Electronic Notes in Theoretical Computer Science 42 (2001)
URL: http://www.elsevier.nl/locate/entcs/volume42.html 22 pages

Axiomatisation of an Interval Calculus for Theorem Proving

Antonio Cerone[1]

Software Verification Research Centre, The University of Queensland, Brisbane, Australia

Abstract

We provide an axiomatisation of the Timed Interval Calculus, a set-theoretic notation for expressing properties of time intervals. We implement the axiomatisation in the Ergo theorem prover in order to allow the machine-checked proof of laws for reasoning about predicates expressed using interval operators. These laws can be then used in the machine-assisted verification of real-time applications.

1 Introduction

In recent years formalisms based on time intervals have been increasingly used for specifying real time systems. However, the complexity of proofs by hand makes their use in verification hard. There is thus the need to express such calculi in an environment that provides a reasonable automatisation for theorem proving. The only work in this direction we are aware of is an early attempt at implementing the Duration Calculus in the PVS theorem prover [13].

In our work we have defined an axiomatisation of the Timed Interval Calculus (TIC) [5], a set-theoretic notation for expressing properties of time intervals based on work by Mahony and Hayes [6]. Many useful laws for reasoning about predicates expressed in TIC have been developed [5,4,15] and used in verifying a wide range of real-time systems [2,4,15]. However, these laws need a more precise characterisation to allow their implementation in a theorem prover. Our axiomatisation gives the infrastructure for such an implementation. The actual implementation has been carried out using the Ergo theorem prover [14,1].

[1] Email: antonio@it.uq.edu.au

2 Timed Interval Calculus

The Timed Interval Calculus (TIC) is based on the notion of a *time interval*. Given a time domain $\mathbb{T}$, time intervals are represented as the set of all times between some infimum a and supremum b. For instance $(a \mathrel{...} b]$ denotes the left-open and right-closed interval between times a (exclusive) and b (inclusive). Similarly for $(a \mathrel{...} b)$, $[a \mathrel{...} b)$ and $[a \mathrel{...} b]$. The set of all time intervals is denoted by $\mathbb{I}$.

The principal specification tool of TIC consists of special brackets for defining the set of all time intervals during which a given predicate is everywhere true [5]. For instance $(\!\!-P\!-\!\!]$ is the set of all left-open and right-closed time intervals $i{\in}\mathbb{I}$ such that for each time $t{\in}i$ predicate P is true at t. Similarly for $(\!\!-P\!-\!\!)$, $[\!\!-P\!-\!\!)$ and $[\!\!-P\!-\!\!]$. TIC allows within predicate P occurrences of functions on $\mathbb{I}$ and functions on $\mathbb{T}$ that are not applied to arguments; they must be interpreted as applied respectively to the interval i defined by P and to every point t in the whole interval i [7]. However, when implementing the calculus in a theorem prover we need to introduce an explicit notation for such a *lifted* form of functions. We also define $[\![\!-P\!-\!\!) = (\!\!-P\!-\!\!) \cup [\!\!-P\!-\!\!)$. Similarly for $[\![\!-P\!-\!\!]$, $(\!\!-P\!-\!]\!]$, $[\!\!-P\!-\!]\!]$ and $[\![\!-P\!-\!]\!]$.

An important capability of TIC is an operator for connecting intervals end-to-end, to support reasoning about sequences of behaviours. The concatenation of two sets of intervals X and Y is the set $X;Y$ of all the intervals z such that there exist two disjoint intervals $x \in X$ and $y \in Y$ with supremum of x equal to infimum of y and whose union is z.

3 Time Model

The first step in mechanising an axiomatic approach to the specification of real-time systems is the axiomatisation of the *time domain*. When specifying real-time systems in TIC the time domain may be the set $\mathbb{R}$ of real numbers [2,5] or an appropriate proper subset of $\mathbb{R}$, such as the set $\mathbb{R}_0^+$ of the non-negative reals, the set $\mathbb{N}$ of natural numbers, or even an appropriate subset of the extension $\mathbb{R}_\infty$ of $\mathbb{R}$ with the $+\infty$ and $-\infty$ special values [4]. In this section we present an axiomatisation of the time domain, whose purpose is to capture the general properties that a time domain has to meet, such as having the same granularity on the whole time flow [3].

Let $\mathbb{T}$ denote the *open* time domain, 0 denote the *zero* of the time domain, $-\infty$ denote the *infimum* of the time domain, $+\infty$ denote the *supremum* of the time domain, and $\mathbb{T}_\infty$ denote the *closure* of the open time domain with the infimum and the supremum.

A first set of axioms (*type axioms*) defines properties of $\mathbb{T}$ and $\mathbb{T}_\infty$ and defines the value of *non-negative* predicate *nn* on such domains.

T1 $\mathbb{T} \subseteq \mathbb{T}_\infty$ **T2** $0 : \mathbb{T}$
T3 $nn(\mathbb{T}) \Leftrightarrow (\forall\, x : \mathbb{T}.\, (0 \leqslant x))$ **T4** $nn(\mathbb{T}_\infty) \Leftrightarrow nn(\mathbb{T})$
T5 $\forall\, x : \mathbb{T}_\infty.\, (x \in \mathbb{T} \vee x = +\infty \vee (\neg\, nn(\mathbb{T}) \Leftrightarrow x = -\infty))$

Axioms **T1** defines $\mathbb{T}_\infty$ as an extension of $\mathbb{T}$. Axioms **T2** defines a special element (0) of $\mathbb{T}$. **T3** and **T4** define predicate *nn* which is true on $\mathbb{T}$ and $\mathbb{T}_\infty$ if the time domain does not contain negative time points. Axiom **T5** introduces $+\infty$ as an element of $\mathbb{T}_\infty$; it also introduces $-\infty$ as an element of $\mathbb{T}_\infty$ if $\mathbb{T}$ contains at least one negative time; moreover it ensures that there are no other elements of $\mathbb{T}_\infty$ which are not in $\mathbb{T}$.

Axioms (**T6**–**T11**) characterise $\leqslant$ as a partial order and $<$ as the strict order. The special values $-\infty$ and $+\infty$ are respectively defined as the bottom and the top of the partial order in $\mathbb{T}_\infty$.

T6 $\forall\, x, y : \mathbb{T}_\infty.\, (x \leqslant y \vee y \leqslant x)$
T7 $\forall\, x, y : \mathbb{T}_\infty.\, (x \leqslant y \wedge y \leqslant z \Rightarrow x \leqslant z)$
T8 $\forall\, x, y : \mathbb{T}_\infty.\, (x \leqslant y \wedge y \leqslant x \Rightarrow x = y)$
T9 $\forall\, x, y : \mathbb{T}_\infty.\, (x < y \Leftrightarrow x \neq y \wedge x \leqslant y)$
T10 $\forall\, x : \mathbb{T}.\, (x < +\infty)$
T11 $\neg\, nn(\mathbb{T}_\infty) \Rightarrow \forall\, x.\, \mathbb{T}.\, (-\infty < x)$
T12 $\exists\, x : \mathbb{T}.\, (0 < x)$

Axiom **T6** ensures that $\leqslant$ is defined for every pair of elements of $\mathbb{T}_\infty$. Axiom **T7** ensures that $\leqslant$ is transitive. Axiom **T8** ensures that $\leqslant$ is antisymmetric. Axiom **T9** defines $<$ in terms of $\leqslant$. From Axioms **T7**, **T8** and **T9** we can derive the transitivity of $<$. Axiom **T10** characterises $+\infty$ as an upper bound for $\mathbb{T}_\infty$. Axiom **T11** characterises $+\infty$ as a lower bound for $\mathbb{T}_\infty$. From Axioms **T5**, **T10** and **T11** we can derive that $+\infty$ is the least upper bound (supremum) of $\mathbb{T}_\infty$ and that $-\infty$ is the geatest lower bound (infimum) of $\mathbb{T}_\infty$. Axiom **T12** ensures that there exists at least a positive time.

Notice that reflexivity of $\leqslant$ is an obvious consequence of Axiom **T6**. Moreover, $+\infty \notin \mathbb{T}$ can be derived from **T9** and **T10**, while $-\infty \notin \mathbb{T}$ can be derived from **T9** and **T11**.

The $+$ addition operator is defined for time points by the following axioms.

T13 $\forall\, x, y : \mathbb{T}_\infty.\, ((x + y) : \mathbb{T}_\infty \Leftrightarrow x : \mathbb{T} \vee y : \mathbb{T} \vee x = y)$
T14 $\forall\, x : \mathbb{T}_\infty.\, (x + 0 = x)$
T15 $\neg\, nn(\mathbb{T}) \Rightarrow \forall\, x : \mathbb{T}.\, \exists\, y : \mathbb{T}.\, (x + y = 0)$
T16 $\forall\, x, y : \mathbb{T}_\infty.\, (x + y = y + x)$
T17 $\forall\, x, y, z : \mathbb{T}_\infty.\, (x + (y + z) = (x + y) + z)$
T18 $\forall\, x : \mathbb{T}_\infty.\, (x < +\infty \Rightarrow x + (-\infty) = -\infty)$
T19 $\forall\, x : \mathbb{T}_\infty.\, (-\infty < x \Rightarrow x + (+\infty) = +\infty)$

Axiom **T13** deserves a special remark. It ensures that $+$ is defined between finite time points or identical infinite time points. This excludes the sum between $-\infty$ and $+\infty$, which would lead to an indefinite result. Axioms **T14**–**T17** define 0 as the identity for $+$, the existence of the opposite for a time domain that includes negative time points and the properties of commutivity and associativity for $+$. Axioms **T18** and **T19** consider the special cases when at least one of the arguments of $+$ is infinite.

Some axioms relate the partial order relation and the sum function.

T20 $\forall\, x, y : \mathbb{T}.\, (x + y = 0 \Rightarrow (x \leqslant 0 \wedge 0 \leqslant y) \vee (y \leqslant 0 \wedge 0 \leqslant x))$
T21 $\forall\, x, y : \mathbb{T}.\, (0 < y \Rightarrow x < x + y)$
T22 $\forall\, x, y : \mathbb{T}.\, (y < 0 \Rightarrow x + y < x)$

Axiom **T20** ensures that if the sum of two time points is 0, then they cannot be both positive or both negative. Axioms **T21** and **T22** ensure that a finite time is increased by adding a positive finite time and decreased by adding a finite negative time.

Notice that Axioms **T9**, **T12** and **T21** together with the transitivity of $<$ ensure that there exists an infinite number of positive time points. Analogously, **T9** and **T22** together with the transitivity of $<$ ensure that if there exists a negative time point, then there exists an infinite number of negative time points.

The last set of axioms define the time distance or duration d between time points.

T23 $\forall\, x, y : \mathbb{T}_\infty.\, (d(x, y) : \mathbb{T}_\infty \Leftrightarrow$
$\qquad (x = +\infty \Rightarrow y < +\infty) \wedge (x = -\infty \Rightarrow -\infty < y))$
T24 $\forall\, x, y : \mathbb{T}_\infty.\, (d(x, y) : \mathbb{T}_\infty \Rightarrow d(x, y) = d(y, x))$
T25 $\forall\, x, y, z : \mathbb{T}.\, (d(x, y) = d(x + z, y + z))$
T26 $\forall\, x : \mathbb{T}.\, \forall\, y : \mathbb{T}.\, (x \leqslant y \Leftrightarrow x + d(x, y) = y)$
T27 $\forall\, x : \mathbb{T}_\infty.\, (d(x, +\infty) : \mathbb{T}_\infty \Rightarrow d(x, +\infty) = +\infty)$
T28 $\forall\, x : \mathbb{T}_\infty.\, (d(x, -\infty) : \mathbb{T}_\infty \Rightarrow d(x, -\infty) = +\infty)$

Axiom **T23** defines d between finite time points or non-identical infinite time points. This excludes $d(-\infty, -\infty)$ and $d(+\infty, +\infty)$, which would lead to an indefinite result. **T24** is the commutative property. **T25** ensures that the granularity of time is the same on the whole time flow [3]. **T26** relates sum and distance on finite time points, whereas **T27** and **T28** define the special cases where at least one of the arguments of distance is an infinite time. It is straightforward to prove that d meets the properties of a distance on a metric space.

4 Time Intervals and Sets of Intervals

The first set of axioms defines the set $\mathbb{I}$ of all well-formed time intervals.

I1 $\mathbb{I} \subseteq \mathbb{U}$ **I2** $\forall\, i : \mathbb{I}.\; i \subseteq \mathbb{T}$ **I3** $\mathbb{T} : \mathbb{I}$ **I4** $\neg\, \varnothing : \mathbb{I}$

Axiom **I1** characterises the set of all time intervals $\mathbb{I}$ as a subtype of the class of all sets $\mathbb{U}$, while Axiom **I2** ensures that every time interval is a subset of the time domain. According to Axiom **I3** the whole time domain $\mathbb{T}$ is seen as a time interval, that is, as an element of $\mathbb{I}$. In other words $\mathbb{T}$ has type $\mathbb{I}$. **I4** prevents $\varnothing$ from been considered as a time interval because it has no *position* in the time domain.

A second set of axioms defines functions $\alpha : \mathbb{I} \to \mathbb{T}$ and $\omega : \mathbb{I} \to \mathbb{T}$, which give respectively the left and right endpoints of an interval, and the predicates *lcl* and *rcl*, which are true if the interval is respectively left-closed and right-closed and are false otherwise.

I5 $\forall\, i : \mathbb{I}.\; \forall\, x : \mathbb{T}.\; (\alpha(i) < x \wedge x < \omega(i) \Rightarrow x \in i)$
I6 $\forall\, i : \mathbb{I}.\; (\alpha(i) : \mathbb{T}_\infty \wedge (lcl(i) \Leftrightarrow \alpha(i) \in i))$
I7 $\forall\, i : \mathbb{I}.\; (\omega(i) : \mathbb{T}_\infty \wedge (rcl(i) \Leftrightarrow \omega(i) \in i))$

Axiom **I5** ensures that all the time points that are greater than the left endpoint and smaller than the right endpoint belong to the interval. **I6** ensures that an interval is left-closed iff the left endpoint belongs to the interval. **I7** is analogous for right endpoints of right-closed intervals.

The next set of axioms define the adjacent predicate *adj* (Axiom **I8**) and the concatenation function $\bullet$ (Axiom **I9**).

I8 $\forall\, i, j : \mathbb{I}.\; (adj(i,j) \Leftrightarrow (\omega(i) = \alpha(j) \wedge i \cap j = \varnothing))$
I9 $\forall\, i, j : \mathbb{I}.\; (adj(i,j) \Leftrightarrow i \bullet j = i \cup j)$

Axiom **I8** ensures that intervals i and j are adjacent if the right endpoint of i is the left endpoint of j, but i and j are disjoint. **I9** defines the concatenation of adjacent intervals as their union.

A special function $\phi : \mathbb{I} \to \mathbb{I}$ denotes the identity function on $\mathbb{I}$. It is defined by the following axiom.

I10 $\forall\, i : \mathbb{I}.\; \phi(i) = i$

Let $\mathbb{S}$ denote the the set of all possible interval sets and $\overline{\mathbb{S}}$ denote the extension of $\mathbb{S}$ with the empty set.

S1 $\forall\, I.\; (I : \mathbb{S} \Leftrightarrow \forall\, i.\; ((i \in I) \Rightarrow (i : \mathbb{I})))$
S2 $\forall\, I : \mathbb{S}.\; (I \cup \{\varnothing\}) : \overline{\mathbb{S}})$
S3 $\mathbb{S} \subseteq \overline{\mathbb{S}}$
S4 $\overline{\mathbb{S}} \subseteq \mathbb{U}$

While the empty set is not a time interval, there are situations when it is useful to reason with it [15].

The most interesting operation between interval sets is their *concatenation*.

S5 $\forall\, I, J : \mathbb{S}.\ (I; J = \{k \mid \exists\, i, j.\ i \in I \wedge j \in J \wedge (k = i{\bullet}j)\})$

The concatenation of two interval sets I and J is the set of all possible concatenations of an interval $i \in I$ with an interval $j \in J$. If there are no such intervals, then $I; J = \varnothing$. The concatenation can be extended to elements of $\overline{\mathbb{S}}$ through the following axioms.

S6 $\forall\, I : \mathbb{S}.\ \forall\, J : \overline{\mathbb{S}}.\ ((I \cup \{\varnothing\}); J = I; J \cup J)$
S7 $\forall\, I : \overline{\mathbb{S}}.\ \forall\, J : \mathbb{S}.\ (I; (J \cup \{\varnothing\}) = I; J \cup I)$

It is straightforward to verify that $\{\varnothing\}$ is the unit of concatenation on $\overline{\mathbb{S}}$.

5 Lifting Predicates

The central feature of TIC is the use of functions from the time domain ($\mathbb{T}$) and from the time interval domain ($\mathbb{I}$) to model the dynamic behaviour of observable system properties. In order to elide most explicit references to these two domains, functions may be used in a *lifted* form within predicates [6]. However, to avoid ambiguities in the theorem prover, these uses must be made explicit.

A total function $v : \mathbb{T} \to V$ from the time domain to V can be lifted within a predicate as $v^{\uparrow}$. Analogously for a total function from the interval time domain. For instance $\alpha : \mathbb{I} \to \mathbb{T}$ can be lifted as $\alpha^{\uparrow}$. Let $P(v^{\uparrow}, \alpha^{\uparrow})$ be a predicate expression containing $v^{\uparrow}$ and $\alpha^{\uparrow}$ and no other lifted form, $\downarrow_{\mathbb{T}}$ be a functional infix operator having a predicate as left argument and a time as right argument and giving a predicate as a result, $\downarrow_{\mathbb{I}}$ be a functional infix operator having a predicate as left argument and an interval as right argument and giving a predicate as a result. Then P may be instantiated with time t using substitution as follows.

$$P{\downarrow_{\mathbb{T}}}t = P[v(t)/v^{\uparrow}]$$

Analogously $P(v^{\uparrow}, \alpha^{\uparrow})$ may be instantiated with interval i using substitution as follows.

$$P{\downarrow_{\mathbb{I}}}i = P[\alpha(i)/\alpha^{\uparrow}]$$

Functions $\downarrow_{\mathbb{T}}$ and $\downarrow_{\mathbb{I}}$ are axiomatised by giving the following axioms for v and $v^{\uparrow}$ and for α and $\alpha^{\uparrow}$.

ATv $v^{\uparrow}{\downarrow_{\mathbb{T}}}t = v(t)$
ATα $\alpha^{\uparrow}{\downarrow_{\mathbb{T}}}t = \alpha^{\uparrow}$
ATω $\omega^{\uparrow}{\downarrow_{\mathbb{T}}}t = \omega^{\uparrow}$

AIv $v^{\uparrow}{\downarrow_{\mathbb{I}}}i = v^{\uparrow}$
AIα $\alpha^{\uparrow}{\downarrow_{\mathbb{I}}}i = \alpha(i)$
AIω $\omega^{\uparrow}{\downarrow_{\mathbb{I}}}i = \omega(i)$

Analogous axioms must be given for all the functions from time and from time intervals that need to be used in a lifted form. Moreover, axioms are needed to define how to propagate the instantiation through the structure of the predicate. Constants are not affected by instantiation, whereas instantiation is propagated through the arguments of (non lifted) functions and functional, relational and logical operators. For example, the following axioms are given for the propagation of the $\downarrow_{\mathbb{T}}$ operator through any constant c, functions v and α, operators $+$, $<$ $=$, $\in$, $\wedge$ and $\Rightarrow$ and quantifier $\forall$.

LTc $c\downarrow_{\mathbb{T}}t = c$, for any constant c of any type

LTv $v(\bar{t})\downarrow_{\mathbb{T}}t = v(\bar{t}\downarrow_{\mathbb{T}}t)$

LTα $\alpha(\bar{\imath})\downarrow_{\mathbb{T}}t = \alpha(\bar{\imath}\downarrow_{\mathbb{T}}t)$

LT$+$ $(a+b)\downarrow_{\mathbb{T}}t = ((a\downarrow_{\mathbb{T}}t) + (b\downarrow_{\mathbb{T}}t))$

LT$<$ $(a<b)\downarrow_{\mathbb{T}}t = ((a\downarrow_{\mathbb{T}}t) < (b\downarrow_{\mathbb{T}}t)))$

LT$=$ $(a=b)\downarrow_{\mathbb{T}}t = ((a\downarrow_{\mathbb{T}}t) = (b\downarrow_{\mathbb{T}}t))$

LT$\in$ $(a \in A)\downarrow_{\mathbb{T}}t = ((a\downarrow_{\mathbb{T}}t) \in (A\downarrow_{\mathbb{T}}t))$

LT$\wedge$ $(P \wedge Q)\downarrow_{\mathbb{T}}t = ((P\downarrow_{\mathbb{T}}t) \wedge (Q\downarrow_{\mathbb{T}}t))$

LT$\Rightarrow$ $(P \Rightarrow Q)\downarrow_{\mathbb{T}}t = ((P\downarrow_{\mathbb{T}}t) \Rightarrow (Q\downarrow_{\mathbb{T}}t))$

LI$\forall$ $(\forall\, x.\, P)\downarrow_{\mathbb{T}}t = \forall\, x.\, (P\downarrow_{\mathbb{T}}t)$, if x is not free in t

Analogously, the following axioms are given for the propagation of the $\downarrow_{\mathbb{I}}$ operator through any constant c, functions v and α, operators $+$, $<$ $=$, $\in$, $\wedge$ and $\Rightarrow$ and quantifier $\forall$.

LIc $c\downarrow_{\mathbb{I}}i = c$, for any constant c of any type

LIv $v(t)\downarrow_{\mathbb{I}}i = v(t\downarrow_{\mathbb{I}}i)$

LIα $\alpha(\bar{\imath})\downarrow_{\mathbb{I}}i = \alpha(\bar{\imath}\downarrow_{\mathbb{I}}i)$

LI$+$ $(a+b)\downarrow_{\mathbb{I}}i = (a\downarrow_{\mathbb{I}}i) + (b\downarrow_{\mathbb{I}}i)$

LI$<$ $(a<b)\downarrow_{\mathbb{I}}i = (a\downarrow_{\mathbb{I}}i) < (b\downarrow_{\mathbb{I}}i)$

LI$=$ $(a=b)\downarrow_{\mathbb{I}}i = (a\downarrow_{\mathbb{I}}i) = (b\downarrow_{\mathbb{I}}i)$

LI$\in$ $(a \in A)\downarrow_{\mathbb{I}}i = (a\downarrow_{\mathbb{I}}i) \in (A\downarrow_{\mathbb{I}}i)$

LI$\wedge$ $(P \wedge Q)\downarrow_{\mathbb{I}}i = (P\downarrow_{\mathbb{I}}i) \wedge (Q\downarrow_{\mathbb{I}}i)$

LI$\Rightarrow$ $(P \Rightarrow Q)\downarrow_{\mathbb{I}}i = (P\downarrow_{\mathbb{I}}i) \Rightarrow (Q\downarrow_{\mathbb{I}}i)$

LI$\forall$ $(\forall\, x.\, P)\downarrow_{\mathbb{I}}i = \forall\, x.\, (P\downarrow_{\mathbb{I}}i)$, if x is not free in i

Notice that **LT**-axioms and **LI**-axioms must be given for all function, including the ones that are not used in lifted form.

Then $P(v^{\uparrow}, \alpha^{\uparrow})$ may be instantiated with both an interval i and a time t using substitution as follows.

$$P\downarrow(i,t) = P[\alpha(i)/\alpha^{\uparrow}, v(t)/v^{\uparrow}]$$

The $\downarrow$ functional infix operator having a predicate as left argument and a pair consisting of an interval and a time as right argument and giving a predicate as a result can be defined as the composition of $\downarrow_{\mathbb{I}}$ and $\downarrow_{\mathbb{T}}$.

LD$\downarrow$ $P\downarrow(i,t) = (P\downarrow_{\mathbb{I}} i)\downarrow_{\mathbb{T}} t$

It is easy to prove the following theorems

LDC $(P\downarrow_{\mathbb{I}} i)\downarrow_{\mathbb{T}} t = (P\downarrow_{\mathbb{T}} t)\downarrow_{\mathbb{I}} i$
LDT $(P\downarrow_{\mathbb{T}} t)\downarrow_{\mathbb{T}} t' = P\downarrow_{\mathbb{T}} t$
LDI $(P\downarrow_{\mathbb{I}} i)\downarrow_{\mathbb{I}} i' = P\downarrow_{\mathbb{I}} i$

Each of the following theorems can be derived from **LD**$\downarrow$ and every pair of the corresponding **AT**-axioms and **AI**-axioms.

Av $v^{\uparrow}\downarrow(i,t) = v(t)$
Aα $\alpha^{\uparrow}\downarrow(i,t) = \alpha(i)$
Aω $\omega^{\uparrow}\downarrow_{\mathbb{I}}(i,t) = \omega(i)$

Each of the following theorems can be derived from **LD**$\downarrow$ and every pair of the corresponding **LT**-axioms and **LI**-axioms.

Lc $c\downarrow(i,t) = c$, for any constant c of any type
Lv $v(\bar{t})\downarrow(i,t) = v(\bar{t}\downarrow(i,t))$
Lα $\alpha(\bar{\imath})\downarrow(i,t) = \alpha(\bar{\imath}\downarrow(i,t))$
L$+$ $(a+b)\downarrow(i,t) = (a\downarrow(i,t)) + (b\downarrow(i,t))$
L$<$ $(a<b)\downarrow(i,t) = (a\downarrow(i,t)) < (b\downarrow(i,t))$
L$=$ $(a=b)\downarrow(i,t) = (a\downarrow(i,t)) = (b\downarrow(i,t))$
L$\in$ $(a\in A)\downarrow(i,t) = (a\downarrow(i,t)) \in (A\downarrow(i,t))$
L$\wedge$ $(P\wedge Q)\downarrow(i,t) = (P\downarrow(i,t)) \wedge (Q\downarrow(i,t))$
L$\Rightarrow$ $(P\Rightarrow Q)\downarrow(i,t) = (P\downarrow(i,t)) \Rightarrow (Q\downarrow(i,t))$
L$\forall$ $(\forall\, x.\, P)\downarrow(i,t) = \forall\, x.\, (P\downarrow(i,t))$, if x is not free in i or in t

It is interesting to notice that the ϕ identity function on $\mathbb{I}$ characterises in its lifted form $\phi^{\uparrow}$ a sort of *current interval*, which is the interval whose properties are stated by the predicate in which $\phi^{\uparrow}$ occurs.

$$\phi^{\uparrow}\downarrow(i,t) = \phi(i) = i$$

We have now the infrastructure necessary to axiomatically define the special brackets informally defined in Section 2. The set $\llbracket\!-P-\!\rrbracket$ of all time intervals $i \in \mathbb{I}$ such that for each time $t \in I$ predicate P is true at t is defined by the following axiom.

Puu $\llbracket\!-P-\!\rrbracket = \{i : \mathbb{I} \mid \forall\, t : \mathbb{T}.\, (t \in i \Rightarrow (P\downarrow(i,t)))\}$

The other special brackets can be defined in term of $\llbracket \_ \rrbracket$

$$\begin{array}{llll}
\mathbf{Pcu} & [P\rrbracket = \llbracket P \wedge lcl^{\uparrow}\rrbracket & \mathbf{Puc} & \llbracket P] = \llbracket P \wedge rcl^{\uparrow}\rrbracket \\
\mathbf{Pou} & [P\rrbracket = \llbracket P \wedge \neg\, lcl^{\uparrow}\rrbracket & \mathbf{Puo} & \llbracket P] = \llbracket P \wedge \neg\, lcl^{\uparrow}\rrbracket \\
\mathbf{Pcc} & [P] = [P \wedge rcl^{\uparrow}\rrbracket & \mathbf{Pco} & [P) = [P \wedge \neg\, rcl^{\uparrow}\rrbracket \\
\mathbf{Poc} & (P] = (P \wedge rcl^{\uparrow}\rrbracket & \mathbf{Poo} & (P) = (P \wedge \neg\, rcl^{\uparrow}\rrbracket
\end{array}$$

Notice that we have used the lifted forms of the *lcl* and *rcl* predicate operators within the special brackets. In order to allow nestings of special brackets we need the following axiom.

$$\mathbf{Pit}\ \ \llbracket P\rrbracket{\downarrow}(i,t) = \llbracket P\rrbracket$$

Therefore, in case of nesting of special brackets **Puu** and **Pit** can be combined as follows. Let $P(\llbracket Q\rrbracket)$ be a predicate that uses the set of intervals $\llbracket Q\rrbracket$.

$$\llbracket P(\llbracket Q\rrbracket)\rrbracket = \{i : \mathbb{I} \mid \forall\, t : \mathbb{T}.\ (t \in i \Rightarrow (P{\downarrow}(i,t))(\llbracket Q\rrbracket))\}$$

We can prove this as follows.

$$\begin{array}{ll}
\llbracket P(\llbracket Q\rrbracket)\rrbracket & \\
\quad = \{i : \mathbb{I} \mid \forall\, t : \mathbb{T}.\ (t \in i \Rightarrow (P(\llbracket Q\rrbracket)){\downarrow}(i,t))\} & \mathbf{Puu} \\
\quad = \{i : \mathbb{I} \mid \forall\, t : \mathbb{T}.\ (t \in i \Rightarrow ((P{\downarrow}(i,t))(\llbracket Q\rrbracket{\downarrow}(i,t)))\} & \\
\qquad\qquad \mathbf{L}\text{-axioms depending on the structure of } P & \\
\quad = \{i : \mathbb{I} \mid \forall\, t : \mathbb{T}.\ (t \in i \Rightarrow (P{\downarrow}(i,t))(\llbracket Q\rrbracket)\} & \mathbf{Pit}
\end{array}$$

For instance

$$\llbracket [0 \ldots \alpha^{\uparrow}] \in \llbracket \alpha^{\uparrow} = \omega^{\uparrow}\rrbracket\rrbracket = \llbracket \alpha^{\uparrow} = 0\rrbracket\}$$

because

- $\llbracket \alpha^{\uparrow} = \omega^{\uparrow}\rrbracket$ is the set of all intervals consisting of just one point in time;
- any interval $[0 \ldots t']$, with $t' \in \mathbb{T}$, belongs to such a set if and only if $t' = 0$;
- therefore the outermost special brackets define all intervals with left endpoint equal to 0.

This can be formally proved by applying the given axioms.

$$\begin{array}{ll}
\llbracket [0 \ldots \alpha^{\uparrow}] \in \llbracket \alpha^{\uparrow} = \omega^{\uparrow}\rrbracket\rrbracket & \\
\quad = \{i : \mathbb{I} \mid \forall\, t : \mathbb{T}.\ (t \in i \Rightarrow ([0 \ldots \alpha^{\uparrow}] \in \llbracket \alpha^{\uparrow} = \omega^{\uparrow}\rrbracket{\downarrow}(i,t)))\} & \mathbf{Puu} \\
\quad = \{i : \mathbb{I} \mid \forall\, t : \mathbb{T}.\ (t \in i \Rightarrow & \\
\qquad\qquad (([0 \ldots \alpha^{\uparrow}]{\downarrow}(i,t)) \in (\llbracket \alpha^{\uparrow} = \omega^{\uparrow}\rrbracket{\downarrow}(i,t))))\} & \mathbf{L}{\in} \\
\quad = \{i : \mathbb{I} \mid \forall\, t : \mathbb{T}.\ (t \in i \Rightarrow (([0 \ldots \alpha^{\uparrow}]{\downarrow}(i,t)) \in \llbracket \alpha^{\uparrow} = \omega^{\uparrow}\rrbracket))\} & \mathbf{Pit} \\
\quad = \{i : \mathbb{I} \mid \forall\, t : \mathbb{T}.\ (t \in i \Rightarrow ([0 \ldots \alpha(i)] \in \llbracket \alpha^{\uparrow} = \omega^{\uparrow}\rrbracket))\} & \mathbf{A}\alpha \\
\quad = \{i : \mathbb{I} \mid \forall\, t : \mathbb{T}.\ (t \in i \Rightarrow &
\end{array}$$

$$
\begin{aligned}
&\qquad ([0 \mathinner{\ldotp\ldotp\ldotp} \alpha(i)] \in \{i' : \mathbb{I} \mid \forall\, t' : \mathbb{T}.\, (t' \in i' \Rightarrow \\
&\qquad\qquad\qquad (\alpha^{\uparrow} = \omega^{\uparrow}){\downarrow}(i', t'))\}))\} && \mathbf{A}\alpha \\
&= \{i : \mathbb{I} \mid \forall\, t : \mathbb{T}.\, (t \in i \Rightarrow \\
&\qquad ([0 \mathinner{\ldotp\ldotp\ldotp} \alpha(i)] \in \{i' : \mathbb{I} \mid \forall\, t' : \mathbb{T}.\, (t' \in i' \Rightarrow \\
&\qquad\qquad\qquad (\alpha^{\uparrow}{\downarrow}(i', t') = \omega^{\uparrow}{\downarrow}(i', t')))\}))\} && \mathbf{L}{=} \\
&= \{i : \mathbb{I} \mid \forall\, t : \mathbb{T}.\, (t \in i \Rightarrow \\
&\qquad ([0 \mathinner{\ldotp\ldotp\ldotp} \alpha(i)] \in \{i' : \mathbb{I} \mid \forall\, t' : \mathbb{T}.\, (t' \in i' \Rightarrow \\
&\qquad\qquad\qquad (\alpha(i') = \omega(i')))\}))\} && \mathbf{A}\alpha \text{ and } \mathbf{A}\omega \\
&= \{i : \mathbb{I} \mid \forall\, t : \mathbb{T}.\, (t \in i \Rightarrow ([0 \mathinner{\ldotp\ldotp\ldotp} \alpha(i)] \in \{[t' \mathinner{\ldotp\ldotp\ldotp} t'] \mid t' : \mathbb{T}\}))\} \\
&= \{i : \mathbb{I} \mid \forall\, t : \mathbb{T}.\, (t \in i \Rightarrow \alpha(i) = 0)\} \\
&= \{i : \mathbb{I} \mid \alpha(i) = 0\} \\
&= \{i : \mathbb{I} \mid \forall\, t : \mathbb{T}.\, t \in i \Rightarrow \alpha(i) = 0)\} \\
&= \{i : \mathbb{I} \mid \forall\, t : \mathbb{T}.\, (t \in i \Rightarrow \alpha^{\uparrow}{\downarrow}(i, t) = 0)\} && \mathbf{A}\alpha \\
&= \llbracket \alpha^{\uparrow} = 0 \rrbracket && \mathbf{Puu}
\end{aligned}
$$

The meaning of lifted predicates within the special brackets is now clear. However we need to define the meaning of predicates at the top level. We introduce other special brackets, $\langle\!| \_ |\!\rangle$ to enclose the top level predicate.

$$
\begin{aligned}
&\mathbf{Ptt}\ \ \langle\!| P |\!\rangle = \forall\, i : \mathbb{I}.\ \forall\, t : \mathbb{T}.\, (t \in i \Rightarrow (P{\downarrow}(i, t))), \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{with } i, t \text{ not free in } P
\end{aligned}
$$

When writing a specification on paper we can assume any top level predicate is implicitly enclosed between $\langle\!|$ and $|\!\rangle$. However, in a theorem prover we need to explicitly indicate $\langle\!|$ and $|\!\rangle$. Anyway $\langle\!|$ and $|\!\rangle$ may also appear in a subformula rather than just at the top level. We will see such an example in Section 7.

Since TIC is an interval calculus, lifting a function defined on intervals is more critical than lifting a function defined just on times. When performing proofs we would like to extend properties of intervals to subintervals. However, if the interval is defined through the special brackets from a predicate P that contains lifted forms of functions defined on intervals, P might not be true on the subinterval we are interested in. In order to extend properties to subintervals, which is a very powerful proving mechanism, we introduce another form of special bracket that instantiates all lifted forms of functions defined on intervals.

$$
\mathbf{Pii}\ \ \langle\!\langle P \rangle\!\rangle = \forall\, i : \mathbb{I}.\, (P{\downarrow_{\mathbb{I}}} i), \qquad \text{with } i \text{ not free in } P
$$

6 Implementation in Ergo

Ergo is a term rewriting theorem prover designed and implemented at the Software Verification Research Centre at the University of Queensland. It is based on the *window inference* [11,8] proof paradigm.

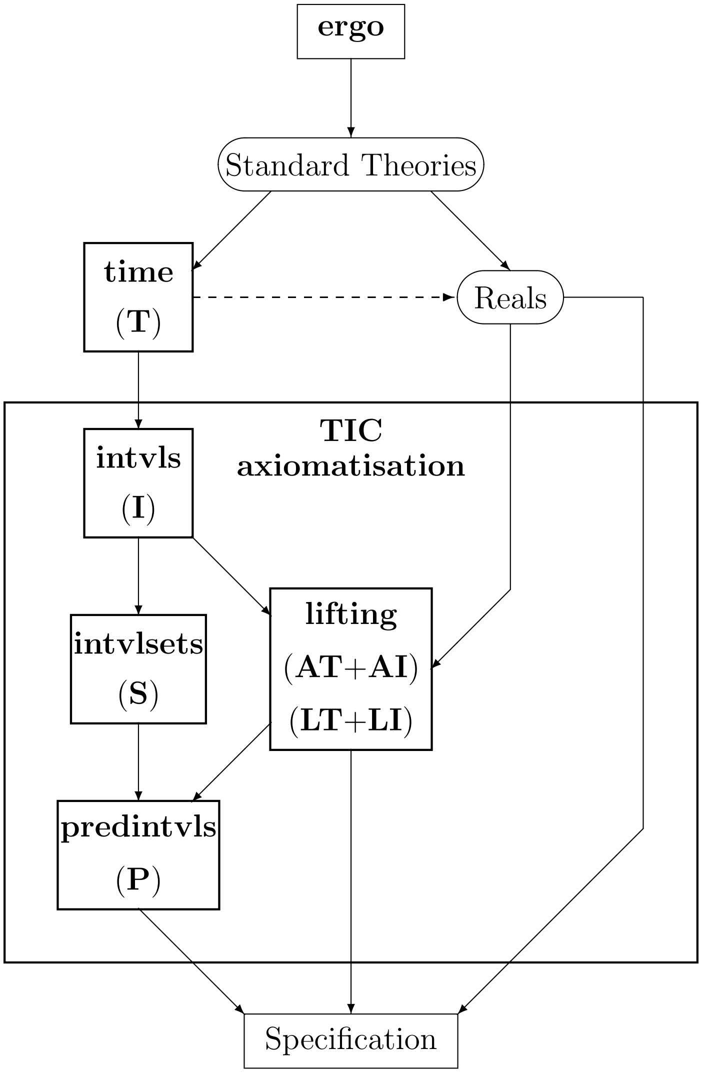


Fig. 1. Theory graph for TIC

In this work we use Ergo 4 [14,1]. The architecture of Ergo 4 consists of the *proof engine*, which is the core of Ergo, the *theory database*, which is Ergo's repository of information containing object logics, and the *tactics* that implement the command-line interface and a higher level Ergo-Emacs interface [9]. Tactics are user-writable and support the construction of the theory database and control the proof engine. They are written using the Qu-Prolog language [10], a high level extension of Prolog that includes features such as user defined quantifiers.

An Ergo theory is a collection of declarations, axioms, tactics and heuristics. Theories can *inherit* other theories, thus forming a *theory graph*. The theory graph for TIC is sketched in Figure 1. An oval represents a subgraph of theories while a rectangle represents a single theory that is a node in the theory graph. The theories in thick boxes have been explicitly added to the theory database to implement TIC. Under the names of such theories we have indicated the corresponding group of axioms introduced in the previous sections.

The **ergo** theory is the base theory for the theorem prover. The “Standard Theories” box is a subgraph of theories for arithmetic, predicate calculus, equality, types, sets, etc. The “Reals” box is a subgraph of theories for axiomatising the real numbers [12]. The dashed arrow between the **time** theory and the theories in box “Reals” means that the **time** abstract theory is interpreted into the “Reals” current theories. However, the **time** theory could also be interpreted into the Naturals or into any other consistent domain. The SPEC theory is the specific application that is modelled in TIC. The **lifting** theory is not a general one; it contains some general axioms such as **AT**α defined in Section 5, but also axioms specific to the application. For example, **ATv** is in **lifting** only if function v is used in SPEC.

We now analyse some details of the Ergo implementation. Fixed values such as 0, $+\infty$, $-\infty$, etc., lifted forms such as $v^{\uparrow}$, $\alpha^{\uparrow}$, $\omega^{\uparrow}$, etc. and new domains such as $\mathbb{T}$, $\mathbb{I}$, $\mathbb{T}_{\infty}$, $\mathbb{S}$, $\overline{\mathbb{S}}$, etc. are declared as constants. For instance

```
declare top_infty.
declare ’alpha^’.
declare time.
declare cltime.
declare intvls.
```

are declarations for $+\infty$, $\alpha^{\uparrow}$, $\mathbb{T}$, $\mathbb{T}_{\infty}$ and $\mathbb{I}$, respectively. Variables to be bound by a quantifier must be declared as *object variables* (apart from x, which is an object variable by default).

```
declare_object_variable i.
declare_object_variable t.
```

Functional and logical operators are declared together with their arity. For instance

```
declare alpha(_).
declare omega(_).
declare left_closed(_).
declare over_time(_,_).
declare over_intvl(_,_).
```

are declarations for functions $\alpha, \omega : \mathbb{I} \rightarrow \mathbb{T}$ and predicates lcl, $\downarrow_{\mathbb{T}}$ and $\downarrow_{\mathbb{I}}$, respectively. The $\downarrow$ derived operator is implemented as an abbreviation.

```
abbreviation over_time(P,[i,t]) ===
        over_time(over_intvl(P,i),t)).
```

Most of the axioms given in the previous section are easily implemented. For instance, Axiom **I5** is implemented as follows.

```
axiom intvl_5 === all i:intvls all x:time
        (alpha(i) < x and x < omega(i) => x in i).
```

The axioms of the **predintvls** theory are more complex to implement. The special brackets defined in Section 5 are implemented in Ergo as quantifiers. In fact $[\!(\!-P-\!)\!]$ is a set of intervals obtained by quantifying on all possible intervals $i : \mathbb{I}$ and all possible time points $t : \mathbb{T}$ such that if $t \in i$ then all occurrences of lifted form in P are instantiated on i and t.

The syntax of an Ergo quantifier definition is as follows.

$$QUANTIFIER \ ::= \ \texttt{define !!}NAME\ OVL\ BODY\ \texttt{===}\ TERM.$$

This command defines a quantifier called *NAME*, denoted by `!!`*NAME*. *OVL* is a list of object variables, *BODY* is a Prolog variable and *TERM* can be any legal term of the current theory, provided it has no free object variables or meta-variables except for *BODY*. Also, all occurrences of *BODY* in *TERM* must be within a binding of all object variables in *OVL*.

Therefore Axiom **Puu** is expressed in Ergo as the following definition.

```
define !!puu [i,t] P === set_of i:intvls all t:time
                            (t in i => (over(P,[i,t]))).
```

The operator `set_of` is defined in **zfc**, which is one of the Standard Theories of Ergo and axiomatises the Zermelo Frænkel Set Theory.

Axioms **Ptt** and **Pii** are implemented in Ergo as the following definitions.

```
define !!ptt [i,t] P === all i:intvls all t:time
                            (t in i => (over(P,[i,t]))).

define !!pii [i] P === all i:intvls (over_intvl(P,i)).
```

Theories **time**, **intvls**, **intvlsets**, **predintvls** and **lifting** are implemented respectively by the files `time.thy`, `intvls.thy`, `intvlsets.thy`, `predintvls.thy` and **lifting.thy** with a total of about 1000 lines of code, plus additional code to extend **lifting.thy** depending on the application domain.

7 TIC Laws

Modelling an application in TIC consists of constructing a theory SPEC that models the application. In addition, theory **lifting** defines all lifted forms that have been used in SPEC and contains the axioms for the propagation of the $\downarrow_{\mathbb{T}}$ and $\downarrow_{\mathbb{I}}$ instantiation operators through the arguments of all functions and functional, relational and slash logical operators used in SPEC. The **time** abstract theory is interpreted in the appriopriate current theory, which may be extended by imposing additional axioms on **time**. For example, if **time** is extended with the axiom

T31 $nn(\mathbb{T})$

and interpreted into the theory of the real numbers, then the resulting time domain is the set of the non-negative real numbers. We can also restrict $\mathbb{I}$ to contain only bounded intervals.

I13 $\forall\, i \in \mathbb{I}.\, -\infty < \alpha(i)$
I14 $\forall\, i \in \mathbb{I}.\, \omega(i) < +\infty$

In the following, we will assume TIC theories extended with the axioms **T31**, **I13** and **I14** and the **time** abstract theory interpreted in the Reals. that is time domain $\mathbb{T} = \mathbb{R}_0^+$ and the set of intervals $\mathbb{I}$ contains only unbounded intervals.

In order to facilitate the proof, high level laws for TIC may be given in the form of theorems and proved in Ergo. Previous works [5,15] contain many such laws. In this paper we consider only the following five laws.

Law1 $\langle\!| P \Rightarrow Q |\!\rangle \Rightarrow \llbracket P \rrbracket \subseteq \llbracket Q \rrbracket$
Law2 $\llbracket P \rrbracket \cap \llbracket Q \rrbracket = \llbracket P \wedge Q \rrbracket$
Law3 $\llbracket P \rrbracket \cup \llbracket Q \rrbracket \subseteq \llbracket P \vee Q \rrbracket$
Law4 $\langle\!| P |\!\rangle \Rightarrow \llbracket Q \rrbracket = \llbracket Q \wedge P \rrbracket$
Law5 $\{(n \cdot t \ldots (n+1) \cdot t] \mid n \in \mathbb{N}\} \subseteq \llbracket \langle\langle P \rangle\rangle \rrbracket \Leftrightarrow \llbracket \alpha^{\uparrow} > 0 \rrbracket \subseteq \llbracket \langle\langle P \rangle\rangle \rrbracket$

The proofs of these laws are performed using the theories that implement TIC and some "Standard Theories" such as **zfc**, the Zermelo Frænkel Set Theory, **cprop**, the Classical Propositional Logic Theory and **cpred**, the Classical First Order Logic Theory.

Law1 defines the monotonicity of $\llbracket \_ \rrbracket$ on predicates: more restrictive predicates define smaller sets of intervals. Notice that $P \Rightarrow Q$ needs to be enclosed between $\langle\!|$ and $|\!\rangle$ in order to instantiate all lifted functions on all possible time intervals and points in time

$$\begin{aligned}
&\langle\!| P \Rightarrow Q |\!\rangle \\
&\quad = \forall\, i : \mathbb{I}.\ \forall\, t : \mathbb{T}.\ (t \in i \Rightarrow ((P \Rightarrow Q){\downarrow}(i,t))) && \textbf{Ptt} \\
&\quad = \forall\, i : \mathbb{I}.\ \forall\, t : \mathbb{T}.\ (t \in i \Rightarrow \\
&\qquad\qquad ((P{\downarrow}(i,t)) \Rightarrow (Q{\downarrow}(i,t)))) && \textbf{L}{\Rightarrow} \\
&\quad \Rightarrow \forall\, i : \llbracket P \rrbracket.\ \forall\, t : \mathbb{T}.\ (t \in i \Rightarrow \\
&\qquad\qquad ((P{\downarrow}(i,t)) \Rightarrow (Q{\downarrow}(i,t)))) && \text{in theory } \textbf{zfc}
\end{aligned}$$

Moreover, from **Puu**, again using theory **zfc**, we have

$$\forall\, i : \llbracket P \rrbracket.\ \forall\, t : \mathbb{T}.\ (t \in i \Rightarrow (P{\downarrow}(i,t)))$$

The conjunction of these two predicates can be transformed using **cprop** into the following predicate.

$$\forall\, i : \llbracket P \rrbracket.\ \forall\, t : \mathbb{T}.\ (t \in i \Rightarrow ((P{\downarrow}(i,t)) \wedge ((P{\downarrow}(i,t)) \Rightarrow (Q{\downarrow}(i,t)))))$$

$$\begin{aligned}
&= \forall\, i : \llbracket P \rrbracket .\ \forall\, t : \mathbb{T}.\ (t \in i \Rightarrow (Q{\downarrow}(i,t))) && \text{in theory } \mathbf{cprop}\\
&= \forall\, i : \mathbb{I}.\ (i \in \llbracket P \rrbracket \Rightarrow \forall\, t : \mathbb{T}.\ (t \in i \Rightarrow (Q{\downarrow}(i,t)))) && \text{in theory } \mathbf{zfc}\\
&= \forall\, i : \mathbb{I}.\ (i \in \llbracket P \rrbracket \Rightarrow i \in \llbracket Q \rrbracket) && \text{in theory } \mathbf{zfc}\\
&= \llbracket P \rrbracket \subseteq \llbracket Q \rrbracket && \text{in theory } \mathbf{zfc}
\end{aligned}$$

Law2 defines a pleasing relationship between intersection and conjunction. The proof of **Law2** proceeds as follows.

$$\begin{aligned}
&\llbracket P \rrbracket \cap \llbracket Q \rrbracket\\
&\quad = \{i : \mathbb{I} \mid \forall\, t : \mathbb{T}.\ (t \in i \Rightarrow (P{\downarrow}(i,t)))\} \cap\\
&\qquad \{i : \mathbb{I} \mid \forall\, t : \mathbb{T}.\ (t \in i \Rightarrow (Q{\downarrow}(i,t)))\} && \mathbf{Puu}\\
&\quad = \{i : \mathbb{I} \mid (\forall\, t : \mathbb{T}.\ (t \in i \Rightarrow (P{\downarrow}(i,t))))\ \wedge\\
&\qquad\qquad (\forall\, t : \mathbb{T}.\ (t \in i \Rightarrow (Q{\downarrow}(i,t)))\} && \text{in theory } \mathbf{zfc}\\
&\quad = \{i : \mathbb{I} \mid (\forall\, t : \mathbb{T}.\ (t \in i \Rightarrow (P{\downarrow}(i,t))\ \wedge\\
&\qquad\qquad\qquad\qquad (Q{\downarrow}(i,t)))\} && \text{in theory } \mathbf{cpred}\\
&\quad = \{i : \mathbb{I} \mid \forall\, t : \mathbb{T}.\ (t \in i \Rightarrow (P \wedge Q){\downarrow}(i,t))\} && \mathbf{L}\wedge\\
&\quad = \llbracket P \wedge Q \rrbracket && \mathbf{Puu}
\end{aligned}$$

Law3 defines a weaker relationship between union and disjunction. The proof of **Law3** proceeds as follows.

$$\begin{aligned}
&\llbracket P \rrbracket \cup \llbracket Q \rrbracket\\
&\quad = \{i : \mathbb{I} \mid \forall\, t : \mathbb{T}.\ (t \in i \Rightarrow (P{\downarrow}(i,t)))\} \cup\\
&\qquad \{i : \mathbb{I} \mid \forall\, t : \mathbb{T}.\ (t \in i \Rightarrow (Q{\downarrow}(i,t)))\} && \mathbf{Puu}\\
&\quad \subseteq \{i : \mathbb{I} \mid \forall\, t : \mathbb{T}.\ (t \in i \Rightarrow (P{\downarrow}(i,t)) \vee (Q{\downarrow}(i,t)))\} \cup\\
&\qquad \{i : \mathbb{I} \mid \forall\, t : \mathbb{T}.\ (t \in i \Rightarrow (P{\downarrow}(i,t)) \vee (Q{\downarrow}(i,t)))\} && \text{in theory } \mathbf{zfc}\\
&\quad = \{i : \mathbb{I} \mid \forall\, t : \mathbb{T}.\ (t \in i \Rightarrow (P{\downarrow}(i,t)) \vee (Q{\downarrow}(i,t)))\} && \text{in theory } \mathbf{zfc}\\
&\quad = \{i : \mathbb{I} \mid \forall\, t : \mathbb{T}.\ (t \in i \Rightarrow (P \vee Q){\downarrow}(i,t))\} && \mathbf{L}\vee\\
&\quad = \llbracket P \vee Q \rrbracket && \mathbf{Puu}
\end{aligned}$$

The relationship between union and disjunction is weaker than the relationship between intersection and conjunction because we can distribute the universal quantification on the conjunction of predicates, but not on the disjunction of predicates. In fact, if the disjunction of two predicates is true on an interval, it is not necessary that one of the two predicates is true on the whole interval.

Law4 allows the insertion of a predicate that is true everywhere (i.e. at all points in time) within the special brackets as a conjunction. Let us suppose that $\langle\!| P |\!\rangle$ is true.

$$\begin{aligned}
&\llbracket Q \rrbracket\\
&\quad = \{i : \mathbb{I} \mid \forall\, t : \mathbb{T}.\ (t \in i \Rightarrow Q{\downarrow}(i,t))\} && \mathbf{Puu}\\
&\quad = \{i : \mathbb{I} \mid \forall\, t : \mathbb{T}.\ (t \in i \Rightarrow (Q{\downarrow}(i,t)) \wedge \langle\!| P |\!\rangle)\} && \text{since } \langle\!| P |\!\rangle \text{ is true}
\end{aligned}$$

$$
\begin{aligned}
&= \{ i : \mathbb{I} \mid \forall\, t : \mathbb{T}.\, (t \in i \Rightarrow \\
&\qquad (Q{\downarrow}(i,t)) \wedge \forall\, i' : \mathbb{I}.\ \forall\, t' : \mathbb{T}.\ P{\downarrow}(i',t'))\} && \textbf{Ptt} \\
&= \{ i : \mathbb{I} \mid \forall\, t : \mathbb{T}.\, (t \in i \Rightarrow \\
&\qquad (Q{\downarrow}(i,t)) \wedge (P{\downarrow}(i,t)))\} && \text{in theory } \textbf{zfc} \\
&= \{ i : \mathbb{I} \mid \forall\, t : \mathbb{T}.\, (t \in i \Rightarrow ((Q \wedge P){\downarrow}(i,t)))\} && \textbf{L}\wedge \\
&= [\![- Q \vee P -]\!] && \textbf{Puu}
\end{aligned}
$$

Law5 defines an induction law. It allows a property that holds on all consecutive intervals of a fixed length to be assumed everywhere. The proof of **Law5** is the following.

$$
\begin{aligned}
&\{(n \cdot t \ldots (n+1) \cdot t] \mid n \in \mathbb{N}\} \subseteq [\![- \langle\!\langle P \rangle\!\rangle -]\!] \\
&\quad\Leftrightarrow \forall\, n \in \mathbb{N}.\, (n \cdot t \ldots (n+1) \cdot t] \in [\![- \langle\!\langle P \rangle\!\rangle -]\!] && \text{in theory } \textbf{zfc} \\
&\quad\Leftrightarrow \forall\, n \in \mathbb{N}.\, (n \cdot t \ldots (n+1) \cdot t] \in \{ i \in \mathbb{I} \mid \forall\, t' : \mathbb{T}. \\
&\qquad\qquad t' \in i \Rightarrow \langle\!\langle P \rangle\!\rangle{\downarrow}(i,t')\} && \textbf{Puu} \\
&\quad\Leftrightarrow \forall\, n \in \mathbb{N}.\ \forall\, t' : \mathbb{T}.\ t' \in (n \cdot t \ldots (n+1) \cdot t] \Rightarrow \\
&\qquad\qquad \langle\!\langle P \rangle\!\rangle{\downarrow}((n \cdot t \ldots (n+1) \cdot t], t') && \text{in theory } \textbf{zfc} \\
&\quad\Leftrightarrow \forall\, n \in \mathbb{N}.\ \forall\, t' : \mathbb{T}.\ t' \in (n \cdot t \ldots (n+1) \cdot t] \Rightarrow \\
&\qquad\qquad (\forall\, i' : \mathbb{I}.\ P{\downarrow_{\mathbb{I}}}i'){\downarrow}((n \cdot t \ldots (n+1) \cdot t], t') && \textbf{Pii} \\
&\quad\Leftrightarrow \forall\, n \in \mathbb{N}.\ \forall\, t' : \mathbb{T}.\ t' \in (n \cdot t \ldots (n+1) \cdot t] \Rightarrow \\
&\qquad\qquad (\forall\, i' : \mathbb{I}.\ P{\downarrow_{\mathbb{I}}}i'){\downarrow_{\mathbb{I}}}(n \cdot t \ldots (n+1) \cdot t]{\downarrow_{\mathbb{T}}}t' && \textbf{LD}{\downarrow} \\
&\quad\Leftrightarrow \forall\, n \in \mathbb{N}.\ \forall\, t' : \mathbb{T}.\ t' \in (n \cdot t \ldots (n+1) \cdot t] \Rightarrow \\
&\qquad\qquad (\forall\, i' : \mathbb{I}.\ (P{\downarrow_{\mathbb{I}}}i'{\downarrow_{\mathbb{I}}}(n \cdot t \ldots (n+1) \cdot t])){\downarrow_{\mathbb{T}}}t' && \textbf{LI}\forall \\
&\quad\Leftrightarrow \forall\, n \in \mathbb{N}.\ \forall\, t' : \mathbb{T}.\ t' \in (n \cdot t \ldots (n+1) \cdot t] \Rightarrow \\
&\qquad\qquad (\forall\, i' : \mathbb{I}.\ P{\downarrow_{\mathbb{I}}}i'){\downarrow_{\mathbb{T}}}t' && \textbf{LDI} \\
&\quad\Leftrightarrow \forall\, t' : \mathbb{T}.\ 0 < t' \Rightarrow (\forall\, i : \mathbb{I}.\ P{\downarrow_{\mathbb{I}}}i'){\downarrow_{\mathbb{T}}}t' && \text{in theory } \textbf{zfc} \\
&\quad\Leftrightarrow \forall\, i : \mathbb{I}.\ 0 < \alpha(i) \Rightarrow (\forall\, t' : \mathbb{T}.\ t' \in i \Rightarrow (\forall\, i' : \mathbb{I}.\ P{\downarrow_{\mathbb{I}}}i'){\downarrow_{\mathbb{T}}}t') \\
&\quad\Leftrightarrow \forall\, i : \mathbb{I}.\ i \in \{ j : \mathbb{I} \mid 0 < \alpha(j)\} \Rightarrow \\
&\qquad\qquad i \in \{ j : \mathbb{I} \mid (\forall\, t' : \mathbb{T}.\ t' \in j \Rightarrow (\forall\, i' : \mathbb{I}.\ P{\downarrow_{\mathbb{I}}}i'){\downarrow_{\mathbb{T}}}t')\} \\
&\qquad\qquad\qquad \text{in theory } \textbf{zfc} \\
&\quad\Leftrightarrow \forall\, i : \mathbb{I}.\ i \in \{ j : \mathbb{I} \mid \forall\, t : \mathbb{T}.\ t \in j \Rightarrow 0 < \alpha(j)\} \Rightarrow \\
&\qquad\qquad i \in \{ j : \mathbb{I} \mid (\forall\, t' : \mathbb{T}.\ t' \in j \Rightarrow (\forall\, i' : \mathbb{I}.\ P{\downarrow_{\mathbb{I}}}i'){\downarrow_{\mathbb{T}}}t')\} \\
&\qquad\qquad\qquad \text{in theory } \textbf{zfc} \\
&\quad\Leftrightarrow \forall\, i : \mathbb{I}.\ i \in \{ j : \mathbb{I} \mid \forall\, t : \mathbb{T}.\ t \in j \Rightarrow 0 < \alpha^{\uparrow}{\downarrow}(j,t)\} \Rightarrow \\
&\qquad\qquad i \in \{ j : \mathbb{I} \mid (\forall\, t' : \mathbb{T}.\ t' \in j \Rightarrow (\forall\, i' : \mathbb{I}.\ P{\downarrow_{\mathbb{I}}}i'){\downarrow_{\mathbb{T}}}t')\} \\
&\qquad\qquad\qquad \textbf{A}\alpha \\
&\quad\Leftrightarrow \forall\, i : \mathbb{I}.\ i \in \{ j : \mathbb{I} \mid \forall\, t : \mathbb{T}.\ t \in j \Rightarrow 0{\downarrow}(j,t) < \alpha^{\uparrow}{\downarrow}(j,t)\} \Rightarrow \\
&\qquad\qquad i \in \{ j : \mathbb{I} \mid (\forall\, t' : \mathbb{T}.\ t' \in j \Rightarrow (\forall\, i' : \mathbb{I}.\ P{\downarrow_{\mathbb{I}}}i'){\downarrow_{\mathbb{T}}}t')\} \\
&\qquad\qquad\qquad \textbf{A0}
\end{aligned}
$$

$$\begin{aligned}
&\Leftrightarrow \forall\, i : \mathbb{I}.\; i \in \{j : \mathbb{I} \mid \forall\, t : \mathbb{T}.\; t \in j \Rightarrow (0 < \alpha^{\uparrow}){\downarrow}(j,t)\} \Rightarrow \\
&\qquad\qquad i \in \{j : \mathbb{I} \mid (\forall\, t' : \mathbb{T}.\; t' \in j \Rightarrow (\forall\, i' : \mathbb{I}.\; P{\downarrow_{\mathbb{I}}}i'){\downarrow_{\mathbb{T}}}t')\} \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \mathbf{L<} \\
&\Leftrightarrow \forall\, i : \mathbb{I}.\; i \in \llbracket\!-0 < \alpha^{\uparrow}-\!\rrbracket \Rightarrow \\
&\qquad\qquad i \in \{j : \mathbb{I} \mid (\forall\, t' : \mathbb{T}.\; t' \in j \Rightarrow (\forall\, i' : \mathbb{I}.\; P{\downarrow_{\mathbb{I}}}i'){\downarrow_{\mathbb{T}}}t')\} \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \mathbf{L<} \\
&\Leftrightarrow \forall\, i : \mathbb{I}.\; i \in \llbracket\!-0 < \alpha^{\uparrow}-\!\rrbracket \Rightarrow \\
&\qquad\qquad i \in \{j : \mathbb{I} \mid (\forall\, t' : \mathbb{T}.\; t' \in j \Rightarrow (\forall\, i' : \mathbb{I}.\; P{\downarrow_{\mathbb{I}}}i'{\downarrow_{\mathbb{I}}}j){\downarrow_{\mathbb{T}}}t')\} \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \mathbf{LDI} \\
&\Leftrightarrow \forall\, i : \mathbb{I}.\; i \in \llbracket\!-0 < \alpha^{\uparrow}-\!\rrbracket \Rightarrow \\
&\qquad\qquad i \in \{j : \mathbb{I} \mid (\forall\, t' : \mathbb{T}.\; t' \in j \Rightarrow (\forall\, i' : \mathbb{I}.\; P{\downarrow_{\mathbb{I}}}i'){\downarrow_{\mathbb{I}}}j{\downarrow_{\mathbb{T}}}t')\} \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \mathbf{LI}\forall \\
&\Leftrightarrow \forall\, i : \mathbb{I}.\; i \in \llbracket\!-0 < \alpha^{\uparrow}-\!\rrbracket \Rightarrow \\
&\qquad\qquad i \in \{j : \mathbb{I} \mid (\forall\, t' : \mathbb{T}.\; t' \in j \Rightarrow \langle\!\langle P \rangle\!\rangle{\downarrow_{\mathbb{I}}}j{\downarrow_{\mathbb{T}}}t')\} \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \mathbf{Pii} \\
&\Leftrightarrow \forall\, i : \mathbb{I}.\; i \in \llbracket\!-0 < \alpha^{\uparrow}-\!\rrbracket \Rightarrow i \in \{j : \mathbb{I} \mid (\forall\, t' : \mathbb{T}.\; t' \in j \Rightarrow \langle\!\langle P \rangle\!\rangle{\downarrow}(j,t')\} \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \mathbf{LD}{\downarrow} \\
&\Leftrightarrow \forall\, i : \mathbb{I}.\; i \in \llbracket\!-0 < \alpha^{\uparrow}-\!\rrbracket \Rightarrow i \in \llbracket\!-\langle\!\langle P \rangle\!\rangle-\!\rrbracket \quad \mathbf{Puu} \\
&\Leftrightarrow \llbracket\!-0 < \alpha^{\uparrow}-\!\rrbracket \subseteq \llbracket\!-\langle\!\langle P \rangle\!\rangle-\!\rrbracket \quad \text{in theory } \mathbf{zfc} \\
&\Leftrightarrow \llbracket\!-\alpha^{\uparrow} > 0-\!\rrbracket \subseteq \llbracket\!-\langle\!\langle P \rangle\!\rangle-\!\rrbracket
\end{aligned}$$

8 Example

Let us consider a system consisting of an *object* and a *measurement device*. The object is moving at a speed whose magnitude never exceeds a constant value V (see Figure 2(a)).

$$\textbf{MaxSpeed} \quad \langle\!|\; |\,(\mathrm{d}d_r/\mathrm{d}t)^{\uparrow}\,| \leqslant V \;|\!\rangle$$

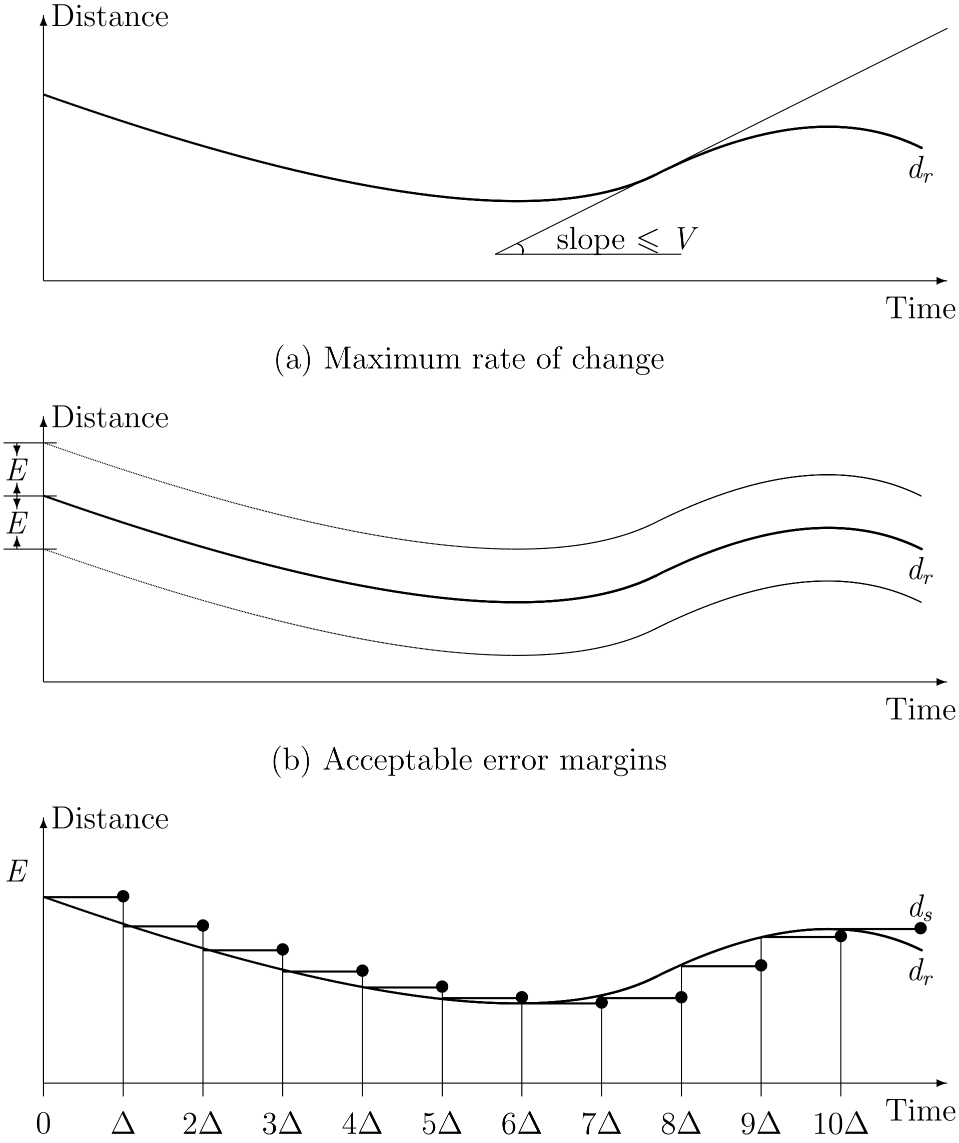
where $d_r : \mathbb{T} \to \mathbb{R}_0^+$ defines the distance between the object and the measurement device and $\mathrm{d}d_r/\mathrm{d}t$ is the first derivative of the distance with respect to time, that is the speed of the object.

The *requirement* of the system is that the device must be able to measure the object's distance with an accuracy of E (see Figure 2(b)).

$$\textbf{Requirement} \quad \llbracket\!-\alpha^{\uparrow} > 0-\!\rrbracket \subseteq \llbracket\!-{d_s}^{\uparrow} \in [{d_r}^{\uparrow} - E \;...\; {d_r}^{\uparrow} + E]-\!\rrbracket$$

where $d_s : \mathbb{T} \to \mathbb{R}_0^+$ defines the distance displayed by the device. At any time after the initial time 0 the measured distance d_s must approximate the actual distance d_r with an error not greater than E, that is $d_s = d_r \pm E$.

In a proposal *design* of the system the device samples the object's distance



(a) Maximum rate of change

(b) Acceptable error margins

(c) Periodically sampled measurements

Fig. 2. Parameters of the distance measurement device.

every Δ time units (see Figure 2(c)).

$$\textbf{Design}\ \ \{(n\cdot\Delta\ \ldots\ (n+1)\cdot\Delta] \mid n\in\mathbb{N}\} \subseteq [\![\lceil {d_s}^{\uparrow} = d_r(\alpha)\rceil]\!]$$

The device samples the distance at times 0, Δ, $2\cdot\Delta$, ..., $n\cdot\Delta$, ... and, for every interval $(n\cdot\Delta\ \ldots\ (n+1)\cdot\Delta]$, at each time t in such an interval the displayed distance $d_s(t)$ is equal to the actual distance at the left endpoint of the interval $d_r(n\cdot\Delta)$.

In order to prove that the *design* meets the *requirement* it is necessary to axiomatise the algebraic domains underlying our application. In our example we make use of the domain $\mathbb{R}_0^+$ of the non-negative real numbers to describe

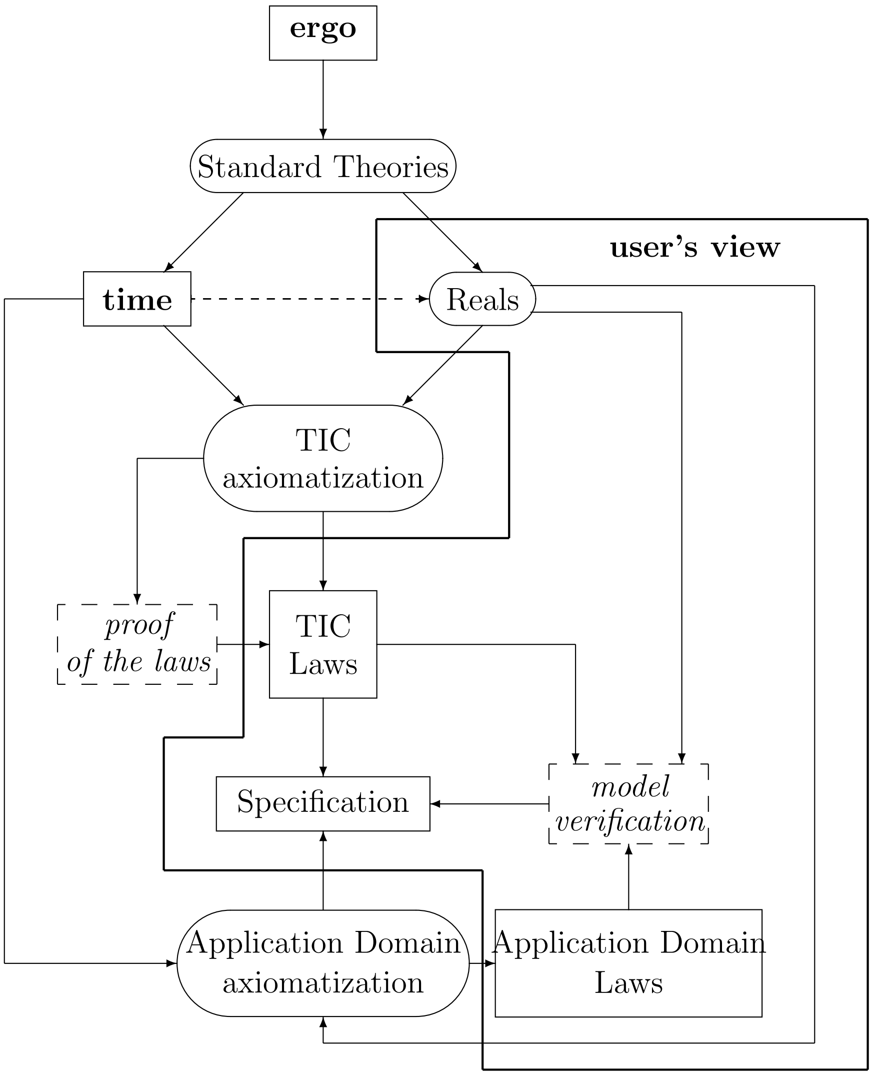


Fig. 3. User's view of the theory graph

the spatial distance, extended with the notion of first derivative with respect to time. This axiomatisation is represented in Figure 3 by the oval block "Application Domain axiomatisation", which makes use of both the **time** theory and the "Reals" theory graph. Analogously to what has been done for the "TIC axiomatisation" we can define the theory of "Application Domain Laws", whose laws are more high level than the axioms in the underlying theory graph. In our proof we are going to use the following application domain law.

LawD1 $\llparenthesis\ |\,(\mathrm{d}d_r/\mathrm{d}t)^{\uparrow}\,| \leqslant V \ \Rightarrow |\, {d_r}^{\uparrow} - d_r(\alpha^{\uparrow})\,| \leqslant V(\omega^{\uparrow} - \alpha^{\uparrow})\rrparenthesis$

This law ensures that for each time interval i, if the magnitude of the first derivative of the distance d_r is not greater than constant V, then the value of

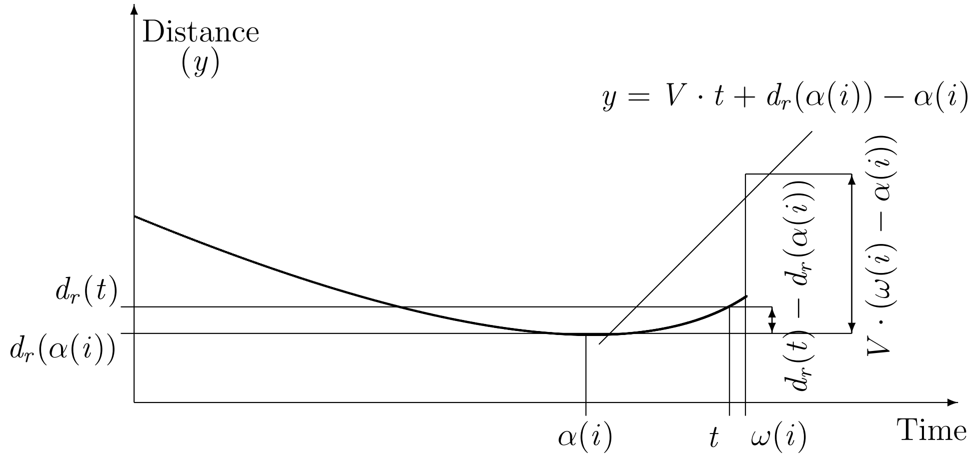

Distance
(y)
$y = V \cdot t + d_r(\alpha(i)) - \alpha(i)$
$d_r(t)$
$d_r(\alpha(i))$
$d_r(t) - d_r(\alpha(i))$
$V \cdot (\omega(i) - \alpha(i))$
$\alpha(i)$
t
$\omega(i)$
Time

Fig. 4. Bounded rate of change

the d_r distance at any time t within the interval is below the line

$$y = V \cdot t + d_r(\alpha(i)) - \alpha(i)$$

as shown in Figure 4.

The verification of the requirement is carried out by performing the following proof.

$$\begin{array}{ll}
\{(n \cdot \Delta \ldots (n+1) \cdot \Delta] \mid n \in \mathbb{N}\} & \\
\quad \subseteq \llbracket\!- {d_s}^{\uparrow} = d_r(\alpha^{\uparrow}) -\!\rrbracket & \textbf{Design} \\
\quad = \llbracket\!- {d_s}^{\uparrow} = d_r(\alpha^{\uparrow}) \wedge \mid (\mathrm{d}d_r/\mathrm{d}t)^{\uparrow} \mid \leqslant V -\!\rrbracket & \textbf{Maxspeed} \text{ and } \textbf{Law4} \\
\quad \subseteq \llbracket\!- {d_s}^{\uparrow} = d_r(\alpha^{\uparrow}) \wedge \mid {d_r}^{\uparrow} - d_r(\alpha^{\uparrow}) \mid \leqslant V \cdot (\omega^{\uparrow} - \alpha^{\uparrow}) -\!\rrbracket & \\
 & \textbf{LawD1} \text{ and } \textbf{Law1} \\
\quad \subseteq \llbracket\!- {d_s}^{\uparrow} = d_r(\alpha^{\uparrow}) \wedge \mid {d_r}^{\uparrow} - d_r(\alpha^{\uparrow}) \mid \leqslant V \cdot \Delta -\!\rrbracket & \\
\qquad\qquad\qquad \text{since } \forall\, n \in \mathbb{N}.\ (\omega - \alpha)((n \cdot \Delta \ldots (n+1) \cdot \Delta]) = \Delta & \\
\quad = \llbracket\!- {d_s}^{\uparrow} = d_r(\alpha^{\uparrow}) \wedge\ d_r(\alpha^{\uparrow}) \in [{d_r}^{\uparrow} - V \cdot \Delta \ldots {d_r}^{\uparrow} + V \cdot \Delta] -\!\rrbracket & \\
\quad = \llbracket\!- {d_s}^{\uparrow} = d_r(\alpha^{\uparrow}) \wedge\ {d_s}^{\uparrow} \in [{d_r}^{\uparrow} - V \cdot \Delta \ldots {d_r}^{\uparrow} + V \cdot \Delta] -\!\rrbracket & \\
\quad \subseteq \llbracket\!- {d_s}^{\uparrow} \in [{d_r}^{\uparrow} - V \cdot \Delta \ldots {d_r}^{\uparrow} + V \cdot \Delta] -\!\rrbracket & \textbf{Law1}
\end{array}$$

Then:

$$\begin{array}{ll}
\llbracket\!- \alpha^{\uparrow} > 0 -\!\rrbracket & \\
\quad \subseteq \llbracket\!- {d_s}^{\uparrow} \in [{d_r}^{\uparrow} - V \cdot \Delta \ldots {d_r}^{\uparrow} + V \cdot \Delta] -\!\rrbracket & \textbf{Law5} \\
\quad \subseteq \llbracket\!- {d_s}^{\uparrow} \in [{d_r}^{\uparrow} - E \ldots {d_r}^{\uparrow} + E] -\!\rrbracket & \textbf{Law1} \text{ and } V \cdot \Delta \leqslant E
\end{array}$$

the final step tell us the essential relationship between the worst case rate of change V, the acceptable error E, and the sampling period Δ.

We can notice that the proof is performed with no recourse to the axioms used to define TIC. Only TIC Laws and Application Domain Laws, such as

LawD1 are applied in the proof, together with some simple properties of the real numbers. Therefore, a user can perform proofs using only TIC Laws, without any knowledge about the axioms used to implement TIC. Any change to the axiomatisation would be transparent to the user, provided the laws are still valid.

9 Conclusion

We have provided an axiomatisation for the Timed Interval Calculus. The axiomatisation is based on a general notion of time domain, which captures all the minimal properies that a time domain has to meet [3]. It has been implemented in the Ergo theorem prover and has allowed the machine-assisted proof of many laws, such as those presented in Section 7, for reasoning about predicates expressed using TIC's special brackets. The proof of some of these laws was actually too complex to be performed in detail without tool support. We are currently using the implementation of TIC in Ergo to prove some of the applications that have been previously verified by hand [2,15].

Acknowledgements

I would like to thank Tony Cant, Colin Fidge, Ian Hayes, Brendan Mahony and Luke Wildman for helpful discussions. Colin also suggested the example used in this paper and reviewed a draft of this paper providing many useful comments. This research has been supported by the Information Technology Division of the Defence Science & Technology Organisation, and by a University of Queensland ECR grant.

References

[1] H. Becht, A. Bloesch, R. Nickson, and M. Utting. Ergo 4.1 reference manual. Technical Report 96-31, Software Verification Research Centre, School of Information Technology, The University of Queensland, Brisbane, Australia, Nov 1996.

[2] A. Cerone. Process algebra versus axiomatic specification of a real-time protocol. In *Algebraic Metodology and Software Technology (AMAST'00)*, volume 1816 of *Lecture Notes in Computer Science*, pages 57–72. Springer-Verlag, 2000.

[3] A. Cerone and A. Maggiolo-Schettini. Time-based expressivity of time Petri nets for system specification. *Theoretical Computer Science*, 216(1–2):1–53, March 1999.

[4] C. J. Fidge, I. J. Hayes, B. P. Mahony, and A. K. Wabenhorst. Real-time specification and reasoning using maximal intervals. In *Proc. of the 6th Annual*

Australasian Conference on Parallel and Real-Time Systems (PART'99), pages 344–354, Singapore, 1999. Springer-Verlag.

[5] C. J. Fidge, I. J. Hayes, A. P. Martin, and A. K. Wabenhorst. A set-theoretic model for real-time specification and reasoning. In *Mathematics of Program Construction (MPC'98)*, volume 1422 of *Lecture Notes in Computer Science*, pages 188–206. Springer-Verlag, 1998.

[6] B. P. Mahony and I. J. Hayes. A case-study in timed refinement: A mine pump. *IEEE Transactions on Software Engineering*, 18(9):817–826, Sep 1992.

[7] A. Martin and C. Fidge. Lifting in Z. In *Computing: The Australasian Theory Symposium (CATS'01)*, volume 42 of *Electronic Notes in Theoretical Computer Science*. Elsevier, 2001.

[8] R. Nickson and I. Hayes. Supporting context in program refinement. *Science of Computer Programming*, 29(1):279–302, 1997.

[9] R. Nickson and M. Utting. A new face for Ergo: Adding a user interface to programmable theorem prover. Technical Report 95-42, Software Verification Research Centre, School of Information Technology, The University of Queensland, Brisbane, Australia, 1995.

[10] P. Robinson and A. Cheng. Qu-Prolog 3.2 refeence manual. Technical Report 93-18, Software Verification Research Centre, School of Information Technology, The University of Queensland, Brisbane, Australia, 1993.

[11] P. J. Robinson and J. Staples. Formalizing a hierarchical structure of practical mathematical reasoning. *J. Logic Computat.*, 3(1):47–61, 1993.

[12] J. Shield, I. Hayes, and D. Carrington. Using theory interpretation to mechanise the reals in a theorem prover. In *Computing: The Australasian Theory Symposium (CATS'01)*, volume 42 of *Electronic Notes in Theoretical Computer Science*. Elsevier, 2001.

[13] J. U. Skakkebæk. *A Verification Assistent for Real-Time Logic*. PhD thesis, Department of Computer Science, Technical University of Denmark, NOV 1994.

[14] M. Utting and K. Whitwell. Ergo user manual. Technical Report 93-19, Software Verification Research Centre, School of Information Technology, The University of Queensland, Brisbane, Australia, Feb 1994.

[15] A. Wabenhorst. Induction in the timed interval calculus. Technical Report 99-36, Software Verification Research Centre, School of Information Technology, The University of Queensland, Brisbane, Australia, Dec. 1999.