2 finishes. On the other hand, the second car should travel less distance in total ($[\Delta|x_{\Delta}|] = [-]$). Since the cars cannot stop or change direction ($[V_j] \neq [?], [0]$), after the end of \mathbf{I} , the first car should have continued to travel for overcoming the distance traveled by the second car, therefore \mathbf{I}_2 is shorter than \mathbf{I}_1 , that is: $[\Delta|t_{\Delta}|] = [-]$.

Theorem 3.8 (Main pointwise comparison constraint). Given a pair of functions x_1, x_2 and their continuous derivatives v_1, v_2 on the intervals $\mathbf{I}_1 = (t_{b1}, t_{e1}), \mathbf{I}_2 = (t_{b2}, t_{e2})$, such that $[V_1] \neq [?], [0]$ and $[V_2] \neq [?], [0]$, the following constraint holds for any two directions α, β :

$$[\Delta|x_{\Delta}|] \subseteq [\Delta|V|^{\alpha\beta}] + [\Delta|t_{\Delta}|].$$

Proof. $t_{\Delta} = \min(t_{\Delta 1}, t_{\Delta 2})$ as in definition 3.6.

$$\begin{aligned} \Delta|x_{\Delta}| &= |x_{\Delta 2}| - |x_{\Delta 1}| = \left| \int_0^{t_{\Delta 2}} v_2^{\beta}(s) \, ds \right| - \left| \int_0^{t_{\Delta 1}} v_1^{\alpha}(s) \, ds \right| \\ &= \int_0^{t_{\Delta 2}} |v_2^{\beta}(s)| \, ds - \int_0^{t_{\Delta 1}} |v_1^{\alpha}(s)| \, ds \\ &= \int_0^{t_{\Delta}} |v_2^{\beta}(s)| \, ds + \int_{t_{\Delta}}^{t_{\Delta 2}} |v_2^{\beta}(s)| \, ds - \int_0^{t_{\Delta}} |v_1^{\alpha}(s)| \, ds \\ &\quad - \int_{t_{\Delta}}^{t_{\Delta 1}} |v_1^{\alpha}(s)| \, ds \quad (\text{since } [V_j] \neq [?]). \end{aligned}$$

Combining the first and third terms using definition 3.8(iii):

$$\Delta|x_{\Delta}| = \int_0^{t_{\Delta}} \Delta|v|^{\alpha\beta}(s) \, ds + \int_{t_{\Delta}}^{t_{\Delta 2}} |v_2^{\beta}(s)| \, ds - \int_{t_{\Delta}}^{t_{\Delta 1}} |v_1^{\alpha}(s)| \, ds.$$

Letting

$$\varepsilon = \int_{t_{\Delta}}^{t_{\Delta 2}} |v_2^{\beta}(s)| \, ds - \int_{t_{\Delta}}^{t_{\Delta 1}} |v_1^{\alpha}(s)| \, ds$$

we get:

$$\Delta|x_{\Delta}| = \int_0^{t_{\Delta}} \Delta|v|^{\alpha\beta}(s) ds + \varepsilon.$$

Let $h(t) = \Delta|v|^{\alpha\beta}(t)$, $h_I(t) = \int_0^t h(s) ds$ (h is continuous like v_1 and v_2)

$$\Delta|x_{\Delta}| = h_I(t_{\Delta}) + \varepsilon.$$

This is abstracted to:

$$\begin{aligned} [\Delta|x_{\Delta}|] &\subseteq [h_I(t_{\Delta})] + [\varepsilon] && \text{(theorem 2.2(ii))} \\ &\subseteq [H] + [\varepsilon] && \text{(theorem 3.4(i))} \end{aligned}$$

hence:

$$[\Delta|x_{\Delta}|] \subseteq [\Delta|V|^{\alpha\beta}] + [\varepsilon].$$

We consider three possible values for $[\Delta|t_{\Delta}|]$ ($[\Delta|t_{\Delta}|] \neq [?]$ because $t_{\Delta 1}$ and $t_{\Delta 2}$ are reals).

Case 1. $[\Delta|t_{\Delta}|] = [+]$. Then $t_{\Delta 2} > t_{\Delta 1} = t_{\Delta}$, therefore,

$$[\varepsilon] = \left[\int_{t_{\Delta}}^{t_{\Delta 2}} |v_2^{\beta}(s)| ds \right] = [+]$$
 (since $[V_2] \neq [0], [?]$).

Case 2. $[\Delta|t_{\Delta}|] = [-]$. Then $t_{\Delta 1} > t_{\Delta 2} = t_{\Delta}$, therefore,

$$[\varepsilon] = \left[- \int_{t_{\Delta}}^{t_{\Delta 1}} |v_1^{\alpha}(s)| ds \right] = [-]$$
 (since $[V_1] \neq [0], [?]$).

Case 3. $[\Delta|t_{\Delta}|] = [0]$. Then $t_{\Delta 1} = t_{\Delta 2} = t_{\Delta}$, therefore,

$$[\varepsilon] = [0].$$

Since $[\varepsilon] = [\Delta|t_{\Delta}|]$ holds in all cases, we conclude:

$$[\Delta|x_{\Delta}|] \subseteq [\Delta|V|^{\alpha\beta}] + [\Delta|t_{\Delta}|]. \quad \square$$

The result obtained in the example presented just before theorem 3.8 can be replicated using this new constraint. Since the preconditions of the main pointwise comparison constraint are satisfied in the example situation, we can insert $[\Delta|V|^{\rightarrow}] = [+]$ and $[\Delta|x_{\Delta}|] = [-]$ into that constraint to obtain $[-] \subseteq [+] + [\Delta|t_{\Delta}|]$, implying $[\Delta|t_{\Delta}|] = [-]$ as expected.

The comparison calculus rules derived in this section play an important role in the development of the new qualitative simulation filter to be presented in the remainder of the paper. A comparison of this formalism and Weld's *differential qualitative (DQ) analysis* [16] technique, a method developed for comparing different behaviors of the same function for perturbation analysis purposes, will be presented in section 9. See [8] for a much more detailed presentation of the comparison calculus, and its applications to other qualitative reasoning tasks.

PART II. IMPOSING DURATION CONSISTENCY IN QUALITATIVE SIMULATION

4. Qualitative simulation

In this section, we give a brief overview of the QSIM [10] qualitative simulation algorithm's aspects relevant to our work.

The *variables* of a system modeled in QSIM are continuously differentiable functions of time. Each variable has a *quantity space*; a totally ordered collection of symbols (*landmarks*) representing important real values that it can take. QSIM has the ability of asserting new landmarks during simulation. The points and intervals in its quantity space make up the set of possible *qualitative magnitudes* (denoted *qmag*) of a variable. The *qualitative direction* (*qdir*) of a variable is defined to be the sign of its derivative. We will also use the symbols “↑”, “↓” and “⊖” to denote, respectively, the values [+], [−], and [0] for qualitative directions. A variable's *qualitative value* is the pair consisting of its qualitative magnitude and qualitative direction. For example, $\langle(0, +\infty), \uparrow\rangle$ describes a variable with a positive increasing value. The collection of the qualitative values of a system's variables makes up its *state*.

The “laws” according to which the system operates are represented by *constraints* describing time-independent relations between the variables. At each step of the simulation, QSIM uses a set of transition rules to implicitly generate all possible “next” qualitative values of the variables. The combinations of these values are filtered so that only those which constitute complete and legal states for the system remain. The constraints supply “checklists” during this filtering; every constraint must still be satisfied by the newly proposed values of its variables.

There are seven “basic” types of constraints in QSIM: *addition*, *constant function*, d/dt , M^+ , M^- , *minus*, and *multiplication*. Each type of constraint imposes a different kind of relation on its arguments. For example, if we have the constraint *minus*(x, y), which just stands for $\forall t(x(t) = -y(t))$, then any combination of variable states in which variables x and y have the same (nonzero) sign in their magnitudes or directions will be filtered out. The *monotonic function* constraints M^+ and M^- are the ingredients that make QSIM models correspond to infinitely many ordinary differential equations: An M^+ (M^-) relationship between two variables z and w just indicates that we know the existence of a function f , such that $z = f(w)$, and f' is positive (negative) throughout its domain.

QSIM generates a tree of system states to represent the possible solutions of the model. The root of this tree is the initial state with time label t_0 . Every path from the root to a leaf is a predicted *behavior* of the system. Time point and interval states appear alternately in behaviors. If a state in which all variables have point magnitudes appears twice in a behavior, simulation ends on that branch, since this corresponds to a “cycle” that will repeat forever.

Spurious behaviors do not correspond to any solution consistent with the model and the initial state. Faced with the inadequacy of the individual constraints in “locally” filtering some spurious behaviors by looking only at the information in the current state,

QSIM uses a set of *global filters*, which examine different mathematical properties of the entire history of the variables to eliminate inconsistent candidate states.

5. Duration consistency filtering: The basic idea

Let us begin with an example showing the kind of spurious behaviors that we will deal with. Consider a system (figure 4) consisting of two balls thrown upward from ground level at the same moment. The simulator is set to stop extending a prediction when either ball hits the ground, that is, at time-points where h_1 or h_2 has the value $(0, \downarrow)$.

The QSIM algorithm predicts 13 distinct behaviors in this simulation. Figure 5 depicts one of these predictions. (Only the height variables are shown.) It is easy to

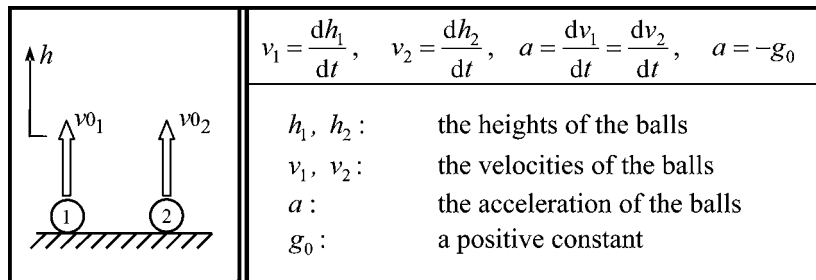


Figure 4. The two-ball system.

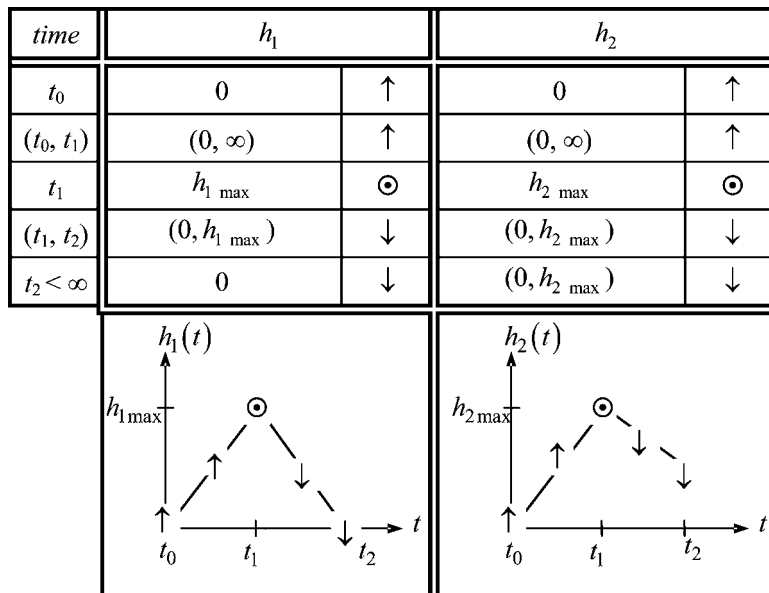


Figure 5. A (spurious) prediction for the two-ball system.

see that this is a spurious prediction, since it describes a behavior in which it takes the balls the same time to reach their maximum heights, but then the first ball overtakes the second ball in the next half of what is clearly a symmetric trajectory.

If a component that detects the symmetry of the trajectories and deduces $|t_0, t_1| = |t_1, t_2|$ from the behavior of the first ball, and $|t_0, t_1| > |t_1, t_2|$ from the behavior of the second one as a result were added to the algorithm, this spurious prediction could be eliminated by noticing the inconsistency. This model generates five other similar spurious predictions that can be eliminated in a similar way.

We define *duration comparison facts* as statements of the form “ $|t_a, t_b| = |t_c, t_d|$ ”, or “ $|t_a, t_b| > |t_c, t_d|$ ” where t_a, t_b, t_c , and t_d are QSIM time points. Duration comparison facts can be described using duration comparison signs presented in the first part of this paper. For example, by letting $\mathbf{I}_1 = (t_a, t_b)$ and $\mathbf{I}_2 = (t_c, t_d)$, these statements will translate to $[\Delta|t_\Delta|] = [0]$ and $[\Delta|t_\Delta|] = [-]$, respectively. In the next section, we describe methods based on analysis of symmetry and periodicity, as well as the comparison calculus, to extract duration comparison facts for a given QSIM model and partially computed behavior. If these facts lead to an inconsistency, the spurious prediction can be eliminated.

6. Extracting duration comparison facts in qualitative simulation

6.1. Symmetric functions

6.1.1. Theory

Symmetry is an important qualitative property. In the next subsection, we will describe how the equations in the input model can be used to deduce the existence of symmetric functions in a partial behavior. This section is an introduction to the terminology and mathematics that will be employed during that procedure.

Definition 6.1 (Symmetric functions). If a continuous function $f(t)$ has, for a given point t_i in its domain $\mathbf{I} = (t_b, t_e)$, the property that

$$f(t_i - s) = f(t_i + s)$$

for all s such that $t_i - s \in \mathbf{I}$ and $t_i + s \in \mathbf{I}$, then f is said to be *even symmetric around* t_i , denoted $even(f, t_i)$.

If a continuous function $f(t)$ has, for a given point t_i in its domain \mathbf{I} , the property that

$$f(t_i - s) = -f(t_i + s)$$

for all s such that $t_i - s \in \mathbf{I}$ and $t_i + s \in \mathbf{I}$, then f is said to be *odd symmetric around* t_i , denoted $odd(f, t_i)$. The positive legal range for s described above, namely, $(0, \min(t_i - t_b, t_e - t_i))$, is called the *symmetry radius around* t_i .

If a function f is (even or odd) symmetric around t_i , t_i is said to be f 's *symmetry point*. In the remainder of this section, all appearances of s are assumed to be universally quantified over the symmetry radius around the symmetry point under discussion. Note that the function $x(t) \equiv 0$ is both even and odd symmetric everywhere in its domain.

Theorem 6.1 (Propagation of symmetry in composition relation). Given $y(t) = f(x(t))$,

- (i) $even(x, t_i) \rightarrow even(y, t_i)$,
- (ii) $odd(x, t_i) \wedge odd(f, 0) \rightarrow odd(y, t_i)$.

Proof.

- (i) $y(t_i - s) = f(x(t_i - s)) = f(x(t_i + s)) = y(t_i + s)$,
- (ii) $y(t_i - s) = f(x(t_i - s)) = f(-x(t_i + s)) = -f(x(t_i + s)) = -y(t_i + s)$. \square

The following theorems establish the correctness of a set of rules used by the symmetry recognition procedure incorporated to QSIM.

Theorem 6.2 (Symmetry of constant functions). For a constant k , $x(t) = k$ is even symmetric around every point.

Theorem 6.3 (Propagation of symmetry in addition relation). Given $x(t) = y(t) + z(t)$,

- (1) if any two of x , y , and z are even symmetric around t_i , then the third one is also even symmetric around t_i :
 - (i) $even(y, t_i) \wedge even(z, t_i) \rightarrow even(x, t_i)$,
 - (ii) $even(x, t_i) \wedge even(z, t_i) \rightarrow even(y, t_i)$,
 - (iii) $even(x, t_i) \wedge even(y, t_i) \rightarrow even(z, t_i)$;
- (2) if any two of x , y , and z are odd symmetric around t_i , then the third one is also odd symmetric around t_i :
 - (iv) $odd(y, t_i) \wedge odd(z, t_i) \rightarrow odd(x, t_i)$,
 - (v) $odd(x, t_i) \wedge odd(z, t_i) \rightarrow odd(y, t_i)$,
 - (vi) $odd(x, t_i) \wedge odd(y, t_i) \rightarrow odd(z, t_i)$.

- Proof.*
- (i) $x(t_i - s) = y(t_i - s) + z(t_i - s) = y(t_i + s) + z(t_i + s) = x(t_i + s)$.
 - (ii) $y(t_i - s) = x(t_i - s) - z(t_i - s) = x(t_i + s) - z(t_i + s) = y(t_i + s)$.
 - (iii) Similar to the proof of (ii).
 - (iv) $x(t_i - s) = y(t_i - s) + z(t_i - s) = -y(t_i + s) - z(t_i + s) = -x(t_i + s)$.
 - (v) $y(t_i - s) = x(t_i - s) - z(t_i - s) = -x(t_i + s) + z(t_i + s) = -y(t_i + s)$.
 - (vi) Similar to the proof of (v). \square

Theorem 6.4 (Propagation of symmetry in multiplication relation). Given $x(t) = y(t) \cdot z(t)$,

- (1) if any two of x , y , and z are even symmetric around t_i , then the third one is also even symmetric around t_i :
 - (i) $even(y, t_i) \wedge even(z, t_i) \rightarrow even(x, t_i)$,
 - (ii) $even(x, t_i) \wedge even(z, t_i) \rightarrow even(y, t_i)$ if $z(t) \neq 0$ for $t \in (t_b, t_e)$,
 - (iii) $even(x, t_i) \wedge even(y, t_i) \rightarrow even(z, t_i)$ if $y(t) \neq 0$ for $t \in (t_b, t_e)$;
- (2) if any two of x , y , and z are odd symmetric around t_i then the third one is even symmetric around t_i :
 - (iv) $odd(y, t_i) \wedge odd(z, t_i) \rightarrow even(x, t_i)$,
 - (v) $odd(x, t_i) \wedge odd(z, t_i) \rightarrow even(y, t_i)$ if $z(t) \neq 0$ for $t \in (t_b, t_e)$,
 - (vi) $odd(x, t_i) \wedge odd(y, t_i) \rightarrow even(z, t_i)$ if $y(t) \neq 0$ for $t \in (t_b, t_e)$;
- (3) if any one of x , y , and z is odd symmetric and another one is even symmetric around t_i , then the remaining one is odd symmetric around t_i :
 - (vii) $even(y, t_i) \wedge odd(z, t_i) \rightarrow odd(x, t_i)$,
 - (viii) $even(z, t_i) \wedge odd(y, t_i) \rightarrow odd(x, t_i)$,
 - (ix) $even(x, t_i) \wedge odd(z, t_i) \rightarrow odd(y, t_i)$ if $z(t) \neq 0$ for $t \in (t_b, t_e)$,
 - (x) $even(z, t_i) \wedge odd(x, t_i) \rightarrow odd(y, t_i)$ if $z(t) \neq 0$ for $t \in (t_b, t_e)$,
 - (xi) $even(x, t_i) \wedge odd(y, t_i) \rightarrow odd(z, t_i)$ if $y(t) \neq 0$ for $t \in (t_b, t_e)$,
 - (xii) $even(y, t_i) \wedge odd(x, t_i) \rightarrow odd(z, t_i)$ if $y(t) \neq 0$ for $t \in (t_b, t_e)$.

Proof. (i) $x(t_i - s) = y(t_i - s)z(t_i - s) = y(t_i + s)z(t_i + s) = x(t_i + s)$.

(ii) $y(t_i - s) = x(t_i - s)/z(t_i - s) = x(t_i + s)/z(t_i + s) = y(t_i + s)$.

(iii) Similar to the proof of (ii).

(iv) $x(t_i - s) = y(t_i - s)z(t_i - s) = (-y(t_i + s))(-z(t_i + s)) = x(t_i + s)$.

(v) $y(t_i - s) = x(t_i - s)/z(t_i - s) = -x(t_i + s)/(-z(t_i + s)) = y(t_i + s)$.

(vi)–(xii) Similar to the above proofs. \square

Theorem 6.5 (Propagation of symmetry in monotonic composition relation). Given $y(t) = f(x(t))$, where $f \in M^+ \cup M^-$,

- (i) $even(x, t_i) \leftrightarrow even(y, t_i)$,
- (ii) if $odd(f, 0)$ ($f(-x) = -f(x)$) then $odd(x, t_i) \rightarrow odd(y, t_i)$.

Proof. If $f \in M^+ \cup M^-$, then $f^{-1} \in M^+ \cup M^-$, and we get the results using theorem 6.1. \square

Substituting s by $(t-t_i)$ in definition 6.1, we see that a function f is even symmetric around t_i if and only if

$$f(t) = f(-t + 2t_i).$$

Similarly, odd symmetry around t_i can be redefined with the equation

$$f(t) = -f(-t + 2t_i).$$

These alternative definitions will be useful in the proof of the next theorem.

Theorem 6.6 (Propagation of symmetry in derivative relation). Given $x = dy/dt$,

- (i) $even(y, t_i) \leftrightarrow odd(x, t_i)$,
- (ii) $odd(y, t_i) \leftrightarrow even(x, t_i) \wedge y(t_i) = 0$.

Proof. (i)(a) Assume $even(y, t_i)$. Then we have $y(t) = y(-t + 2t_i)$. Differentiating, we obtain $x(t) = -x(-t + 2t_i)$ meaning $odd(x, t_i)$.

(b) Assume $odd(x, t_i)$. Letting $v = -\tau + t_i$ and $w = \tau - t_i$,

$$\begin{aligned} y(t_i + s) - y(t_i - s) &= \int_{t_i-s}^{t_i+s} x(\tau) d\tau = \int_{t_i-s}^{t_i} x(\tau) d\tau + \int_{t_i}^{t_i+s} x(\tau) d\tau \\ &= -\int_s^0 x(t_i - v) dv + \int_0^s x(t_i + w) dw \\ &= \int_0^s x(t_i - v) dv + \int_0^s x(t_i + w) dw \\ &= -\int_0^s x(t_i + v) dv + \int_0^s x(t_i + w) dw = 0, \end{aligned}$$

hence $y(t_i + s) = y(t_i - s)$, therefore $even(y, t_i)$.

(ii)(a) Assume $odd(y, t_i)$. This means that $y(t) = -y(-t + 2t_i)$. Differentiating, one obtains $x(t) = x(-t + 2t_i)$, meaning $even(x, t_i)$. Moreover, $y(t_i - s) = -y(t_i + s) \rightarrow y(t_i) = -y(t_i) \rightarrow y(t_i) = 0$.

(b) Assume $even(x, t_i)$ and $y(t_i) = 0$. Letting $v = -\tau + t_i$ and $w = \tau - t_i$,

$$\begin{aligned} y(t_i - s) &= y(t_i) + \int_{t_i}^{t_i-s} x(\tau) d\tau = \int_{t_i}^{t_i-s} x(\tau) d\tau \\ &= -\int_0^s x(t_i - v) dv = -\int_0^s x(t_i + v) dv, \\ y(t_i + s) &= y(t_i) + \int_{t_i}^{t_i+s} x(\tau) d\tau = \int_{t_i}^{t_i+s} x(\tau) d\tau = \int_0^s x(t_i + w) dw, \end{aligned}$$

hence we get $y(t_i - s) = -y(t_i + s)$, therefore $odd(y, t_i)$. \square

How can symmetry information be exploited for comparing durations? Note that the definition of a function x being even symmetric around t_i entails that

$$x(t_i - s) = k \leftrightarrow x(t_i + s) = k$$

which, when translated to the QSIM representation, means the following: If we “see” x to be at a landmark k at a time-point t_a before t_i , then x is “destined” to reach k again at some point t_c after t_i (unless the simulation terminates for another reason.) Furthermore, we can conclude that $|t_a, t_i| = |t_i, t_c|$, and, of course, $|t_a, t_i| < |t_i, t_b|$ for any t_b in which x has not yet reached k after t_i .

For example, assume that x , as illustrated in figure 6, has been discovered to be even symmetric at time-point t_6 , and the list of landmarks crossed by x in $[t_0, t_6]$ is $\{x_a, 0, x_b, 0\}$. “ x_c ” is a new landmark discovered at the symmetry point t_6 . In the continuation of this behavior, it is certain that x will cross the landmarks listed above in the reverse order such as $\{0, x_b, 0, x_a\}$, also crossing x_c along the way to x_a . Whenever x arrives at a landmark in this list, we will be sure that exactly the same amount of time has elapsed from t_i as it took x to reach the symmetry point from the corresponding appearance of that landmark before the symmetry point.

For odd symmetric functions, zero crossings contribute duration comparison facts. To see this, we consider the definition of odd symmetry around t_i , that is: $f(t_i - s) = -f(t_i + s)$, which entails:

$$f(t_i - s) = 0 \leftrightarrow f(t_i + s) = 0.$$

Qualitative directions of odd symmetric variables are useful too. Since the derivative of an odd symmetric variable f will be even symmetric around the symmetry point t_i , it must be the case that

$$f'(t_i - s) = 0 \leftrightarrow f'(t_i + s) = 0,$$

which means that the qualitative direction of x becoming steady s units before t_i forces a “mirror-event” where x stops again s units after t_i .

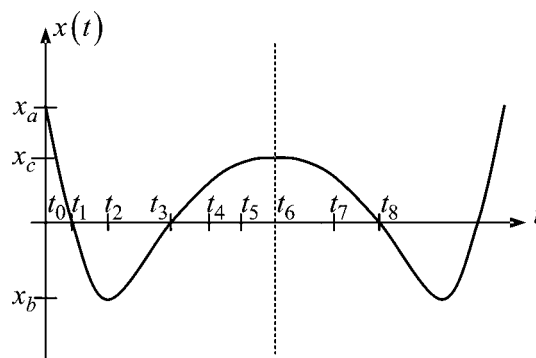


Figure 6. An even symmetric variable.

The next subsection illustrates the algorithm for extracting the duration comparison facts related to symmetric functions in more detail.

6.1.2. Recognizing and using symmetries in qualitative simulation

The theorems in the previous subsection describe the ways in which symmetry information about functions can be propagated through a model. The only way of obtaining symmetry information from “scratch”, as it were, is provided by theorem 6.2. In our modifications which enable QSIM to recognize symmetric variables, the results of theorems 6.2–6.6 are used as rules which add new symmetry data whenever they are able to “fire” in a given state.

We will describe the working of the symmetry recognition procedure using our introductory example about the two-ball system (figure 4). Before the simulation starts, a preprocessor marks variables with constant value to use the rule in theorem 6.2 for deducing symmetry information about the variables. A newly derived piece of information of that kind can cause other rules to fire and the firing of rules continues until no new conclusions can be drawn. At this stage, a , the only constant function in the model, is found to be even symmetric (everywhere) by an application of that rule and no other rule fires. This single item of symmetry information is placed into the *symmetry list*, a structure that will be inherited by all behaviors that are continuations of this state. New rules can only fire if new information is added to the system. The only dynamic information that can start a new rule-firing event is the value of a variable reaching zero during the simulation. If a variable reaches the value zero at a time point t_i and if the derivative of that variable is even symmetrical at that point, the rule of theorem 6.6(ii) fires in the backward direction and this may cause a chain of firings. Since the starting of new firings is only possible at time-points where a variable has the value zero, we can make maximum use of the symmetry derivation rules if we run them just for each completed time-point state. Our modified algorithm therefore submits each time-point state to the set of symmetry rules, and any new symmetry information obtained as a result is added to the symmetry list associated with the current behavior.

In our example for the behavior in figure 5, the state t_1 causes the reasoning steps described in table 5 to be performed. Further simulation of this model does not lead to the discovery of any new symmetry information.

Each candidate time-point state is examined by our algorithm to see if it contributes any new duration comparison facts due to previously discovered symmetries. For this purpose, we make use of the fact that the behavior of a symmetric variable up to the

Table 5
Derivation of new symmetry information from the state at t_1 .

Trigger	Fired rule	Conclusion
a is even everywhere and $v_1(t_1) = 0$	theorem 6.6(ii) backwards	v_1 is odd around t_1
a is even everywhere and $v_2(t_1) = 0$	theorem 6.6(ii) backwards	v_2 is odd around t_1
v_1 is odd around t_1	theorem 6.6(i) backwards	h_1 is even around t_1
v_2 is odd around t_1	theorem 6.6(i) backwards	h_2 is even around t_1

symmetry point determines a prefix of that variable's future behavior, as explained in the previous subsection.

Our algorithm performs that reasoning to assert new duration comparison facts. Each symmetric variable past its symmetry point can contribute one such fact at each time-point. For the even symmetric variable of figure 6, when the symmetry is discovered at t_6 , the behavior history of the variable is traced and for each time point $t_i < t_6$ where the variable is at one of its landmarks l_i , the tuple $\langle l_i, t_i \rangle$ is pushed in a stack such that the tuples with newest time points are at the top of the stack. In this case, the stack created at the symmetry point will look like $\{\langle x_a, t_0 \rangle, \langle 0, t_1 \rangle, \langle x_b, t_2 \rangle, \langle 0, t_3 \rangle\}$ where $\langle 0, t_3 \rangle$ is the top element of the stack. As the simulation continues, the qualitative values of the variable are compared with the stack to extract duration comparison facts. At t_7 , the first global time point after the symmetry point, the simulation does not reach the expected landmark 0 at the top of the stack and as a result, it extracts the duration fact $|t_3, t_6| > |t_6, t_7|$. At t_8 , the simulator detects that it has reached the expected landmark 0, and extracts the duration fact $|t_3, t_6| = |t_6, t_8|$. Since the expected landmark is reached, the tuple $\langle 0, t_3 \rangle$ is pulled from the stack and the simulation continues with the expected landmark x_b .

Odd symmetric functions, which contribute useful duration information when they cross zero and/or "stop", as explained in the previous subsection, are treated using a variant of the procedure described above.

Symmetries of "non-analytic" functions, which stay at the same landmark value for a finite time interval during their behavior (figure 7) are handled in a somewhat more sophisticated way. In this case, only the end points of constant regions such as (t_2, t_3) and (t_5, t_7) are considered, so that premature and wrong results such as $|t_2, t_3| = |t_5, t_6|$ are avoided.

Returning to our two-balls example, the duration fact extraction procedure works as follows when it is called during the creation of state t_2 of figure 5: Variable h_1 is known to be even symmetric around t_1 , and its "before" stack indicates that it is supposed to reach zero exactly $|t_1 - t_0|$ time units after t_1 . The proposed magnitude of zero for h_1 causes the assertion of $|t_0, t_1| = |t_1, t_2|$ to the duration comparison fact list. A similar reasoning about h_2 adds $|t_0, t_1| > |t_1, t_2|$ to the same data structure.

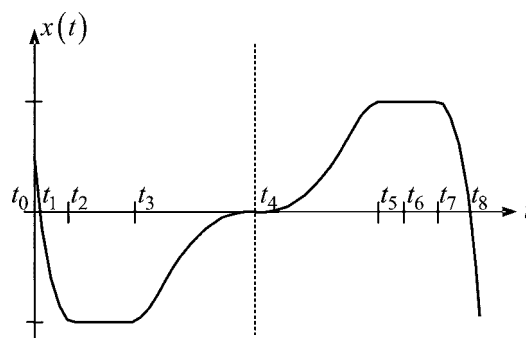


Figure 7. A non-analytic odd symmetric function.

6.2. Periodicity

The QSIM algorithm already has a cycle detection feature which lets it decide that a branch of the state tree corresponds to a periodic behavior and therefore need not be expanded any more. For instance, figure 8 depicts the behavior of the position x in the spring-mass model [10,14] in figure 9. If the entire system consists of these three variables, QSIM will perform simulation only up to time point t_4 , noticing that the current state represents a point identical to state t_0 in the phase space. It is then inferred that the rest of the behavior consists of infinite repetitions of the segment between these two points. Clearly, this lets us conclude that

$$|t_0, t_1| = |t_4, t_5| = |t_8, t_9| = \dots,$$

$$|t_1, t_2| = |t_5, t_6| = |t_9, t_{10}| = \dots,$$

and so on.

Now assume that this three-constraint set is “embedded” in a bigger model, containing other variables and constraints. Since the values of the other variables will probably be different at t_0 and the state corresponding to figure 8’s t_4 , the two global system states will not be identical at these points, and the simulation will continue. But it is clear that the variables x , v , and a will behave periodically throughout all possible behaviors of the overall system. If all three do not have the value $\langle 0, \odot \rangle$ at t_0 , this “clock” subsystem will “tick” at time-points where either v or both x and a reach their zeros.

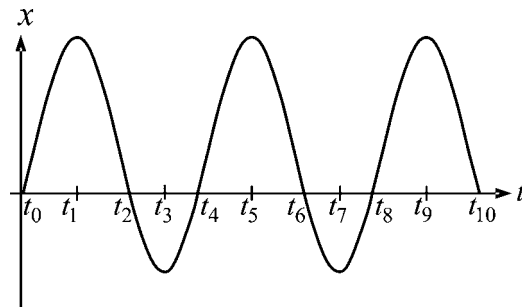


Figure 8. Behavior of a non-linear spring.

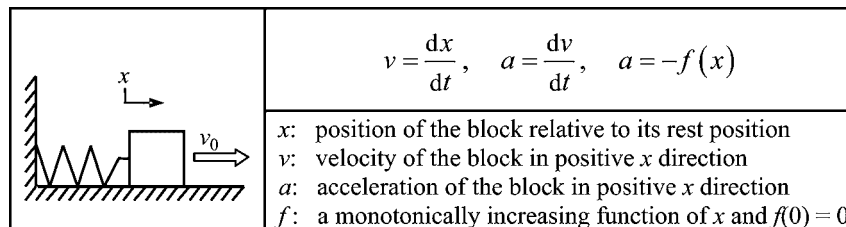


Figure 9. Spring-block system. (This constraint set forms a “clock” subsystem in any greater model.)

This property can be exploited for our purposes. A preprocessor would scan the constraint model for subsets known to cause periodic behavior to see if any embedded clock subsystem can be identified. (These sets, and the behavior segment–duration equality patterns associated with them, can be manually added to a “periodicity library” if an explicit periodicity proof, such as the ones in [10,14] for the spring-block model, is available. Alternatively, QSIM itself may be used as the “mathematician”: Whenever it detects a cycle during a simulation, the model under consideration and the starting state of the cycle can be automatically asserted to the library as a new kind of clock.) If such a clock were found during preprocessing, its variables would be noted for future use. During the global filtering of each time-point state, the current behavior prefix would be examined to see if one of the noted variables has “ticked”, contributing a new duration comparison fact to be used by the duration consistency filter.

Our present implementation of periodicity checking is limited to using, for each clock subsystem, partial qualitative state descriptions called *periodicity templates*, which are known to be repeated periodically. During simulation, each time-point state is examined to see whether it satisfies a periodicity template (of which there may be more than one). If the state at time t_k satisfies a template and if t_i and t_j are the last two time points that have satisfied it before, $|t_i, t_j| = |t_j, t_k|$ is added to the list of duration comparison facts. For each matching template-subsystem pair in memory, only the time points of the last two matching states are stored and whenever a new state satisfies a template, a duration comparison fact is created without examining the entire history of the behavior.

For the case where the subsystem is a spring-block equivalent such as the one in figure 9, we may use the periodicity template $\{qmag(x) = [0], qmag(v) = [+]\}$ and we obtain (for a behavior like figure 8):

$$|t_0, t_4| = |t_4, t_8| = \dots$$

Similarly, $\{qmag(x) = [0], qmag(v) = [-]\}$ asserts:

$$|t_2, t_6| = |t_6, t_{10}| = \dots$$

$\{qmag(x) = [+], qmag(v) = [0]\}$ asserts:

$$|t_1, t_5| = |t_5, t_9| = \dots$$

and finally $\{qmag(x) = [-], qmag(v) = [0]\}$ asserts:

$$|t_3, t_7| = |t_7, t_{11}| = \dots$$

6.3. Multiple traversals of the same interval

In section 3, we developed two comparison calculus rules (theorems 3.2 and 3.8) which can be used for comparing the lengths of time intervals \mathbf{I}_1 and \mathbf{I}_2 , given comparison information about the changes of two “position” variables x_1 and x_2 , and their “speeds” $|v_1|$ and $|v_2|$, over these intervals. These prerequisites can be unambiguously computed from a partially built QSIM behavior when x_1 and x_2 are the same variable, say, x , and the landmark interval spanned by x in one of \mathbf{I}_1 and \mathbf{I}_2 is a subset of the

other one (which forces v_1 and v_2 to be a single “velocity” variable v as well). So we can compare the durations of two traversals of the same landmark interval by the same variable.

6.3.1. Applying the qualitative average value constraint

Let us recall the qualitative average value constraint. Given functions x_1, x_2 and their derivatives v_1 and v_2 :

$$[\Delta|x_\Delta|] \subseteq [\Delta|\bar{v}|] + [\Delta|t_\Delta|] \quad \text{if } [\bar{v}_1] \neq [0] \text{ or } [\bar{v}_2] \neq [0].$$

Comparison of the average speeds in two intervals can be performed via ordinal comparisons on upper and lower bounds. For example, if we know that the velocity is positive in both \mathbf{I}_1 and \mathbf{I}_2 , and all values attained by it during \mathbf{I}_2 are greater than all of its values during \mathbf{I}_1 , we will say that the magnitude of v_2 in \mathbf{I}_2 is *totally greater* than the magnitude of v_1 in \mathbf{I}_1 . This information entails that $|\bar{v}_2| > |\bar{v}_1|$, and hence $[\Delta|\bar{v}|] = [+]$.

For instance, consider the QSIM behaviors of a variable x and its derivative v in figure 10. If we let $\mathbf{I}_1 = (t_0, t_1)$ and $\mathbf{I}_2 = (t_4, t_5)$, we get $[\Delta|\bar{v}|] = [+]$. This can be detected since $qmag(v, \mathbf{I}_2) = (+v_0, +\infty)$ is greater than $qmag(v, \mathbf{I}_1) = (0, +v_0)$. On the other hand, the change of x in both intervals is the same. We can detect this by comparing the end points of the two intervals, and we get $[\Delta|x_\Delta|] = [0]$. As a result, the qualitative average value constraint lets us conclude that $[\Delta|t_\Delta|] = [-]$, i.e.: $|t_4, t_5| < |t_0, t_1|$.

The method we used above for making an average value comparison needs a very strong condition that one velocity is totally greater than the other. In the situation shown in figure 11, although this condition is not satisfied, one can still conclude that $[\Delta|\bar{v}|] = [+]$.

If the “distance” functions are known to have linear velocities, (which requires that a further pair of “acceleration” variables be present in the model, and remain constant for the time intervals in question,) the average value comparison sign can be computed by

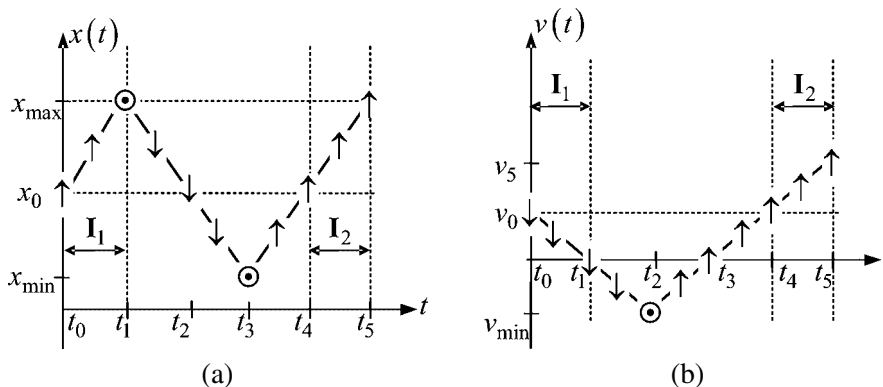


Figure 10. x traverses the same interval with higher speed in \mathbf{I}_2 : (a) position, (b) velocity.

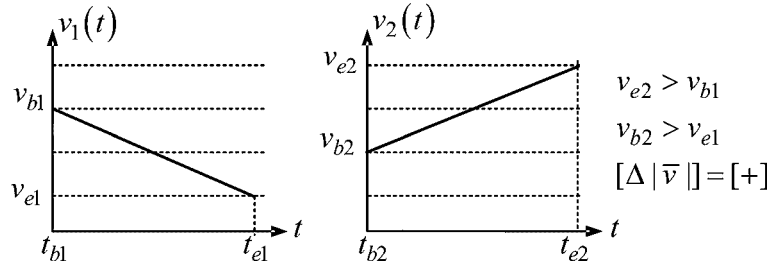


Figure 11. Comparison of linear velocities.

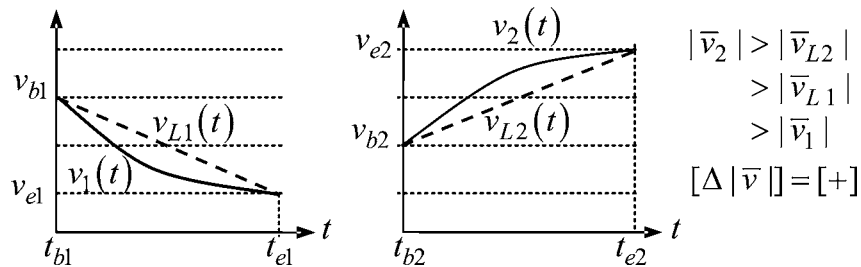


Figure 12. Comparison of average values using linearization.

comparing the end points of the velocities. In figure 11, since $v_{e2} > v_{b1}$ and $v_{b2} > v_{e1}$, and since the velocities are linear, we get:

$$\frac{v_{e2} + v_{b2}}{2} > \frac{v_{e1} + v_{b1}}{2} \quad \text{therefore } \bar{v}_2 > \bar{v}_1$$

and since all velocities are positive, we have $[\Delta|\bar{v}|] = [+]$.

We can generalize this approach to perform average value comparison between some nonlinear velocity functions. Given the velocities v_1 and v_2 in figure 12, we construct the *linearized functions* v_{L1} and v_{L2} by joining the endpoints of v_1 and v_2 with straight lines. Using the same argument we have used for figure 11, we can compare the average values of the linearized functions, and since all functions are positive at all times in this problem, we get $|\bar{v}_{L2}| > |\bar{v}_{L1}|$. Considering the monotonicity (which can be detected from qualitative direction) and the signs of the curvatures (which are just the qualitative directions of the acceleration variables) of v_1 and v_2 , we get $|\bar{v}_{L1}| > |\bar{v}_1|$ and $|\bar{v}_2| > |\bar{v}_{L2}|$. Using these, we can derive $|\bar{v}_2| > |\bar{v}_1|$, which means $[\Delta|\bar{v}|] = [+]$. See [8] for the constraint and sign patterns that enable this kind of deduction under different situations.

6.3.2. Pointwise comparison at change of direction

We can extract more duration comparison facts by observing the time intervals where the variables change direction. Consider the system in figure 13, depicting an upward thrown ball in an elevator with increasing upward acceleration. Figure 14 shows the behaviors of the position x , velocity v and the acceleration a of the ball with respect

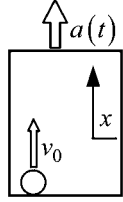
	$v = \frac{dx}{dt}, \quad a = \frac{dv}{dt}, \quad a' = \frac{da}{dt}, \quad a' = -c$
	<p>x: height of the ball (reference frame: elevator)</p> <p>v: velocity of the ball (reference frame: elevator)</p> <p>a: acceleration of the ball (reference frame: elevator)</p> <p>c: a positive constant</p>

Figure 13. Upwards thrown ball in elevator with increasing acceleration.

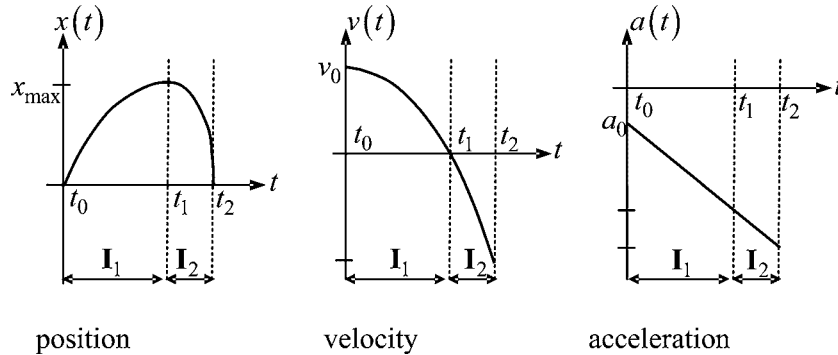


Figure 14. Comparison at change of direction.

to the reference frame of the elevator. (To keep the model simple, we have assumed a constant value for a' , but the only important thing for our calculation is that a is decreasing during the compared intervals.) In this problem, we should be able to deduce $|t_1, t_2| < |t_0, t_1|$.

This result cannot be obtained using the methods described in previous subsection. It is easy to check that the change in position is the same in both intervals, but how should we compare the average speeds? The speeds are clearly not “totally comparable” since it is not the case that the speed at all points in one of the intervals is greater than the speed at all points in the second interval. Moreover, it is shown in [8] that the linearized average value comparison technique cannot be used here to obtain an average speed comparison.

If we superpose the behaviors of two imaginary balls starting at time t_1 at the maximum height with zero velocity and moving according to the graph in figure 14, one backward in time and the other forward, we observe that the ball moving forward in time will have higher speed at each corresponding time point, since its acceleration is greater in magnitude at each corresponding time point. Consequently, we can conclude that the imaginary ball in the forward direction will hit the ground in a shorter time compared to the one in the backward direction, therefore $|t_1, t_2| < |t_0, t_1|$.

This idea can be realized using the tools developed in section 3.2, by comparing the functions x, v, a on the two intervals $\mathbf{I}_1 = (t_0, t_1)$ and $\mathbf{I}_2 = (t_1, t_2)$ in the outward

direction, so that time is “running backwards” in the first interval, and forwards in the second one.

The following quantities can be extracted from the partial QSIM behavior in a straightforward way by examining the qualitative magnitudes of the variables throughout the interval $[t_0, t_2]$:

$$[v_{e1}] = [v_{b2}] = [v(t_1)] = [0], \quad [V_1] = [+], \quad [V_2] = [-],$$

$$[A_1] = [-], \quad [A_2] = [-].$$

Moreover, $[\Delta|A|^{\leftrightarrow}] = [+]$ is easy to detect, because the magnitude of the acceleration in the second interval is totally greater than the magnitude of the acceleration in the first interval, and this fact can be extracted from a QSIM behavior by ordinal comparison of the endpoints of a .

We apply the mapping in theorem 3.7 by letting $(\alpha_1, \alpha_2) = (\leftarrow, \rightarrow)$ such that $f_1(t) = v_1^{\leftarrow}(t)$, $f_2(t) = v_2^{\rightarrow}(t)$ and we get:

$$[f_{b1}] = [v_{e1}] = [0], \quad [f_{b2}] = [v_{b2}] = [0] \quad (\text{theorem 3.7(ii)}),$$

$$[\Delta|F'|] = [\Delta|A|^{\leftrightarrow}] = [+]$$

$$[F_1] = [V_1] = [+], \quad [F_2] = [V_2] = [-] \quad (\text{theorem 3.7(iv)}),$$

$$[F'_1] = -[A_1] = [+], \quad [F'_2] = [A_2] = [-] \quad (\text{theorem 3.7(v)}).$$

Using these values we can apply the constraints derived in section 3, and the signs propagate as depicted in table 6.

The result is translated back to directed comparison as $[\Delta|V|^{\leftrightarrow}] = [\Delta|F|] = [+]$ (theorem 3.7(vi)). Since we also know that the distances traveled in the two intervals are the same ($[\Delta|x_{\Delta}|] = [0]$) and since the precondition of theorem 3.8 is satisfied, we can use the constraint $[\Delta|x_{\Delta}|] \subseteq [\Delta|V|^{\leftrightarrow}] + [\Delta|t_{\Delta}|]$ to deduce $[\Delta|t_{\Delta}|] = [-]$. This leads to the addition of the appropriate duration comparison fact $|t_1, t_2| < |t_0, t_1|$ to the list.

To calculate the comparison sign $[\Delta|V|^{\leftrightarrow}]$, we have used the comparison sign $[\Delta|A|^{\leftrightarrow}]$ and the sign constants $[v_{e1}]$, $[v_{b2}]$, $[V_1]$, $[V_2]$, $[A_1]$, and $[A_2]$. In general, the sign constants can be extracted easily from a QSIM behavior. A pointwise comparison sign is either calculated from a total comparison as in the case of $[\Delta|A|^{\leftrightarrow}]$, or as in the

Table 6
Derivative pointwise comparison rules firing example.

Fired rule	Reason	Result
Definition 3.5	$[f_{b1}] = [f_{b2}] = [0]$	$\Sigma[f_b] = [0]$
Theorem 3.1(ii)	$\Sigma[f_b] = [0]$	$[\Sigma f_b] = [0]$
Definition 3.5	$[F'_1] = [+]$ and $[F'_2] = [-]$	$\Delta[F'] = [-]$
Theorem 3.3(iv)	$[\Delta F'] = [+]$ and $\Delta[F'] = [-]$	$[\Sigma F'] = [-]$
Theorem 3.6(ii)	$[\Sigma f_b] = [0]$ and $[\Sigma F'] = [-]$	$[\Sigma F] = [-]$
Definition 3.5	$[F_1] = [+]$ and $[F_2] = [-]$	$\Delta[F] = [-]$
Theorem 3.3(iii)	$\Delta[F] = [-]$	$[\Delta F] = [-]$
Theorem 3.3(i)	$[\Sigma F] = [-]$ and $[\Delta F] = [-]$	$[\Delta F] = [+]$

case of $[\Delta|V|^{\leftrightarrow}]$, comparison information can propagate over the derivative relation, by application of the comparison theorems (theorems 3.1, 3.3 and 3.6) together with the mapping described in theorem 3.7. It is also possible for a pointwise comparison sign to be computed from its higher order derivatives using the above scheme in a recursive way. Moreover, [8] contains further constraints that can be used to propagate pointwise comparison over the other QSIM relations of addition, multiplication and monotonic functions.

When we use the comparison calculus in QSIM to compare durations, an important issue is how to select which intervals are going to be compared. Theoretically, one could create comparison signs for all pairs of intervals that could be formed by picking ordered (but not necessarily adjacent) pairs of time points from the partial behavior, compute unambiguous comparison signs for the pairs for which this is possible, generate all possible sign assignments for the remaining pairs, and eliminate the assignments that fail to satisfy all constraints. The duration comparison signs that have a single possible assignment remaining would be marked as the obtained results. Obviously, this algorithm has a significant time cost.

Comparing intervals at extreme points in the outward direction as in the previous example seems to be an important special case that could return duration comparison facts in many simulations (figure 15). Könik [8] describes how the kind of derivation exemplified above can be obtained as the result of a general rule-based constraint propagation scheme. Finding other useful special cases is an interesting research direction.

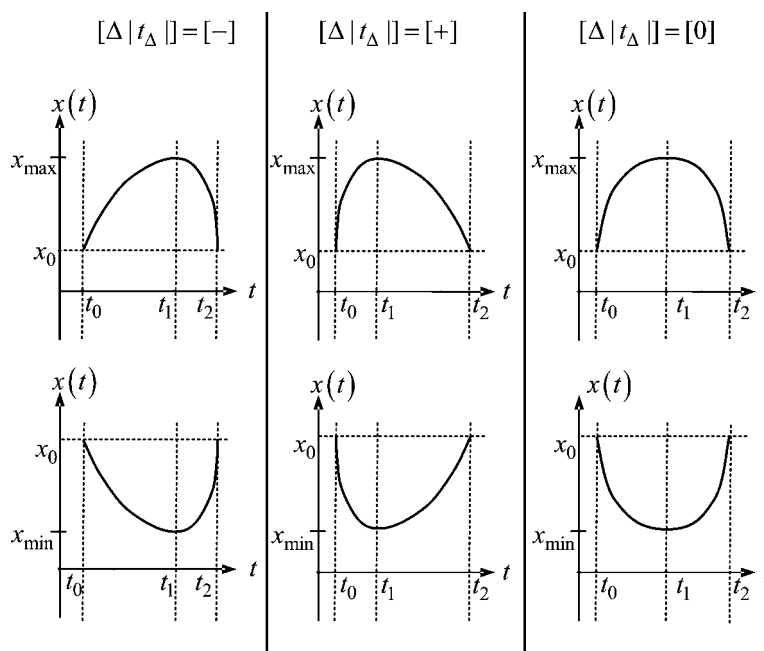


Figure 15. Other examples for comparison at change of direction.

7. Using the duration comparison facts

7.1. The duration consistency filter

The duration comparison facts accumulated as a result of the application of the methods explained in the previous section are in one of two forms: “ $|t_a, t_b| = |t_c, t_d|$ ”, or “ $|t_a, t_b| > |t_c, t_d|$ ”. Checking the consistency of a list of these facts can be viewed as a problem of the determination of the satisfiability of a set of linear inequalities as follows: Each interval in the comparison facts is rewritten as a subtraction of its endpoints, and the natural ordering between time-points occurring in the facts is reflected by additional inequalities, such that $t_i < t_j$, where $i < j$. The resulting system of inequalities is fed to a complete consistency analyzer such as [4], and the last states of behaviors seen to lead to inconsistencies are eliminated by the filter.

In our two-balls example of section 5, the duration comparison facts available during the generation of state t_2 are, once again, $|t_0, t_1| = |t_1, t_2|$ and $|t_0, t_1| > |t_1, t_2|$. The implied set of linear inequalities, namely,

$$t_1 - t_0 = t_2 - t_1, \quad t_1 - t_0 > t_2 - t_1, \quad t_0 < t_1 < t_2,$$

is easily found to be inconsistent, and figure 5 is eliminated from the output.

7.2. Richer behavior descriptions

Our modified simulator annotates the output predictions with the additional information about variables and intervals that it extracts during the computation of each behavior. Table 7 illustrates this for one of the seven surviving predictions for the two-balls system. (Once again, only the behaviors of the height variables have been shown in the table.)

Table 7
A rich prediction description for the two-balls system.

Time	h_1	h_2
t_0	0, \uparrow	0, \uparrow
(t_0, t_1)	$(0, \infty)$, \uparrow	$(0, \infty)$, \uparrow
t_1	$h_1 \text{ max}$, \odot	$(0, \infty)$, \uparrow
(t_1, t_2)	$(0, h_1 \text{ max})$, \downarrow	$(0, \infty)$, \uparrow
t_2	$(0, h_1 \text{ max})$, \downarrow	$h_2 \text{ max}$, \odot
(t_2, t_3)	$(0, h_1 \text{ max})$, \downarrow	$(0, h_2 \text{ max})$, \downarrow
$t_3 < \infty$	0, \downarrow	$(0, h_2 \text{ max})$, \downarrow

Symmetry information:

a is even symmetric everywhere.
 v_1 is odd symmetric around t_1 .
 v_2 is odd symmetric around t_2 .
 h_1 is even symmetric around t_1 .
 h_2 is even symmetric around t_2 .

Interval lengths:

$|t_0, t_1| > |t_1, t_2|$ (due to even symmetry of h_1).
 $|t_0, t_1| = |t_1, t_3|$ (due to even symmetry of h_1).
 $|t_0, t_2| > |t_2, t_3|$ (due to even symmetry of h_2).

One application of qualitative reasoning to monitoring and diagnosis of continuously running dynamic systems [6] requires qualitative simulation predictions to be matched automatically with streams of quantitative observations. The information on relative interval lengths supplied by our technique may help eliminating some additional qualitative behaviors from the tracking set, leading to increased usefulness of the overall procedure.

8. An evaluation of the duration consistency filter

8.1. Correctness

Using QSIM with the duration consistency filter is a good idea only if the following is true:

- (i) when the duration consistency filter is used, the algorithm does not predict some of the spurious behaviors that pure QSIM predicts, and
- (ii) the duration consistency filter is a *conservative* filter [10], i.e. it only eliminates provably inconsistent states.

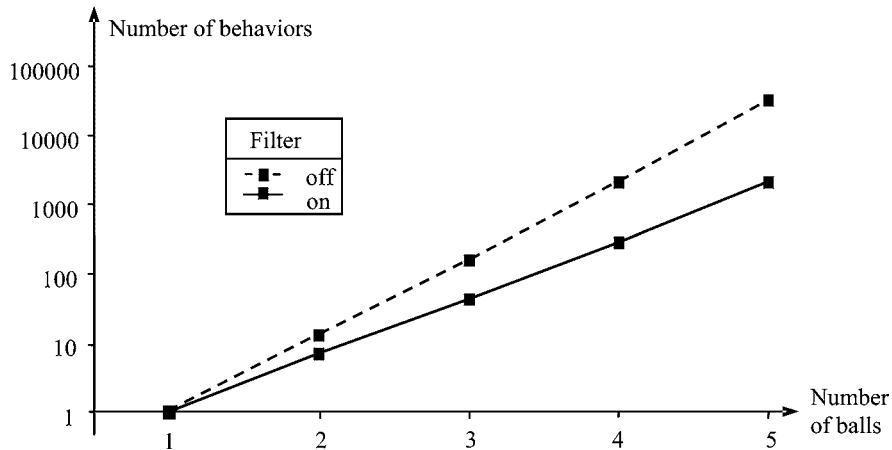
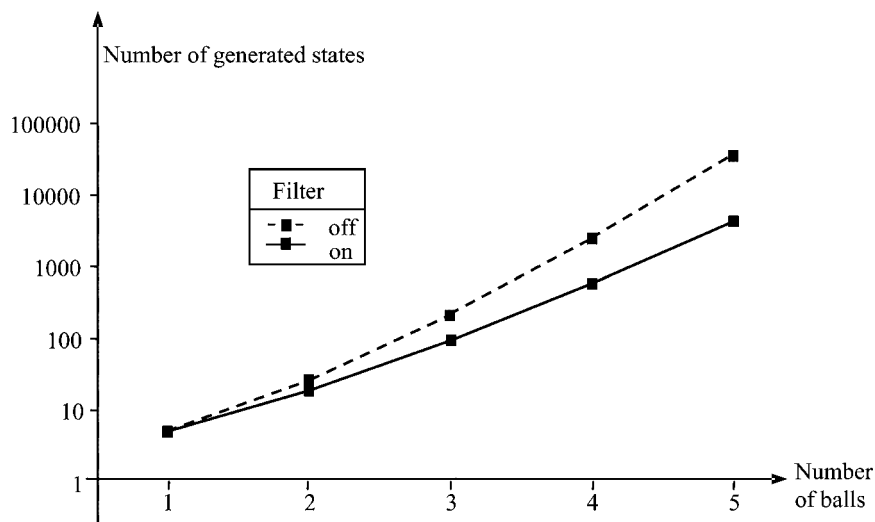
(i) has already been proven by demonstration in the previous section. As for (ii), the proofs of sections 3 and 6 show that each duration comparison fact is a mathematically justified conclusion that can be drawn from the information depicted by the corresponding model and qualitative behavior. Since the consistency checker only marks provably inconsistent combinations of such facts, the duration consistency filter is conservative.

8.2. Performance

We ran our implementations of both versions of the algorithm (with and without the duration consistency filter) on various input systems; famous examples from literature, and those of our own. No performance difference was registered when duration comparison facts cannot be extracted from the simulation output.

To measure the overheads and advantages brought by duration consistency filtering, we prepared the following experimental setup: Qualitative models of n -ball systems (where $n \in \{2, 3, 4, 5\}$) based on the model of figure 4 were simulated both with the filter turned on and off. Figures 16–18 contain the results of the experiments. Models with six or more balls produce unacceptably long execution times, regardless of whether the filter is on or off.

Figures 16 and 17 show the improvement in predictive performance brought by the application of our filter. In the 5-ball simulation, 30330 of the 32461 behaviors predicted by the previous version of the algorithm are found to be spurious! As the figures indicate, this particular family of models leads to an exponential relation between the number of balls and the number of predictions in the simulation output, both in the presence and

Figure 16. Number of predicted behaviors vs. number of balls in n -ball system.Figure 17. Number of generated states vs. number of balls in n -ball system.

absence of duration consistency filtering. The number of successfully eliminated behaviors also increases exponentially with the number of balls. Note that every eliminated state can potentially be the root of a sizable subtree of the behavior tree, and therefore each additional deletion of a spurious state is a significant contribution to the eventual utility expected from the simulation output.

The additional time requirement accrued when the filter is turned on (figure 18) is mainly due to the inefficiency of the extremely simplistic inequality analyzer employed in our implementation. The main contribution provided by our inequality extraction method is to the predictive correctness, rather than time performance, of qualitative simulation.

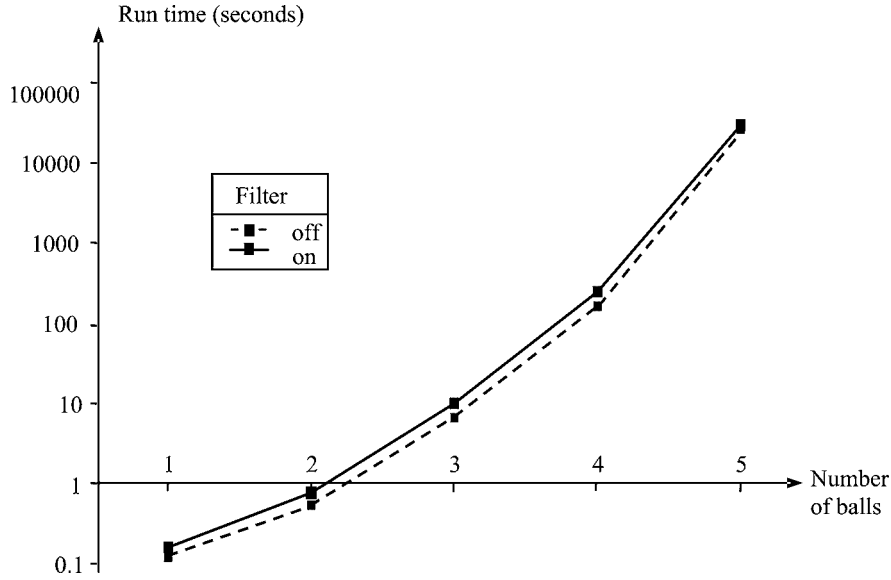


Figure 18. Simulation duration vs. number of balls in n -ball system.

9. Related work and conclusion

We have presented two new qualitative reasoning formalisms, SR1* algebra and comparison calculus, and used them in the construction of a new type of filtering mechanism, duration consistency filtering, for eliminating a class of spurious predictions from the output of qualitative simulators. Predictions of this class are identified by inconsistencies in the sets of conclusions which can be drawn about the relative lengths of the time intervals that they contain. Duration comparisons of this nature can be soundly based on several mathematical properties of the simulated functions, including symmetry and periodicity. The symmetry recognition and analysis procedure, the periodicity template processor, and the duration consistency filter itself have been implemented and tested in our Prolog version of QSIM.

A more detailed treatment of SR1* and the comparison calculus, including a general mathematical framework for analyzing the completeness of comparison calculus constraints, is presented in [8].

Weld's *differential qualitative (DQ) analysis* [16] technique involves conceptually comparing two behaviors of the same variable for purposes of perturbation analysis. DQ analysis and comparison calculus use different representations for similar concepts. In DQ analysis, the statements are defined in terms of propositional symbols, while the basic symbols in the comparison calculus are variables or sets. For example, the DQ proposition F_{\uparrow} is equivalent to the statement $[\Delta|F|^{\rightarrow}] = [+]$ and "DISTANCE-BY X_{\downarrow} " is equivalent to $[\Delta|x_{\Delta}|] = [-]$. Some theorems and concepts of DQ analysis are special instances of what we have in the comparison calculus, and some are complementary to the theorems proven here. Specifically, DQ analysis does not deal with comparison in

different directions (i.e. like $[\Delta|V|^{\leftrightarrow}]$ in figure 14) and with comparison of quantities having different signs. In [8], Könik shows how the comparison calculus can solve a perturbation analysis problem that DQ analysis cannot solve. More research is required for a possible unification of DQ analysis and comparison calculus.

Struss [14] uses symmetry arguments in phase space to infer that the spring/block system has periodic behavior.

Time interval comparison fact extraction was first implemented by Çivi [5], who presents a postprocessor which annotates QSIM outputs with deduced temporal interval comparisons using some fixed model templates. Çivi's work does not deal with spurious behaviors noticeable due to comparison information.

Some of the simulations improved by the duration consistency filter involve *occurrence branching* [15], in which multiple branches are added to the behavior tree to represent different possible time-orderings of two "unrelated" variables reaching their respective landmarks. "History"-based reasoners like Williams' TCP [18] were designed with the purpose of eliminating this phenomenon. Recently, there has been work on the model decomposition front [3] to modify the QSIM framework in this direction. The DecSIM algorithm [2] takes a user-specified partitioning of the model variables, along with the standard QSIM input items, at the start of its execution. Each partition is interpreted as a different *component* (subsystem) of the model, and simulated almost independently from the other components using QSIM. Our approach would be useful in cases where the distinctions created by the "global state"-based branching mechanisms are relevant from the user's point of view, and incorrect predictions in this format need to be minimized.

Since the introduction of "pure" QSIM in [9], several other global filters [10–13] have been added to the repertory, dealing with different classes of spurious predictions. The duration consistency filter eliminates a new class (namely, the set of predictions from which inconsistent conclusions about duration comparison facts can be drawn) which improves the predictive performance of the overall simulator, and increases the average level of complexity of the set of systems that can be reasoned about.

Hybrid qualitative–quantitative reasoners [1,7] enable the association of real-bounded intervals with the time-points and other landmarks in the qualitative simulation output. Comparing durations in this representation is a matter of utilizing this partial information through interval arithmetic. Our work shows that such comparisons are possible and useful in pure qualitative simulation as well. Since many of these hybrid simulators [10] are built around a "pure core", applying the quantitative information for pruning a behavior tree produced by a qualitative engine, our contribution here may be useful for such a system as well. A thorough comparison, and a study of the feasibility of a possible unification of the pure and hybrid methods are on our research agenda.

It might be possible to achieve symmetry propagation, as described in section 6.1, within the framework of comparison calculus. $even(f, t_i)$ on $\mathbf{I} = (t_b, t_e)$ might be described as $[\Delta F^{\leftrightarrow}] = [0]$ for the intervals $\mathbf{I}_1 = (t_b, t_i)$ and $\mathbf{I}_2 = (t_i, t_e)$, and $odd(f, t_i)$ may be described as $[\Sigma F^{\leftrightarrow}] = [0]$ on the same intervals. It is an interesting exercise to

try to obtain a set comparison calculus constraints that will cover all symmetry propagation rules.

Once we derived the comparison calculus constraints, we have obtained our results by propagation of sign values in these constraints. An alternative is to solve comparison calculus constraints by simplification. Techniques from [19], possibly with some extensions to cover SR1* algebra, could be used for that purpose.

Just as multiple traversals of the same landmark interval leads to conclusions about temporal length comparisons, duration comparison information can be used for comparing the “distances” among various landmark pairs in the same quantity space. This can, in turn, lead to the detection and elimination of a class of spurious behaviors containing inconsistencies involving landmark distances. We plan to extend the power of the filtering mechanism described here in this manner, so that qualitative simulators with even greater predictive performance and applicability can be built.

Availability of the program

The Prolog source code of our implementation of QSIM with the duration consistency filter is available to interested researchers. Contact e-mail address konik@umich.edu.

Acknowledgements

We thank the anonymous reviewers for their helpful comments. Özer Yalçın provided technical contributions in the early stages of this research. This work was partially supported by the Boğaziçi University Research Fund (Grant No. 97HA101).

References

- [1] D. Berleant and B.J. Kuipers, Qualitative and quantitative simulation: Bridging the gap, *Artificial Intelligence* 95 (1997) 215–255.
- [2] D.J. Clancy, Solving complexity and ambiguity problems within qualitative simulation, Ph.D. Thesis, Department of Computer Sciences, The University of Texas at Austin, Austin, TX (1997).
- [3] D.J. Clancy and B. J. Kuipers, Qualitative simulation as a temporally-extended constraint satisfaction problem, in: *Proc. of 15th National Conference on Artificial Intelligence* (AAAI/MIT Press, 1998).
- [4] E. Clarke and X. Zhao, Analytica: A theorem prover for Mathematica, Technical Report CMU-CS-92-117, School of Computer Science, Carnegie Mellon University, USA (1992).
- [5] H. Çivi, Duration analysis in QSIM and extension of QSIM to discrete time systems, M.S. Thesis, Boğaziçi University, İstanbul, Turkey (1992).
- [6] D.L. Dvorak, Monitoring and diagnosis of continuous dynamic systems using semiquantitative simulation, Ph.D. Thesis, Department of Computer Sciences, The University of Texas at Austin, Austin, TX (1992).
- [7] H. Kay, Refining imprecise models and their behaviors, Ph.D. Thesis, Department of Computer Sciences, The University of Texas at Austin, Austin, TX (1996).
- [8] T. Könik, Exploiting relative durations in qualitative simulation, M.S. Thesis, Boğaziçi University, İstanbul, Turkey (2000), <http://www.cmpe.boun.edu.tr/~say/theses/konik/>.

- [9] B.J. Kuipers, Qualitative simulation, *Artificial Intelligence* 29 (1986) 289–338.
- [10] B.J. Kuipers, *Qualitative Reasoning: Modeling and Simulation with Incomplete Knowledge* (MIT Press, Cambridge, MA, 1994).
- [11] A.C.C. Say, L'Hôpital's filter for QSIM, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 20 (1998) 1–8.
- [12] A.C.C. Say, Improved reasoning about infinity using qualitative simulation, *Computing and Informatics* 20 (2001) 487–507.
- [13] A.C.C. Say and S. Kuru, Improved filtering for the QSIM algorithm, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 15 (1993) 967–971.
- [14] P. Struss, Global filters for qualitative behaviors, in: *Proc. of 7th National Conference on Artificial Intelligence* (Morgan Kaufmann, San Mateo, CA, 1988) pp. 275–279.
- [15] L. Tokuda, Managing occurrence branching in qualitative simulation, in: *Proc. of 13th National Conference on Artificial Intelligence* (AAAI/MIT Press, 1996).
- [16] D.S. Weld, Comparative analysis, *Artificial Intelligence* 36 (1988) 333–373.
- [17] D.S. Weld and J.D. Kleer (eds.), *Readings in Qualitative Reasoning About Physical Systems* (Morgan Kaufmann, San Mateo, CA, 1990).
- [18] B.C. Williams, Doing time: Putting qualitative reasoning on firmer ground, in: *Proc. of 5th National Conference on Artificial Intelligence* (Morgan Kaufmann, San Mateo, CA, 1986) pp. 105–112.
- [19] B.C. Williams, A theory of interactions: Unifying qualitative and quantitative algebraic reasoning, *Artificial Intelligence* 51 (1991) 39–94.