On the Completeness and Decidability of Duration Calculus with Iteration[*]

Dimitar P. Guelev and Dang Van Hung
Institute of Mathematics and Informatics,
Bulgarian Academy of Sciences
e-mail: `gelevdp@math.bas.bg`
United Nations University
International Institute for Software Technology
P.O.Box 3058, Macau
Fax: +853 712 940, Phone: +853 712 930
e-mail: `dvh@iist.unu.edu`

Abstract

The extension of the Duration Calculus (*DC*) by *iteration*, which is also known as *Kleene star*, enables the straightforward specification of repetitive behaviour in *DC* and facilitates the translation of design descriptions between *DC*, timed regular expressions and timed automata. In this paper we present axioms and a proof rule about iteration in *DC*. We consider abstract-time *DC* and its extension by a state-variable binding existential quantifier known as *higher-order DC* (*HDC*). We show that the ω-complete proof systems for *DC* and *HDC* known from our earlier work can be extended by our axioms and rule in various ways in order to axiomatise iteration completely. The additions we propose include either the proof rule or an induction axiom. We also present results on the decidability of a subset of the extension DC^* of *DC* by iteration.

Keywords: real-time systems, formal methods, duration calculus, completeness, decidability.

Introduction

The *Duration Calculus* (*DC*) was introduced by Zhou, Hoare and Ravn in [ZHR91] as a logic to specify requirements on real-time systems. *DC* is a first order classical interval-based real-time logic with one normal binary modality

[*]This paper supercedes the paper "Completeness and Decidability of a Fragment of Duration Calculus with Iteration", LNCS 1742, Springer-Verlag, 1999, pp. 139–150

known as *chop*. It was developed by augmenting the real-time variant of Interval Temporal Logic (*ITL*, [Mos85, Mos86]), with boolean expressions for *state* and real-valued terms to denote state *durations*. *DC* has been used successfully in many case studies such as [ZZ94, YWZP94, HZ95, DW96, WHCZ96, XH95, Dan98, PD99]. We refer the reader to [HZ97] or the recent monograph [ZH04] for a comprehensive introduction to *DC*.

DC was originally introduced for real time. Variants of *DC* have been developed for discrete time, combinations of real time and discrete time [PD98, He 99a] and abstract time [Gue98], where an arbitrary (commutative) linearly ordered group can be the model of time. Real-time and discrete-time semantics are best suited for applications. The more general abstract-time semantics gives some technical advantages for theoretical studies.

Iteration, also known as *Kleene star*, was introduced to *DC* to facilitate the reasoning about repetitive behaviour. Iteration is particularly important for the description of the repetitive behaviour of timed automata in *DC*. It facilitates the translation of designs between timed regular expressions, timed automata and *DC*. DC^* stands for the extension of *DC* by iteration. In [DW96] we developed a method for designing real-time hybrid systems from specifications written using a subset of DC^* which consists of the so-called *simple* DC^* formulas. Simple DC^* formulas are sufficient to describe the behaviour of timed automata. One can reason about the correctness of designs in terms of the semantics of DC^*. However, it would be more practical and interesting to be able to prove correctness syntactically. This requires the development of a proof system for DC^*.

A Hilbert-style proof system for *DC* (without iteration) was first presented in [HZ92]. This proof system was shown to be complete for real-time *DC* relative to real-time *ITL*. Validity in *DC* is not recursively enumerable and therefore no finitary complete proof system for *DC* exists. A small non-recursively enumerable subset of *DC* was presented in [Gue05]. The ω-completeness of a system with an ω-rule for abstract-time *DC* was shown in [Gue98]. In [Gue00c] it was shown that by adding a few axioms and (finitary) rules the scope of the ω-completeness of the system from [Gue98] can be extended to *DC* with quantification over state as introduced in [Pan95] also known as *Higher-order DC* (*HDC* or *HODC*, see [ZGZ00], where *HDC* has other useful features such as neighbourhood values [ZL94] and super-dense chop [HZ96] which we omit here.)

In this paper we study the deductive power of three groups of axioms and a rule for iteration in *DC* which we add to a variant of the ω-complete proof system from [Gue98]. The first group of axioms has been obtained from the axioms about Kleene star in Propositional Dynamic Logic (*PDL*, cf e.g. [AGM92]). It contains an induction axiom. This group was presented in the precursor of this paper [DG99]. Some examples which demonstrate the working of this group of axioms can be found in [He 99b]. The second group contains an induction axiom too. Unlike the first, its induction axiom is the instance about iteration of an axiom about the general least fixed point operator μ which was introduced to *DC* in [Pan95] and later studied in [Gue00b]. This axiom corresponds to *Park's rule* as known from the modal μ-calculus [Koz83]. This rule was formulated for

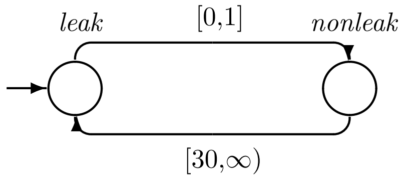

Figure 1: A simple gas burner design

DC in [Pan95] too. The third group has a proof rule instead of an induction axiom. Unlike the induction axioms, which are examples of a general pattern occurring in the axiomatisation of Kleene star in other logics, that proof rule refers to the *DC*-specific notion of state and its deductive power stems from the finite variability requirement which is imposed on state in *DC*. The idea behind this rule was earlier used in [Gue05] to express a finite variability requirement on propositional temporal letters in *DC*.

We show that adding our third group of axioms, which includes the proof rule, to the proof system from [Gue98] leads to an ω-complete proof system for DC^*. We show that iteration is definable in *HDC* and the correctness of the definition can be proved in the extension of an ω-complete proof system for *HDC* by the second group of axioms. We show that there is a straightforward correspondence between the induction axiom of that second group and the induction axiom of the first group.

To illustrate the working of our axioms, we employ the well-known simple gas burner example taken from [ZHR91]. The *DC* formula

$$S \rightleftharpoons \Box(\ell > 60 \Rightarrow (20 \textstyle\int leak \leq \ell)) \tag{1}$$

specifies that a gas burner can be in the *leak* state for no more than one-twentieth of the time in any time interval that is at least 1 minute long. Consider the gas burner design described by the real-time automaton shown on Fig. 1. We assume that *leak* is the initial state for the sake of simplicity. In this design *leak* becomes detected within 1 second and leaks are separated by at least 30 seconds. This can be specified by the DC^* formula

$$D \rightleftharpoons ((\lceil leak \rceil \wedge \ell \leq 1)^\frown(\lceil nonleak \rceil \wedge \ell \geq 30))^* \,. \tag{2}$$

In Section 6 we give a proof of $D \Rightarrow S$ by means of our axioms about iteration.

Both S and D are *simple* DC^* formulas. Validity is decidable for such formulas. This implies that whether a design written in simple DC^* is implementable can be decided algorithmically. Furthermore, the validity of implications from simple DC^* formulas to *DC* formulas of certain forms such as linear duration invariants (see [ZZYL94, LDZ97, DP98]) can be checked by a simple algorithm.

Structure of the paper

After a brief introduction to *DC* and its extensions by iteration and state variable binding quantifier we present our axioms and rule about iteration and show that adding some of these to a complete proof system for *DC* without iteration leads to a complete proof system for DC^*. Next we show that the proof rule can be replaced by each of the proposed induction axioms in a complete proof system for *HDC*, where iteration can also be defined explicitly. We illustrate the working of the proof rule by giving derivations of the alternative induction axioms and show how one of them can be derived using the other. We show how the explicit definition of iteration in *HDC* can be derived from our axioms about iteration and the state variable binding quantifier too. In a separate short section we explain how one of the proposed induction axioms can be obtained by translating a *PDL* induction axiom into *DC*. We derive some other interesting DC^* theorems in our system to illustrate its working too, and use one of them in a proof of $D \Rightarrow S$ about the gas burner design. Finally, we discuss work on the axiomatisation of iteration in related systems, summarise some decidability results about the subset of DC^* known as *simple* DC^* which have been obtained either independently or using connections with timed regular expressions and timed automata, and make some concluding remarks.

1 Preliminaries on the duration calculus

Here follows the formal definition of *DC*.

1.1 Language

A *DC vocabulary* consists of *constant symbols* $c, d, \ldots$, *individual variables* $x, y, z, \ldots$, *state variables* $P, Q, \ldots$, *temporal variables* $v, \ldots$, *function symbols* $f, g, \ldots$, *relation symbols* $R, \ldots$ and *temporal propositional letters* $A, B, \ldots$. The constant 0, addition $+$ equality $=$ and the temporal variable ℓ are mandatory in *DC* vocabularies.

Given a vocabulary, the definition of a *DC* language is essentially that of its sets of *state expressions* S, *terms* t and *formulas* φ. These sets can be defined by the following BNFs:

$$\begin{array}{rcl} S & ::= & \mathbf{0} \mid P \mid \neg S \mid S \vee S \\ t & ::= & c \mid x \mid v \mid \textstyle\int S \mid f(t, \ldots, t) \\ \varphi & ::= & A \mid R(t, \ldots, t) \mid \neg\varphi \mid (\varphi \vee \varphi) \mid (\varphi{}^\frown\varphi) \mid \exists x \varphi \end{array}$$

Terms and formulas with no occurrences of $^\frown$ (*chop*), nor of temporal variables, nor of $\int$, are called *rigid*.

The set of the variables which have free occurrences in a formula φ is denoted by $FV(\varphi)$. For sets of formulas Γ, $FV(\Gamma)$ is defined as $\bigcup\limits_{\varphi \in \Gamma} FV(\varphi)$. The state variables occurring freely in a formula φ are assumed to be in $FV(\varphi)$ too. All occurrences of state variables are free in *DC*, but not in *HDC*.

1.2 Semantics

Our completeness results about DC^* apply to the abstract semantics which was defined for *DC* in [Gue98] after the semantics of *ITL* from [Dut95a]. The linearly ordered set of the reals is the model of time in the original semantics of DC. The durations of real-time intervals are non-negative reals with ordinary arithmetic on them in that semantics. In the abstract-time case the durations form the monoid of the positive elements of some linearly ordered group [Sko00], and the time domain is isomorphic to a possibly unbounded interval in the same group. Here follow the detailed definitions.

Definition 1 *A* time domain *is a linearly ordered set* $\langle T, \leq \rangle$.

$\langle \mathbf{R}, \leq \rangle$, $\langle \mathbf{Z}, \leq \rangle$ and $\langle \mathbf{N}, \leq \rangle$ are examples of time domains.

Definition 2 *Given a time domain* $\langle T, \leq \rangle$, *we denote the set*

$$\{[t_1, t_2] : t_1, t_2 \in T, t_1 \leq t_2\}$$

of the closed and bounded intervals in T *by* $\mathbf{I}(T)$.

We use (and) to mark open ends of time intervals, for instance, $[t_1, t_2)$ stands for the semi-open interval $\{t \in T : t_1 \leq t < t_2\}$.

Definition 3 *A* duration domain *is a system of the type* $\langle D, +^{(2)}, 0^{(0)} \rangle$ *which satisfies the axioms*

$(D1)$ $\quad x + (y + z) = (x + y) + z$
$(D2)$ $\quad x + 0 = 0 + x = x$
$(D3)$ $\quad x + y = x + z \Rightarrow y = z,\ x + z = y + z \Rightarrow x = y$
$(D4)$ $\quad x + y = 0 \Rightarrow x = y = 0$
$(D5)$ $\quad \exists z(x + z = y \vee y + z = x),\ \exists z(z + x = y \vee z + y = x)$

For example, $\langle \mathbf{R}_+, +, 0 \rangle$ and $\langle \mathbf{N}, +, 0 \rangle$ are duration domains, but $\langle \mathbf{Z}, +, 0 \rangle$ is not, because it violates $D4$. The axioms $D1$-$D5$ do not imply commutativity of $+$ (cf. e.g. [Sko00]). However, to our knowledge, all practically relevant duration domains are commutative. Adding commutativity to $D1$-$D5$ affects neither the validity, nor the proofs of the results in this paper. In the sequel we assume that duration domains are linearly ordered by the relation

$$x \leq y \rightleftharpoons \exists z(x + z = y)\,. \tag{3}$$

We regard $\leq$ as a mandatory symbol in DC with (3) being its definition.

Definition 4 *Given a time domain* $\langle T, \leq \rangle$, *and a duration domain* $\langle D, +, 0 \rangle$, $m : \mathbf{I}(T) \to D$ *is a* measure *if it satisfies the axioms*

$(M1)$ $\quad m([\tau_1, \tau_2]) = m([\tau_1, \tau_2']) \Rightarrow \tau_2 = \tau_2'$
$(M2)$ $\quad m([\tau_1, \tau]) + m([\tau, \tau_2]) = m([\tau_1, \tau_2])$
$(M3)$ $\quad m([\tau_1, \tau_2]) = x + y \Rightarrow \exists \tau (m([\tau_1, \tau]) = x)$.

Definition 5 *An* abstract DC frame *is a tuple of the form* $F = \langle \langle T, \leq \rangle, \langle D, +, 0 \rangle, m \rangle$, *where* $\langle T, \leq \rangle$ *is a time domain,* $\langle D, +, 0 \rangle$ *is a duration domain, and* $m : \mathbf{I}(T) \to D$ *is a measure.*

The existence of a measure $m : \mathbf{I}(T) \to D$ clearly imposes restrictions on $\langle T, \leq \rangle$. Some linearly ordered sets do not admit such a measure for any duration domain.

Definition 6 *Given a DC vocabulary* $\mathbf{L}$ *and an abstract DC frame* $F = \langle\langle T, \leq\rangle, \langle D, +, 0\rangle, m\rangle$, *an* interpretation of $\mathbf{L}$ into F *is a mapping* I *of* $\mathbf{L}$ *which satisfies the following conditions:*

$I(c), I(x) \in D$ *for constant symbols* c *and individual variables* x*;*
$I(f) : D^n \to D$ *for* n*-place function symbols* f*;*
$I(v) : \mathbf{I}(T) \to D$ *for temporal variables* v*;*
$I(R) : D^n \to \{0, 1\}$ *for* n*-place relation symbols* R*;*
$I(P) : T \to \{0, 1\}$ *for state variables* P*;*
$I(A) : \mathbf{I}(T) \to \{0, 1\}$ *for temporal propositional letters* A*.*
$I(0) = 0$, $I(+) = +$, $I(=)$ *is* $=$ *and* $I(\ell) = m$*.*

The following condition, known as finite variability of state*, is imposed on the interpretations of state variables:*

> *For every* $[\tau_1, \tau_2] \in \mathbf{I}(T)$ *such that* $\tau_1 < \tau_2$*, and every state variable* P *there exist* $\tau'_1, \ldots, \tau'_n \in T$ *such that* $\tau_1 = \tau'_1 < \ldots < \tau'_n = \tau_2$ *and* $I(P)$ *is constant on the semi-open intervals* $[\tau'_i, \tau'_{i+1})$*,* $i = 1, \ldots, n-1$*.*

Finite variability can be defined as piece-wise continuity of $I(P)$ in the real-time case. This is less restrictive but leads to the same notion of validity in DC.

Definition 7 *Given DC vocabulary* $\mathbf{L}$, a DC abstract model for $\mathbf{L}$ *is a tuple of the form* $M = \langle F, I\rangle$ *where* F *is a DC abstract frame, and* I *is an* interpretation *of* $\mathbf{L}$ *into* F*.*

Definition 8 *Let* s *be in some DC vocabulary* $\mathbf{L}$*. Interpretations* I *and* J *of* $\mathbf{L}$ *into the same abstract frame are said to* s-agree, *if* $I(s') = J(s')$ *for all* s' *in* $\mathbf{L}$*, except possibly* s*.*

Given intervals σ_1 and σ_2 in the same time domain, we denote $\sigma_1 \cup \sigma_2$ by $\sigma_1{}^\frown\sigma_2$, in case $\max\sigma_1 = \min\sigma_2$. This use of $^\frown$ in our meta-language is related but formally different from its use in DC formulas. Since $^\frown$ is associative, we omit parentheses in expressions with consecutive occurrences of $^\frown$.

Definition 9 *Let* $\langle F, I\rangle$ *be an abstract DC model,* $F = \langle\langle T, \leq\rangle, \langle D, +, 0\rangle, m\rangle$ *and* $\tau \in T$*. Then the values* $I_\tau(S)$ *of state expressions* S *in the vocabulary of* I *are defined by the clauses:*

$$\begin{array}{lcl}
I_\tau(\mathbf{0}) & = & 0 \\
I_\tau(P) & = & I(P)(\tau) \textit{ for state variables } P \\
I_\tau(\neg S) & = & 1 - I_\tau(S) \\
I_\tau(S_1 \vee S_2) & = & \max(I_\tau(S_1), I_\tau(S_2))
\end{array}$$

Given an interval $\sigma \in \mathbf{I}(T)$*, the values* $I_\sigma(t)$ *of terms* t *are defined by the clauses:*

$$\begin{array}{lcl}
I_\sigma(c) & = & I(c) \textit{ for constant symbols } c, \\
I_\sigma(x) & = & I(x) \textit{ for individual variables } x, \\
I_\sigma(v) & = & I(v)(\sigma) \textit{ for temporal variables } v, \\
I_\sigma(\int S) & = & \int\limits_{\min\sigma}^{\max\sigma} I_\tau(S)d\tau \textit{ for state expressions } S, \\
I_\sigma(f(t_1, \ldots, t_n)) & = & I(f)(I_\sigma(t_1), \ldots, I_\sigma(t_n)) \textit{ for } n\textit{-place function symbols } f.
\end{array}$$

To define $\int\limits_{\min\sigma}^{\max\sigma} I_\tau(S)d\tau$, *let* $n < \omega$ *and* $\sigma_1, \ldots, \sigma_n \in \mathbf{I}(\mathbf{T})$ *be such that* $\sigma = \sigma_1{}^\frown \ldots {}^\frown\sigma_n$, *and* $I_\tau(S)$ *is constant for* $\tau \in [\min\sigma_i, \max\sigma_i)$, $i = 1, \ldots, n$. *Then*

$$\int\limits_{\min\sigma}^{\max\sigma} I_\tau(S)d\tau = \sum_{i=1,\ldots,n,\ I_{\min\sigma_i}(S)=1} m(\sigma_i)\,.$$

Clearly, this value does not depend on the precise choice of $\sigma_1, \ldots, \sigma_n$.

The modelling relation $\models$ *is defined by the clauses:*

$\langle F, I\rangle, \sigma \not\models \bot$	
$\langle F, I\rangle, \sigma \models A$	*iff* $I(A)(\sigma) = 1$ *for temporal propositional letters* A
$\langle F, I\rangle, \sigma \models R(t_1, \ldots, t_n)$	*iff* $I(R)(I_\sigma(t_1), \ldots, I_\sigma(t_n)) = 1$
$\langle F, I\rangle, \sigma \models \neg\varphi$	*iff* $\langle F, I\rangle, \sigma \not\models \varphi$
$\langle F, I\rangle, \sigma \models (\varphi \vee \psi)$	*iff either* $\langle F, I\rangle, \sigma \models \varphi$ *or* $\langle F, I\rangle, \sigma \models \psi$
$\langle F, I\rangle, \sigma \models (\varphi{}^\frown\psi)$	*iff* $\langle F, I\rangle, \sigma_1 \models \varphi$ *and* $\langle F, I\rangle, \sigma_2 \models \psi$ *for some* $\sigma_1, \sigma_2 \in \mathbf{I}(T_F)$ *such that* $\sigma_1{}^\frown\sigma_2 = \sigma$
$\langle F, I\rangle, \sigma \models \exists x\varphi$	*iff* $\langle F, J\rangle, \sigma \models \varphi$ *for some* I *which* x*-agrees with* J

Sometimes it is convenient to work with the set of intervals which satisfy a formula. Let $\langle F, I\rangle$ be a *DC* model where $F = \langle\langle T, \leq\rangle, \langle D, +, 0\rangle, m\rangle$. Then we denote the set

$$\{\sigma \in \mathbf{I}(T) : \langle F, I\rangle, \sigma \models \varphi\}$$

by $\tilde{I}(\varphi)$. Let $X, Y \subseteq \mathbf{I}(T)$. Then $X{}^\frown Y$ stands for the set

$$\{\sigma_1{}^\frown\sigma_2 : \sigma_1 \in X, \sigma_2 \in Y, \max\sigma_1 = \min\sigma_2\}.$$

1.3 Abbreviations

The customary *infix* notation for $+$, $\leq$ and $=$ is used in *DC*. $\top$, $\wedge$, $\Rightarrow$ and $\Leftrightarrow$, $\forall$, $\neq$, $\geq$, $<$ and $>$ are used in the usual way. We assume that $^\frown$ binds less tightly than boolean connectives. Since $^\frown$ is associative, we omit parentheses in formulas with consecutive occurrences of $^\frown$. The following abbreviations are generally accepted in *DC*:

$\mathbf{1} \rightleftharpoons \neg\mathbf{0}$	
$\llbracket S \rrbracket \rightleftharpoons \int S = \ell \wedge \ell \neq 0$	
$\Diamond\varphi \rightleftharpoons \top{}^\frown\varphi{}^\frown\top$	(there is a subinterval for which φ holds)
$\Box\varphi \rightleftharpoons \neg\Diamond\neg\varphi$	(for all subintervals φ holds)
$\varphi^0 \rightleftharpoons \ell = 0$	
$\varphi^k \rightleftharpoons \underbrace{\varphi{}^\frown \ldots {}^\frown\varphi}_{k \text{ times}}$	for $k > 0$

The temporal variable ℓ is often introduced as an abbreviation or $\int \mathbf{1}$. As usual, we write nt for $\underbrace{t + \ldots + t}_{n \text{ times}}$.

1.4 Iteration and quantification over state in DC

DC is extended by iteration and quantification over state by allowing formulas of the forms φ^* and $\exists P\varphi$ where P is a state variable, respectively. The relation $\models$ is defined on such formulas by the clauses:

$\langle F, I\rangle, \sigma \models \varphi^*$ iff either $m(\sigma) = 0$, or there exist $n < \omega$ and $\sigma_1, \ldots, \sigma_n \in \mathbf{I}(T_F)$ such that $\sigma_1{}^\frown \ldots {}^\frown\sigma_n = \sigma$ and $\langle F, I\rangle, \sigma_i \models \varphi$ for $i = 1, \ldots, n$,

$\langle F, I\rangle, \sigma \models \exists P\varphi$ iff $\langle F, J\rangle, \sigma \models \varphi$ for some I which P-agrees with J.

Iteration * binds more tightly than $^\frown$ and the propositional connectives.

1.5 Proof systems for DC

Here we present the relatively complete proof system for DC from [HZ92] and the ω-complete system from [Gue98], because our new axioms and rule about iteration are supposed to work as additions to these systems. Next, we give the axioms and rules about quantification over state in HDC. We use them together with our axioms about iteration to prove the correctness of an explicit definition of iteration in HDC.

The Hilbert-style proof system for DC from [HZ92] includes a proof system for first order logic with equality (cf. e.g. [Sho67]), axioms and rules for ITL (cf. e.g. [Dut95a]) and some DC-specific axioms and rules ([HZ92]). We assume that the readers are familiar with Hilbert-style proof systems for first order logic. Here follow the ITL- and DC-specific axioms and rules.

1.5.1 Axioms and rules for ITL

$(A1)$ $(\varphi^\frown\psi) \wedge \neg(\chi^\frown\psi) \Rightarrow (\varphi \wedge \neg\chi^\frown\psi)$, $(\varphi^\frown\psi) \wedge \neg(\varphi^\frown\chi) \Rightarrow (\varphi^\frown\psi \wedge \neg\chi)$

$(A2)$ $((\varphi^\frown\psi)^\frown\chi) \Leftrightarrow (\varphi^\frown(\psi^\frown\chi))$

(R) $(\varphi^\frown\psi) \Rightarrow \varphi$, $(\psi^\frown\varphi) \Rightarrow \varphi$ if φ is rigid

(B) $(\exists x\varphi^\frown\psi) \Rightarrow \exists x(\varphi^\frown\psi)$, $(\psi^\frown\exists x\varphi) \Rightarrow \exists x(\psi^\frown\varphi)$ if $x \notin FV(\psi)$

$(L1)$ $(\ell = x^\frown\varphi) \Rightarrow \neg(\ell = x^\frown\neg\varphi)$, $(\varphi^\frown\ell = x) \Rightarrow \neg(\neg\varphi^\frown\ell = x)$

$(L2)$ $\ell = x + y \Leftrightarrow (\ell = x^\frown\ell = y)$

$(L3)$ $\varphi \Rightarrow (\ell = 0^\frown\varphi)$, $\varphi \Rightarrow (\varphi^\frown\ell = 0)$

(N) $\dfrac{\varphi}{\neg(\neg\varphi^\frown\psi)}$, $\dfrac{\varphi}{\neg(\psi^\frown\neg\varphi)}$

$(Mono)$ $\dfrac{\varphi \Rightarrow \psi}{(\varphi^\frown\chi) \Rightarrow (\psi^\frown\chi)}$, $\dfrac{\varphi \Rightarrow \psi}{(\chi^\frown\varphi) \Rightarrow (\chi^\frown\psi)}$

The presence of the modality $^\frown$ and flexible symbols in ITL brings a restriction on the use of first order logic rules and axioms which involve substitution: $[t/x]\varphi$ is allowed in proofs only if no variable in t becomes bound due to the substitution, and either t is rigid or $^\frown$ does not occur in φ. It is known that the above proof system for ITL is complete with respect to abstract-time models [Dut95a].

1.5.2 *DC* axioms and rules

$(DC1)$ $\int \mathbf{0} = 0$

$(DC2)$ $\int \mathbf{1} = \ell$

$(DC3)$ $\int S \geq 0$

$(DC4)$ $\int S_1 + \int S_2 = \int(S_1 \vee S_2) + \int(S_1 \wedge S_2)$

$(DC5)$ $(\int S = x \frown \int S = y) \Rightarrow \int S = x + y$

$(DC6)$ $\int S_1 = \int S_2$ if S_1 and S_2 are propositionally equivalent

$(IR1)$ $\dfrac{[\ell = 0/A]\varphi \;\; \varphi \Rightarrow [A \vee (A \frown \lceil S \rceil \vee \lceil \neg S \rceil)/A]\varphi}{[\top/A]\varphi}$

$(IR2)$ $\dfrac{[\ell = 0/A]\varphi \;\; \varphi \Rightarrow [A \vee (\lceil S \rceil \vee \lceil \neg S \rceil \frown A)/A]\varphi}{[\top/A]\varphi}$

(ω) $\dfrac{\Gamma \vdash (\lceil S \rceil \vee \lceil \neg S \rceil)^k \Rightarrow \varphi \text{ for all } k < \omega}{\Gamma \vdash \varphi}$

The extension of the proof system for *ITL* by the axioms *DC*1-*DC*6 and the rules *IR*1 and *IR*2 is complete with respect to the real time based frame relative to the class of *ITL* sentences which are valid in this frame [HZ92]. Replacing *IR*1 and *IR*2 by the infinitary rule ω leads to an ω-complete system for *DC* with respect to the class of the abstract time based models [Gue98], which are considered in this paper. The induction rules *IR*1 and *IR*2 are derivable in the system which consists of the *ITL* axioms and rules, *DC*1-*DC*6 and the infinitary rule ω. Despite that, *IR*1 and *IR*2 are preferable for practical purposes because they are finitary. Note that *DC*3 is redundant in our setting.

The rule ω is part of an ω-complete proof system, and ω-completeness is the equivalence between the consistency and the satisfiability of sets of formulas. Consistency means the impossibility to derive $\perp$ in the proof system from the given set of formulas, which may be infinite. Since *DC* is not a compact logic, derivability from infinite sets and derivability from a finite sets differ substantially. That is why we put down the rule ω in the form

$$\frac{\Gamma \vdash \ldots}{\Gamma \vdash \ldots},$$

where Γ stands for a set of formulas, which may as well be infinite. We assume that other proof rules included in the ω-complete proof system have this form too. For instance, we adopt the first order logic left $\exists$-introduction rule in the form

$(\exists_l)$ $\dfrac{\Gamma \vdash \varphi \Rightarrow \psi}{\Gamma \vdash \exists x \varphi \Rightarrow \psi}$, where $x \notin FV(\Gamma), FV(\psi)$.

Semantically, $\Gamma \vdash \varphi$ corresponds to $(\bigwedge \Gamma) \Rightarrow \varphi$, which may be impossible to write as a single *DC* formula for infinite Γ.

1.5.3 Axioms about quantification over state in *DC*

$(\exists_r^s)$ $[S/P]\varphi \Rightarrow \exists P \varphi$ where S is a state expression

(B^s) $(\exists P \varphi \frown \psi) \Rightarrow \exists P(\varphi \frown \psi)$, $(\psi \frown \exists P \varphi) \Rightarrow \exists P(\psi \frown \varphi)$ if $P \notin FV(\psi)$

$(\frown^s)$ $x = 0 \vee \ell \leq x \vee \exists P(\lceil S_1 \Leftrightarrow P \rceil \wedge \ell = x \frown \lceil S_2 \Leftrightarrow P \rceil)$

$(\exists_l^s)$ $\dfrac{\Gamma \vdash \varphi \Rightarrow \psi}{\Gamma \vdash \exists P \varphi \Rightarrow \psi}$, where $P \notin FV(\Gamma), FV(\psi)$.

The meanings of B^s, $\exists_r^s$ and $\exists_l^s$ are self-explanatory. The axiom $^\frown{}^s$ states that, given two states S_1 and S_2 and a chopping point which is internal to the reference interval, there is a state P which equals S_1 on the left of the chopping point and S_2 on the right. Recall here that our definition of finite variability implies that states are right-continuous.

1.5.4 ω-Completeness of DC and HDC

A set Γ of DC (HDC) formulas is called *consistent*, if $\bot$ cannot be derived from Γ, and theorems of the above proof system for DC (HDC) using only the rules MP and ω. Consistency in DC implies consistency in HDC, because it is a conservative extension of DC. The ω-completeness theorem for abstract-time DC is as follows:

Theorem 1 ([Gue98]) *Let Γ be a consistent set of DC formulas in some vocabulary $\mathbf{L}$. Then there exists an abstract model M for $\mathbf{L}$ and an interval σ in its time domain such that $M, \sigma \models \varphi$ for all $\varphi \in \Gamma$.*

The rule ω facilitates the use of maximal consistent sets of formulas in the proof of Theorem 1. We use such sets to prove our completeness results for DC^* and therefore the proof systems we consider are extensions of the system including ω. Theorem 1 applies to the proof system for HDC which includes the axioms from Section 1.5.3 too [Gue00c].

2 Axioms and a rule about iteration in DC

We combine the axioms DC^*1-DC^*4 and the rule DC^*5 below into three groups. All the groups include the axioms DC^*1 and DC^*2. Each group includes one of DC^*3, DC^*4 and DC^*5.
(DC^*1) $\quad \ell = 0 \Rightarrow \varphi^*$
(DC^*2) $\quad (\varphi^* {}^\frown \varphi) \Rightarrow \varphi^*$
The meanings of DC^*1 and DC^*2 are straightforward.
(DC^*3) $\quad (\varphi^* \wedge \psi^\frown \top) \Rightarrow (\psi \wedge \ell = 0^\frown \top) \vee (((\varphi^* \wedge \neg\psi^\frown \varphi) \wedge \psi)^\frown \top)$,
To understand the meaning of DC^*3, assume that some initial subinterval σ' of the reference interval σ satisfies ψ and φ^*. Then σ' can be represented as $\sigma_1'^\frown \ldots ^\frown \sigma_n'$ where $n < \omega$ and all the intervals $\pi_0 = [\min\sigma, \min\sigma] = [\min\sigma', \min\sigma']$, $\pi_k = \sigma_1'^\frown \ldots ^\frown \sigma_k'$, $k = 1, \ldots, n$, satisfy φ^*, and π_n satisfies ψ as well. Now depending on whether the least k such that π_k satisfies ψ is 0 or not, σ satisfies either the left or the right disjunctive member after $\Rightarrow$ in DC^*3.
(DC^*4) $\quad \Box(\ell = 0 \vee (\psi^\frown \varphi) \Rightarrow \psi) \Rightarrow (\varphi^* \Rightarrow \psi)$,
DC^*4 is an expression of the fact that $\tilde{I}(\varphi^*)$ is the least set of time intervals $X \subseteq \mathbf{I}(T)$ which satisfies the inclusion

$$\tilde{I}(\ell = 0) \cup \tilde{I}(\varphi)^\frown X \subseteq X.$$

To put down the rule DC^*5, we need the formula

$$\mathsf{f}(\varphi, Q) \rightleftharpoons \neg((\top^\frown \llbracket Q \rrbracket) \vee \ell = 0^\frown \llbracket \neg Q \rrbracket \wedge \neg\varphi^\frown(\llbracket Q \rrbracket^\frown \top) \vee \ell = 0).$$

This formula means that every maximal non-trivial subinterval of the reference interval which satisfies $\lceil\!\lceil \neg Q \rceil\!\rceil$ satisfies φ too. Let

$$\mathsf{g}(\varphi, P) \rightleftharpoons \mathsf{f}(\varphi, P) \wedge \mathsf{f}(\varphi, \neg P).$$

This formula means that all maximal non-trivial subintervals of the reference interval σ which satisfy either $\lceil\!\lceil P \rceil\!\rceil$ or $\lceil\!\lceil \neg P \rceil\!\rceil$ satisfy φ too. Since these intervals form a finite partition of σ, if σ satisfies $\mathsf{g}(\varphi, P)$, then it also satisfies φ^*. Here follows the rule itself:

$$(DC^*5) \quad \frac{\Gamma \vdash (\alpha^\frown \mathsf{g}(\varphi, P)^\frown \beta) \Rightarrow \psi}{\Gamma \vdash (\alpha^\frown \varphi^{*\frown} \beta) \Rightarrow \psi}, \text{ where } P \notin FV(\Gamma \cup \{\varphi, \psi, \alpha, \beta\}).$$

To understand DC^*5, consider the simpler rule

$$\frac{\Gamma \vdash \mathsf{g}(\varphi, P) \Rightarrow \psi}{\Gamma \vdash \varphi^* \Rightarrow \psi}, \text{ where } P \notin FV(\Gamma \cup \{\varphi, \psi\}),$$

which can be derived from the instance of DC^*5 with α and β both being $\ell = 0$. As mentioned above, if a reference interval σ satisfies $\mathsf{g}(\varphi, P)$, then the time points at which P changes its value inside σ partition it into subintervals which satisfy φ, and therefore σ itself satisfies φ^*. Given a *concrete* interpretation of P, the points at which P changes its value define a *concrete* finite partition of σ, whereas for σ to satisfy φ^* we just need the *existence* of such a partition. The role of the side condition $P \notin FV(\Gamma \cup \{\varphi, \psi\})$ is to handle this. This side condition implies that the validity of

$$\mathsf{g}(\varphi, P) \Rightarrow \psi \tag{4}$$

is equivalent to the validity of

$$\exists P \mathsf{g}(\varphi, P) \Rightarrow \psi \ . \tag{5}$$

Putting P in the scope of $\exists$ gives exactly what needed, because σ satisfies φ^* iff there *exists* a finite partition of σ into subintervals each of which satisfies φ, and a fresh state variable P can always be chosen to change its value exactly at the dividing points of such a finite partition. Hence the antecedent of (5) is equivalent to φ^*. By putting down a proof rule instead of the equivalence

$$\exists P \mathsf{g}(\varphi, P) \Leftrightarrow \varphi^* \ , \tag{6}$$

we axiomatise iteration in DC without resorting to the state variable binding quantifier. Below we show how (6) can be proved in the system for HDC from Section 1 using the axioms DC^*1, DC^*2 and DC^*4 about iteration.

Note that the scope of the soundness of $DC^*1 - DC^*4$ is, in fact, the extension ITL^* of ITL by iteration, because these axioms involve no DC-specific constructs.

3 Completeness of DC^*1, DC^*2 and DC^*5 for iteration in abstract-time DC^*

The ω-completeness of the proof system for abstract-time DC which includes the axioms and the rule ω from Section 1.5.2 can be proved following the example from [Gue98], which on its turn builds on [Dut95a]. In this section we prove that adding DC^*1, DC^*2 and DC^*5 to this system leads to an ω-complete proof system for abstract-time DC^*. The proof involves a Lindenbaum lemma to extend the given consistent set of formulas Γ_0 to a maximal consistent set Γ with appropriately chosen witnesses, the construction of a canonical model M from Γ and a truth lemma to prove that a distinguished interval in M satisfies all the formulas from Γ and, consequently, from Γ_0. The presence of iteration and DC^*1, DC^*2 and DC^*5 affects mostly the Lindenbaum lemma and the truth lemma. The rest of the completeness proof is much like in [Gue98] and therefore we skip most of the details.

3.1 Maximal consistent sets and the Lindenbaum lemma for DC^*

The use of a maximal consistent set of formulas in the completeness proof for a quantified logic involves *Henkin constants* also known as *witnesses* for the existential formulas in the set. In the case of first order predicate logic the only existential formulas are those in which the quantifier binds an individual variable. A constant c is called a *witness* for the existential formula $\exists x\varphi$ in a set of formulas Γ if $\exists x\varphi \Rightarrow [c/x]\varphi \in \Gamma$. Along with the first order quantifier $\exists$, DC^* implicitly involves two more kinds of existential formulas. The finite variability requirement on state means that DC models validate the formula $(\exists k < \omega)(\lceil S\rceil \vee \lceil\neg S\rceil)^k$ for every state expression S. The definition (6) of iteration in HDC involves the quantifier prefix $\exists P$. We use maximal consistent sets which have witnesses for such formulas, despite that they do not occur in DC^* languages explicitly. These witnesses are freely occurring state variables and therefore can be regarded as constants, and formulas of the form $(\lceil S\rceil \vee \lceil\neg S\rceil)^k$ for concrete k, respectively.

Definition 10 *Let C be a set of constants and state variables. A set of DC^* formulas Γ written in some vocabulary* $\mathbf{L}$ has witnesses in *C if for every existential formula $\exists x\varphi \in \Gamma$ there is a constant $c \in C$ such that $[c/x]\varphi \in \Gamma$, for every state expression S written in $\mathbf{L}$ there is a $k < \omega$ such that $(\lceil S\rceil \vee \lceil\neg S\rceil)^k \in \Gamma$ and for every formula of the form $(\alpha^\frown\varphi^{*\frown}\beta) \in \Gamma$ there is a state variable P such that $(\alpha^\frown \mathrm{g}(\varphi, P)^\frown\beta) \in \Gamma$.*

Lemma 1 (Lindenbaum lemma) *Let $\mathbf{L}$ be a countable DC vocabulary and Γ_0 be a consistent set of DC^* formulas in this vocabulary. Let a set C consist of countably many constant symbols and countably many state variables such that $C \cap \mathbf{L} = \emptyset$. Then there exists a maximal consistent set Γ of DC^* formulas in the vocabulary $\mathbf{L} \cup C$ which has witnesses in C.*

Proof: This proof follows a general pattern known from numerous modal logics.

Let the set of all the DC^* formulas written in the vocabulary $\mathbf{L} \cup C$ be $\{\varphi_i : i < \omega\}$ and the set of all the state expressions from $\mathbf{L} \cup C$ be $\{S_j : j < \omega\}$. We define the sequence Γ_k, $k < \omega$ of consistent sets of such formulas by induction on k. Γ_0 is as given in the theorem. Since Γ_0 consists of formulas in $\mathbf{L}$ and we obtain Γ_{k+1} by adding finitely many formulas written in $\mathbf{L} \cup C$ to Γ_k for each k, we can assume that there are both constants and state variables in C which do not occur in Γ_k for all k. To define the sequence Γ_k, $k < \omega$, we use a pair of functions $\pi_1, \pi_2 : \omega \to \omega$ such that for every pair $i, j < \omega$ there is a $k < \omega$ satisfying $\pi_1(k) = i$ and $\pi_2(k) = j$. Here follow the definition of Γ_k for $k > 0$:

Assume that Γ_k has been defined for some $k < \omega$ and let $i = \pi_1(k)$, $j = \pi_2(k)$. If $\Gamma_k \cup \{\varphi_i\}$ is not consistent, then $\Gamma_{k+1} = \Gamma_k$. Otherwise we consider the following cases:

1. φ_i is the existential formula $\exists x\psi$. Then we choose a constant $c \in C$ such that c does not occur in the formulas from Γ_k and put $\Gamma_{k+1} = \Gamma_k \cup \{\varphi_i, [c/x]\psi\}$. Assume that Γ_{k+1} is not consistent for the sake of contradiction. Then we have $\Gamma_k, \exists x\psi \vdash \neg[c/x]\psi$. Since c occurs neither in Γ_k, nor in ψ, replacing it by a fresh individual variable y will give $\Gamma_k, \exists x\psi \vdash \neg[y/x]\psi$ again. This, by an application of $\exists_l$, will bring $\Gamma_k, \exists x\psi \vdash \forall y \neg[y/x]\psi$, which is a contradiction.

2. φ_i is $(\alpha^\frown\psi^{*\frown}\beta)$. Then we choose a state variable $P \in C$ which does not occur in $\Gamma_k \cup \{\varphi_i\}$ and put $\Gamma_{k+1} = \Gamma_k \cup \{\varphi_i, (\alpha^\frown \mathsf{g}(\psi, P)^\frown\beta)\}$. Assuming that $\Gamma_k, \varphi_i \vdash (\alpha^\frown \mathsf{g}(\psi, P)^\frown\beta) \Rightarrow \bot$ brings a contradiction by an application of the rule DC^*5. Hence Γ_{k+1} is consistent.

3. φ_i is $\top$. Then we choose a $l < \omega$ such that $\Gamma_k \cup \{(\llbracket S_j \rrbracket \vee \llbracket \neg S_j \rrbracket)^l\}$ is consistent and put $\Gamma_{k+1} = \Gamma_k \cup \{(\llbracket S_j \rrbracket \vee \llbracket \neg S_j \rrbracket)^l\}$. Such an l exists because assuming that $\Gamma_k \vdash \neg(\llbracket S_j \rrbracket \vee \llbracket \neg S_j \rrbracket)^l$ for all $l < \omega$ implies that Γ_k itself is inconsistent by an application of the infinitary rule ω.

4. φ_i is of none of the above forms. Then we put $\Gamma_{k+1} = \Gamma_k \cup \{\varphi_i\}$.

Now let us prove that the union $\Gamma = \bigcup\limits_{k<\omega} \Gamma_k$ is consistent, has witnesses in $\mathbf{L} \cup C$ and either $\varphi \in \Gamma$ or $\neg\varphi \in \Gamma$ for every formula φ written in $\mathbf{L} \cup C$, which implies that Γ is maximal.

To prove that Γ is maximal, assume that φ is a formula in $\mathbf{L} \cup C$ and $\varphi, \neg\varphi \notin \Gamma$ for the sake of contradiction. Let $k_1, k_2 < \omega$ be such that φ is $\varphi_{\pi_1(k_1)}$ and $\neg\varphi$ is $\varphi_{\pi_1(k_2)}$. Then, by the definition of the sets Γ_k, $\Gamma_{\max k_1, k_2}$ is inconsistent with both φ and $\neg\varphi$, which is a contradiction.

The cases 1-3 of the inductive definition of the sequence Γ_k clearly imply that Γ has witnesses in C.

Since $\bot \notin \Gamma_k$ for all k, $\bot \notin \Gamma$ as well. Hence, to establish the consistency of Γ, we just need to prove that Γ contains all the theorems of our proof system written in the vocabulary $\mathbf{L} \cup C$ and is closed under the rules MP and ω. The only non-trivial part of this proof is about the closedness under ω. Let $(\llbracket S \rrbracket \vee \llbracket \neg S \rrbracket)^l \Rightarrow \varphi \in \Gamma$ for all $l < \omega$. Let S be S_j, $k < \omega$ be such that $\pi_2(k) = j$ and $\varphi_{\pi_1(k)}$ is $\top$. Then there is an $l < \omega$ such that $(\llbracket S_j \rrbracket \vee \llbracket \neg S_j \rrbracket)^l \in \Gamma_{k+1}$ by the definition of this set, whence $\varphi \in \Gamma$. □

We intend to use the set Γ whose existence was shown in Lemma 1 to construct a model M for the vocabulary $\mathbf{L} \cup C$ so that M satisfies the formulas from Γ at a distinguished interval σ. To define the interpretations of the symbols from $\mathbf{L} \cup C$ at the subintervals of σ, we intend to use sets of formulas of the form

$$\{\varphi : ((\ell = c' {}^\frown \varphi) \wedge \ell = c'' {}^\frown \top) \in \Gamma\} \tag{7}$$

for appropriately chosen constants $c', c'' \in C$.

Lemma 2 *Let Γ be a maximal consistent set of formulas written in the vocabulary $\mathbf{L} \cup C$ and c' and c'' be constants from C such that $c' \leq c''$ and $c'' \leq \ell \in \Gamma$. Then (7) is a maximal consistent set in the vocabulary $\mathbf{L} \cup C$ which has witnesses in C.*

Proof: Let us denote the set (7) by Δ. For every formula φ either $((\ell = c' {}^\frown \varphi) \wedge \ell = c'' {}^\frown \top) \in \Gamma$ or $((\ell = c' {}^\frown \neg\varphi) \wedge \ell = c'' {}^\frown \top) \in \Gamma$. Hence for every φ either $\varphi \in \Delta$ or $\neg\varphi \in \Delta$ and therefore Δ is maximal. To realise that Δ has witnesses in C, note that $\exists x \varphi \in \Delta$ is equivalent to $\exists x((\ell = c' {}^\frown \varphi) \wedge \ell = c'' {}^\frown \top) \in \Gamma$, $(\alpha {}^\frown \varphi^* {}^\frown \beta) \in \Delta$ is equivalent to $((\ell = c' {}^\frown \alpha) {}^\frown \varphi^* {}^\frown (\beta {}^\frown \ell = c''')) \in \Gamma$ for some $c''' \in C$ such that $c'' + c''' = \ell \in \Gamma$, and $(\lceil S \rceil \vee \lceil \neg S \rceil)^l \in \Gamma$ implies $(\lceil S \rceil \vee \lceil \neg S \rceil)^{l'} \in \Delta$ for some $l' \leq l$. $\bot \notin \Delta$ and therefore the consistency of Δ follows from its closedness under the proof rules, which is established like in the proof of Lemma 1. $\Box$

3.2 The canonical construction for abstract-time DC^*

Let $\mathbf{L}$ and C be as in the previous section and Γ be a maximal consistent set of formulas in the vocabulary $\mathbf{L} \cup C$ with witnesses in C. Let

$$c_1 \equiv c_2 \text{ iff } c_1 = c_2 \in \Gamma$$

for constants $c_1, c_2 \in C$. Clearly, $\equiv$ is an equivalence relation on the constants from C. Let $[c]$ denote the $\equiv$-equivalence class which contains c for each constant $c \in C$. Let D be the set

$$\{[c] : c \text{ is a constant in } C\}.$$

Let T be the set

$$\{[c] : c \in C, c \leq \ell \in \Gamma\}.$$

Let

$$[c'] \leq [c''] \text{ iff } c' \leq c'' \in \Gamma.$$

Clearly, $\leq$ is a linear ordering on T and $\langle T, \leq \rangle$ is a time domain. Let the mapping I be defined on the vocabulary $\mathbf{L} \cup C$ by the clauses:

1. $I(x), I(d) \in D$ for individual variables x and constants d, and

$$I(x) = \{c \in C : c = x \in \Gamma\},\ I(d) = \{c \in C : c = d \in \Gamma\}.$$

2. $I(f) : D^n \to D$ for n-ary function symbols f and

$$I(f)([c_1], \dots, [c_n]) = \{c \in C : c = f(c_1, \dots, c_n) \in \Gamma\}.$$

3. $I(R) : D^n \to D$ for n-ary relation symbols R and

$$I(R)([c_1], \dots, [c_n]) = 1 \text{ iff } R(c_1, \dots, c_n) \in \Gamma.$$

4. $I(v) : \mathbf{I}(T) \to D$ for temporal variables v and

$$I(v)([[c'], [c'']]) = \{c \in C : ((\ell = c' \frown v = c) \wedge \ell = c'' \frown \top) \in \Gamma\}.$$

5. $I(A) : \mathbf{I}(T) \to \{0, 1\}$ for temporal propositional letters A and

$$I(A)([[c'], [c'']]) = 1 \text{ iff } ((\ell = c' \frown A) \wedge \ell = c'' \frown \top) \in \Gamma.$$

6. $I(P) : (T) \to \{0, 1\}$ for state variables P and

$$I(P)([c']) = 1 \text{ iff } (\ell = c' \frown \lceil P \rceil \frown \top) \in \Gamma.$$

A lengthy but otherwise straighforward argument, which is standard for canonical models, shows that the above definitions are correct, $\langle D, I(0), I(+)\rangle$ is a duration domain, $I(\ell)$ is a measure function from $\mathbf{I}(T)$ to D, $F = \langle\langle T, \leq\rangle, \langle D, I(0), I(+)\rangle, I(\ell)\rangle$ is an abstract DC frame and I is a DC interpretation of $\mathbf{L} \cup C$ into F, which means that $M = \langle F, I\rangle$ is an abstract DC model for $\mathbf{L} \cup C$. What makes M relevant is the following lemma.

Lemma 3 (Truth lemma) *Let $\sigma \in \mathbf{I}(T)$ and $\sigma = [[c'], [c'']]$ for some $c', c'' \in C$. Let $\Delta = \{\varphi : ((\ell = c' \frown \varphi) \wedge \ell = c'' \frown \top) \in \Gamma\}$. Then*

$$I_\sigma(t) = \{c \in C : t = c \in \Delta\} \text{ and } M, \sigma \models \varphi \text{ iff } \varphi \in \Delta$$

for every term t and every formula φ written in the vocabulary $\mathbf{L} \cup C$.

Proof: The proof about terms is by induction on their construction. The proof about formulas is by induction on the lexicographical ordering of the pairs $\langle |\varphi|^*, |\varphi|\rangle$, where $|\varphi|$ denotes the length of φ and $|\varphi|^* = \max\{|\psi| : \psi^* \text{ is a subformula of } \varphi\}$. Since we are focussing on iteration in this paper, we do only the induction step about φ of the form ψ^*.

Let $\psi^* \in \Delta$. Δ has witnesses in C by Lemma 2. This implies that there is a state variable $P \in C$ such that $\mathsf{g}(\psi, P) \in \Delta$ and there is an $l < \omega$ such that $(\lceil P \rceil \vee \lceil \neg P \rceil)^k \in \Delta$. Let k be chosen to be the least one with this property. Then we have $(\lceil Q_1 \rceil \frown \dots \frown \lceil Q_k \rceil) \in \Delta$ where $Q_1, \dots, Q_k \in \{P, \neg P\}$ and Q_i is P iff Q_{i-1} is $\neg P$ for $i \geq 2$. Together with $\mathsf{g}(\psi, P) \in \Delta$, this implies that $\psi^k \in \Delta$, which, by the induction hypothesis, implies that $M, \sigma \models \psi^k$ and, consequently, $M, \sigma \models \psi^*$.

Now let $M, \sigma \models \psi^*$. Then $M, \sigma \models \psi^k$ for some $k < \omega$, which implies that $(\ell = 0 \frown \psi^k) \in \Delta$ for that k. This implies $\psi^* \in \Delta$ by one application of DC^*1 and k applications of DC^*2. $\square$

3.3 The completeness theorem

Now we are ready to prove the ω-completeness of our proof system for abstract-time DC^*.

Theorem 2 *Let Γ_0 be a consistent set of DC^* formulas in the DC vocabulary $\mathbf{L}$. Then there is an abstract DC model M_0 for $\mathbf{L}$ and an interval σ in the time domain of M_0 such that $M_0, \sigma \models \varphi$ for all $\varphi \in \Gamma_0$.*

Proof: According to Lemma 1, Γ_0 can be extended to a maximal consistent set Γ of formulas in the extension $\mathbf{L} \cup C$ by a set C consisting of countably many fresh individual constants and countably many fresh state variables. The set Γ can be constructed to have witnesses in C. Now consider the model $M = \langle F, I \rangle$ which was constructed from such a set Γ in Section 3.2. Let $c', c'' \in C$ satisfy the conditions $c' = 0, c'' = \ell \in \Gamma$. Then $c' \leq c'' \in \Gamma$ and the set (7) is Γ itself. Then Lemma 3 implies that $M, [[c'], [c'']] \models \varphi$ for all $\varphi \in \Gamma$ and, in particular, for all $\varphi \in \Gamma_0$. To obtain M_0 from M, one only needs to replace I by its restriction to the vocabulary $\mathbf{L}$. $\Box$

4 The interderivability between DC^*3, DC^*4 and DC^*5

As seen in the previous section, the proof rule DC^*5 is implicitly related to the state variable binding existential quantifier in DC^*. The key ingredient in this rule is the formula $\mathsf{g}(\varphi, P)$. In this section we first show that in *HDC*, which includes this quantifier, one can prove the correctness of an explicit definition for iteration involving this formula by means of the *HDC*-specific axioms and rules from Section 1.5.3 and the group of axioms about iteration which consists of DC^*1, DC^*2 and DC^*4. This means that DC^*5 can be regarded as a derived rule in the extension of the proof system for *HDC* by the axioms DC^*1, DC^*2 and DC^*4 about iteration. We also prove that DC^*4 is derivable from DC^*3 in the proof system for DC without quantification over state. Furthermore, we prove that DC^*3 and DC^*4 are derivable in the the proof system for DC which has the axioms DC^*1, DC^*2 and the rule DC^*5 about iteration. Taken apart from the other axioms about iteration, DC^*4 is weaker than DC^*3. 5dvh change We give no proof of DC^*3 in a system with DC^*4 being as the induction axiom.

In the deductions below we give in detail only the steps which involve iteration- and state-variable-quantifier-specific axioms and rules. We skip the detail on other steps and mark them by "*DC*" to indicate that they can be made in *DC* with neither iteration, nor state variable quantifier.

Proposition 1 *Let $P \notin FV(\varphi)$. Then*

$$\exists P \mathsf{g}(\varphi, P) \Leftrightarrow \varphi^*$$

is provable in the extension of the proof system for HDC by the axioms DC^*1, DC^*2 and DC^*4, and also in the extension of this proof system by DC^*1, DC^*2 and DC^*3

Proof: Here follow deductions of $\varphi^* \Rightarrow \exists P\mathsf{g}(\varphi,P)$ and $\exists P\mathsf{g}(\varphi,P) \Rightarrow \varphi^*$ involving DC^*4.

$\varphi^* \Rightarrow \exists P\mathsf{g}(\varphi,P)$:

1	$(\mathsf{g}(\varphi,P)^\frown\varphi) \Rightarrow$	
	$\Rightarrow \mathsf{g}(\varphi,P) \vee \varphi \vee \exists x(x \neq 0 \wedge x < \ell \wedge (\mathsf{g}(\varphi,P) \wedge \ell = x^\frown\varphi))$	DC
2	$\varphi \Rightarrow \ell = 0 \vee (\lceil\!\lceil \mathbf{1} \rceil\!\rceil \wedge \varphi)$	DC
3	$\ell = 0 \Rightarrow \mathsf{g}(\varphi,P)$	DC
4	$\lceil\!\lceil \mathbf{1} \rceil\!\rceil \wedge \varphi \Rightarrow \exists P(\lceil\!\lceil P \rceil\!\rceil \wedge \varphi)$	$\exists^s$
5	$\lceil\!\lceil P \rceil\!\rceil \wedge \varphi \Rightarrow \mathsf{g}(\varphi,P)$	DC
6	$\exists P(\lceil\!\lceil P \rceil\!\rceil \wedge \varphi) \Rightarrow \exists P\mathsf{g}(\varphi,P)$	5, $\exists^s_r$, $\exists^s_l$
7	$\varphi \Rightarrow \mathsf{g}(\varphi,P) \vee \exists P\mathsf{g}(\varphi,P)$	2-4, 6, DC
8	$x \neq 0 \wedge x < \ell \Rightarrow \exists Q(\ell = x \wedge \lceil\!\lceil Q \Leftrightarrow P \rceil\!\rceil^\frown\lceil\!\lceil Q \Leftrightarrow \mathbf{0} \rceil\!\rceil)$	${}^{\frown s}$
9	$x \neq 0 \wedge x < \ell \Rightarrow \exists Q(\ell = x \wedge \lceil\!\lceil Q \Leftrightarrow P \rceil\!\rceil^\frown\lceil\!\lceil Q \Leftrightarrow \mathbf{1} \rceil\!\rceil)$	${}^{\frown s}$
10	$x \neq \ell \wedge (\mathsf{g}(\varphi,P) \wedge \ell = x^\frown\varphi) \Rightarrow$	
	$\left(\begin{array}{l} (\mathsf{g}(\varphi,P) \wedge (\top^\frown\lceil\!\lceil P \rceil\!\rceil) \wedge \ell = x^\frown\varphi) \vee \\ (\mathsf{g}(\varphi,P) \wedge (\top^\frown\lceil\!\lceil \neg P \rceil\!\rceil) \wedge \ell = x^\frown\varphi) \end{array}\right)$	DC
11	$\left(\begin{array}{l} (\ell = x \wedge \lceil\!\lceil Q \Leftrightarrow P \rceil\!\rceil^\frown\lceil\!\lceil Q \Leftrightarrow \mathbf{0} \rceil\!\rceil) \wedge \\ \mathsf{g}(\varphi,P) \wedge (\top^\frown\lceil\!\lceil P \rceil\!\rceil) \wedge \ell = x^\frown\varphi) \end{array}\right) \Rightarrow$	
	$\Rightarrow (\mathsf{g}(\varphi,Q) \wedge (\top^\frown\lceil\!\lceil Q \rceil\!\rceil)^\frown\varphi \wedge \lceil\!\lceil \neg Q \rceil\!\rceil)$	DC
12	$(\mathsf{g}(\varphi,Q) \wedge (\top^\frown\lceil\!\lceil Q \rceil\!\rceil)^\frown\varphi \wedge \lceil\!\lceil \neg Q \rceil\!\rceil) \Rightarrow \mathsf{g}(\varphi,Q)$	DC
13	$\left(\begin{array}{l} (\ell = x \wedge \lceil\!\lceil Q \Leftrightarrow P \rceil\!\rceil^\frown\lceil\!\lceil Q \Leftrightarrow \mathbf{0} \rceil\!\rceil) \wedge \\ \mathsf{g}(\varphi,P) \wedge (\top^\frown\lceil\!\lceil P \rceil\!\rceil) \wedge \ell = x^\frown\varphi) \end{array}\right) \Rightarrow \exists Q\mathsf{g}(\varphi,Q)$	11, 12, DC, $\exists^s_r$
14	$\exists Q(\ell = x \wedge \lceil\!\lceil Q \Leftrightarrow P \rceil\!\rceil^\frown\lceil\!\lceil Q \Leftrightarrow \mathbf{0} \rceil\!\rceil) \Rightarrow$	
	$\Rightarrow ((\mathsf{g}(\varphi,P) \wedge (\top^\frown\lceil\!\lceil P \rceil\!\rceil) \wedge \ell = x^\frown\varphi) \Rightarrow \exists Q\mathsf{g}(\varphi,Q))$	13, DC
15	$x \neq 0 \wedge x < \ell \wedge (\mathsf{g}(\varphi,P) \wedge (\top^\frown\lceil\!\lceil P \rceil\!\rceil) \wedge \ell = x^\frown\varphi) \Rightarrow$	
	$\Rightarrow \exists Q\mathsf{g}(\varphi,Q)$	8, 14, DC
16	$x \neq 0 \wedge x < \ell \wedge (\mathsf{g}(\varphi,P) \wedge (\top^\frown\lceil\!\lceil \neg P \rceil\!\rceil) \wedge \ell = x^\frown\varphi) \Rightarrow$	like 15, but
	$\Rightarrow \exists Q\mathsf{g}(\varphi,Q)$	using 9 instead of 8
17	$x \neq 0 \wedge x < \ell \wedge (\mathsf{g}(\varphi,P) \wedge \ell = x^\frown\varphi) \Rightarrow \exists Q\mathsf{g}(\varphi,Q)$	10, 15, 16, DC
18	$\exists x(x \neq 0 \wedge x < \ell \wedge (\mathsf{g}(\varphi,P) \wedge \ell = x^\frown\varphi)) \Rightarrow \exists Q\mathsf{g}(\varphi,Q)$	17, DC
19	$(\mathsf{g}(\varphi,P)^\frown\varphi) \Rightarrow \mathsf{g}(\varphi,P) \vee \exists P\mathsf{g}(\varphi,P) \vee \exists Q\mathsf{g}(\varphi,Q)$	1, 7, 18, DC
20	$\exists Q\mathsf{g}(\varphi,Q) \Rightarrow \exists P\mathsf{g}(\varphi,P)$	$\exists^s_l$c ,$\exists^s_r$, DC
21	$\mathsf{g}(\varphi,P) \Rightarrow \exists P\mathsf{g}(\varphi,P)$	$\exists^s_r$
22	$(\mathsf{g}(\varphi,P)^\frown\varphi) \Rightarrow \exists P\mathsf{g}(\varphi,P)$	19, 20, 21, DC
23	$\exists P(\mathsf{g}(\varphi,P)^\frown\varphi) \Rightarrow \exists P\mathsf{g}(\varphi,P)$	22, $\exists^s_l$
24	$(\exists P\mathsf{g}(\varphi,P)^\frown\varphi) \Rightarrow \exists P(\mathsf{g}(\varphi,P)^\frown\varphi)$	B^s
25	$(\exists P\mathsf{g}(\varphi,P)^\frown\varphi) \Rightarrow \exists P\mathsf{g}(\varphi,P)$	23, 24, DC
26	$\ell = 0 \Rightarrow \exists P\mathsf{g}(\varphi,P)$	3, $\exists^s_r$, DC
27	$\ell = 0 \vee (\exists P\mathsf{g}(\varphi,P)^\frown\varphi) \Rightarrow \exists P\mathsf{g}(\varphi,P)$	25, 26, DC
28	$\Box(\ell = 0 \vee (\exists P\mathsf{g}(\varphi,P)^\frown\varphi) \Rightarrow \exists P\mathsf{g}(\varphi,P)$	27, DC
29	$\varphi^* \Rightarrow \exists P\mathsf{g}(\varphi,P)$	28, DC^*4, DC

$\exists P\mathsf{g}(\varphi, P) \Rightarrow \varphi^*$:

1	$\mathsf{g}(\varphi, P) \wedge (\lceil P \rceil \vee \lceil \neg P \rceil)^k \Rightarrow \bigvee\limits_{l \leq k} \varphi^l$	$k < \omega$, DC
2	$\varphi^l \Leftrightarrow (\ell = 0 \frown \varphi^l)$	$l < \omega$, DC
3	$(\ell = 0 \frown \varphi^l) \Rightarrow \varphi^*$	$l < \omega$, DC^*1, DC^*2, DC
4	$(\lceil P \rceil \vee \lceil \neg P \rceil)^k \Rightarrow (\mathsf{g}(\varphi, P) \Rightarrow \varphi^*)$	$k < \omega$, 1-3, DC
5	$\mathsf{g}(\varphi, P) \Rightarrow \varphi^*$	4, ω
6	$\exists P\mathsf{g}(\varphi, P) \Rightarrow \varphi^*$	5, $\exists^s_l$

A deduction of $\varphi^* \Rightarrow \exists P\mathsf{g}(\varphi, P)$ can be obtained using the instance of DC^*3 with ψ being $\neg\exists\mathsf{g}(\varphi, P)$. We skip it here.

□

Proposition 2 *DC^*4 is provable in the extension of the proof system for DC by just DC^*3.*

Proof: Here follows a deduction of DC^*4 involving DC^*3:

1	$\varphi^* \wedge \neg\psi \Rightarrow (\varphi^* \wedge \neg\psi \frown \top)$	DC
2	$(\varphi^* \wedge \neg\psi \frown \top) \Rightarrow (\ell = 0 \wedge \neg\psi \frown \top) \vee ((\varphi^* \wedge \psi \frown \varphi) \wedge \neg\psi \frown \top)$	DC^*3
3	$\Box(\ell = 0 \vee (\psi \frown \varphi) \Rightarrow \psi) \Rightarrow \neg(\ell = 0 \wedge \neg\psi \frown \top)$	DC
4	$((\varphi^* \wedge \psi \frown \varphi) \wedge \neg\psi \frown \top) \Rightarrow ((\psi \frown \varphi) \wedge \neg\psi \frown \top)$	DC
5	$\Box(\ell = 0 \vee (\psi \frown \varphi) \Rightarrow \psi) \Rightarrow \neg((\psi \frown \varphi) \wedge \neg\psi \frown \top)$	DC
6	$\Box(\ell = 0 \vee (\psi \frown \varphi) \Rightarrow \psi) \Rightarrow (\varphi^* \Rightarrow \psi)$	1-5, DC

□

Proposition 3 *DC^*3 and DC^*4 is provable in the extension of the proof system for DC by the axioms DC^*1, DC^*2 and DC^*5.*

Proof: Here follows a deduction of DC^*3 involving DC^*5:

1	$(\lceil P \rceil \vee \lceil \neg P \rceil)^k \wedge \mathsf{g}(\varphi, P) \Rightarrow \bigvee\limits_{l \leq k} \varphi^l$	$k < \omega$, DC
2	$\neg(\ell = 0 \wedge \psi \frown \top) \wedge \varphi^l \Rightarrow (\varphi^* \wedge \neg\psi \frown \varphi^l)$	$l < \omega$, DC^*1, DC
3	$\neg(\neg((\varphi^* \wedge \neg\psi \frown \varphi) \Rightarrow \neg\psi) \frown \top) \wedge (\varphi^* \wedge \neg\psi \frown \varphi^{m+1}) \Rightarrow (\varphi^* \wedge \neg\psi \frown \varphi^m)$	$m < \omega$, DC^*2, DC^*
4	$\neg(\ell = 0 \wedge \psi \frown \top) \wedge \varphi^l \wedge \neg(\neg((\varphi^* \wedge \neg\psi \frown \varphi) \Rightarrow \neg\psi) \frown \top) \wedge \Rightarrow \neg\psi$	$l < \omega$, 2, 3, DC
5	$\left(\begin{array}{l} (\lceil P \rceil \vee \lceil \neg P \rceil)^k \wedge \mathsf{g}(\varphi, P) \wedge \neg(\ell = 0 \wedge \psi \frown \top) \wedge \\ \neg(\neg((\varphi^* \wedge \neg\psi \frown \varphi) \Rightarrow \neg\psi) \frown \top) \end{array}\right) \Rightarrow \neg\psi$	$k < \omega$, 1, 4, DC
6	$\mathsf{g}(\varphi, P) \wedge \neg(\ell = 0 \wedge \psi \frown \top) \wedge \neg(\neg((\varphi^* \wedge \neg\psi \frown \varphi) \Rightarrow \neg\psi) \frown \top) \Rightarrow \neg\psi$	5, ω
7	$\neg(\ell = 0 \wedge \psi \frown \top) \wedge \neg(\neg((\varphi^* \wedge \neg\psi \frown \varphi) \Rightarrow \neg\psi) \frown \top) \Rightarrow (\varphi^* \Rightarrow \neg\psi)$	6, DC^*5
8	$\varphi^* \wedge \psi \Rightarrow (\ell = 0 \wedge \psi \frown \top) \vee ((\varphi^* \wedge \neg\psi \frown \varphi) \wedge \psi \frown \top)$	7, DC
9	$(\varphi^* \wedge \psi \frown \top) \Rightarrow (\ell = 0 \wedge \psi \frown \top \frown \top) \vee ((\varphi^* \wedge \neg\psi \frown \varphi) \wedge \psi \frown \top \frown \top)$	8, DC
10	$(\varphi^* \wedge \psi \frown \top) \Rightarrow (\ell = 0 \wedge \psi \frown \top) \vee ((\varphi^* \wedge \neg\psi \frown \varphi) \wedge \psi \frown \top)$	9, DC

Here follows a deduction of DC^*4 involving DC^*5:

1	$(\lceil P \rceil \vee \lceil \neg P \rceil)^k \wedge \mathtt{g}(\varphi, P) \Rightarrow \bigvee_{l \leq k} \varphi^l$	$k < \omega$, DC
2	$\Box(\ell = 0 \vee (\psi \frown \varphi) \Rightarrow \psi) \wedge \varphi^l \Rightarrow (\psi \frown \varphi^l)$	$l < \omega$, DC
3	$\Box(\ell = 0 \vee (\psi, \varphi) \Rightarrow \psi) \wedge (\psi \frown \varphi^{m+1}) \Rightarrow (\psi \frown \varphi^m)$	$m < \omega$, DC
4	$\Box(\ell = 0 \vee (\psi \frown \varphi) \Rightarrow \psi) \wedge (\psi \frown \varphi^l) \Rightarrow \psi$	$l < \omega$, 3, DC
5	$(\lceil P \rceil \vee \lceil \neg P \rceil)^k \Rightarrow (\mathtt{g}(\varphi, P) \Rightarrow (\Box(\ell = 0 \vee (\psi \frown \varphi) \Rightarrow \psi) \Rightarrow \psi))$	$k < \omega$, 1, 2, 4, DC
6	$\mathtt{g}(\varphi, P) \Rightarrow (\Box(\ell = 0 \vee (\psi \frown \varphi) \Rightarrow \psi) \Rightarrow \psi)$	5, ω
7	$\varphi^* \Rightarrow (\Box(\ell = 0 \vee (\psi \frown \varphi) \Rightarrow \psi) \Rightarrow \psi)$	6, DC^*5
8	$\Box(\ell = 0 \vee (\psi \frown \varphi) \Rightarrow \psi) \Rightarrow (\varphi^* \Rightarrow \psi)$	7, DC

$\Box$

5 DC^*3 as a translation of a PDL induction axiom

The origins of the axiom DC^*4 and the rule DC^*5 were given in the introduction. In this section we point to a certain degree of semantical compatibility between the models for *ITL* and the models for propositional dynamic logic (*PDL*). We give a truth-preserving translation of *PDL* formulas into *ITL* formulas, that is based on this semantic correspondence. We show that the axiom DC^*3 can be obtained by means of this translation from an induction axiom known from *PDL*. Basic definitions about *PDL* can be found in, e.g., [AGM92].

Let us assume that set of the time points T of some abstract *ITL* frame $F = \langle\langle T, \leq\rangle, \langle D, +, 0\rangle, m\rangle$ is also the set of the possible worlds of some *PDL* frame. Consider a *PDL* vocabulary consisting of the set of propositional letters P and the set of relation letters R. Let v be a *PDL* valuation of the vocabulary $P \cup R$ into the *PDL* frame based on T, that is, let $v(p) \subseteq T$ for $p \in P$ and $v(r) \subseteq T \times T$ for $r \in R$. Let us consider only *PDL* valuations v which satisfy the condition $v(r) \subseteq \leq$ for $r \in R$. Since $Id_T \subseteq \leq$, and $R, S \subseteq \leq$ implies $R \cup S, R \circ S$ and $R^* \subseteq \leq$, we can assume that the standard extensions $\tilde{v}$ of such v over relation terms give only subrelations of $\leq$ too.

Assume that $\langle T, \leq\rangle$ is bounded, that is, let $T = [\min T, \max T]$ and consider an *ITL* vocabulary $\mathbf{L}$ whose set of temporal propositional letters is $P \cup R$. We are not interested in specifying the rest of $\mathbf{L}$. Given the *PDL* valuation v, we can define an interpretation I of these temporal propositional letters into F by putting

$I(p)([\tau_1, \tau_2]) = 1$ iff $\tau_1 \in v(p)$ and $\tau_2 = \max T$ for $p \in P$;

$I(r)([\tau_1, \tau_2]) = 1$ iff $\langle \tau_1, \tau_2\rangle \in v(r)$ for $r \in R$.

Let the translation $\mathtt{t}$ of the *PDL* language based on the vocabulary $P \cup R$ into the ITL^* language based on $\mathbf{L}$ be defined by the clauses:

$\mathsf{t}(\bot) \rightleftharpoons \bot$
$\mathsf{t}(q) \rightleftharpoons q$ for $q \in P \cup R$;
$\mathsf{t}(\varphi \vee \psi) \rightleftharpoons \mathsf{t}(\varphi) \vee \mathsf{t}(\psi)$
$\mathsf{t}(\neg\varphi) \rightleftharpoons \neg\mathsf{t}(\varphi)$
$\mathsf{t}(Id) \rightleftharpoons \ell = 0$
$\mathsf{t}(\alpha \cup \beta) \rightleftharpoons \mathsf{t}(\alpha) \vee \mathsf{t}(\beta)$
$\mathsf{t}(\alpha \circ \beta) \rightleftharpoons \mathsf{t}((\alpha)^\frown \mathsf{t}(\beta))$
$\mathsf{t}(\alpha^*) \rightleftharpoons (\mathsf{t}(\alpha))^*$
$\mathsf{t}(\langle\alpha\rangle\varphi) \rightleftharpoons (\mathsf{t}(\alpha)^\frown \mathsf{t}(\varphi))$

The following proposition explains the correspondence between the *PDL* model based on T and v, and the *ITL* model $\langle F, I\rangle$ using t:

Proposition 4 *Let φ be a PDL formula in the vocabulary $P \cup R$. Then*

$$T, v, \min T \models \varphi \textit{ iff } \langle F, I\rangle, [\min T, \max T] \models \mathsf{t}(\varphi).$$

Proof: Direct check by induction on the construction φ. □

PDL has the following induction axiom about iteration in its proof system ([AGM92]):

$$[\alpha^*](\varphi \Rightarrow [\alpha]\varphi) \Rightarrow (\varphi \Rightarrow [\alpha^*]\varphi)\,.$$

The t-translation of this axiom is equivalent to

$$(\alpha^* {}^\frown(\alpha^* {}^\frown\psi) \wedge \neg\psi) \vee ((\alpha^* {}^\frown\psi) \Rightarrow \psi) \tag{8}$$

where ψ stands for $\neg\varphi$.

The validity of the *PDL* induction axiom enforces $(\tilde{v}(\alpha))^* \supseteq \tilde{v}(\alpha^*)$ on the frame level and therefore characterises iteration the way local correspondents do in modal logic (cf. e.g. [vB83]). Characterisation on the frame level assumes the freedom to choose interpretations into a fixed frame and generally does not imply deductive characterisation. The corresponding inclusion about iteration in DC^* is enforced on the frame level by DC^*3, which can be obtained from (8) by replacing its subformulas with ψ of the form $(\chi_1{}^\frown\psi \wedge \chi_2)$ by $(\chi_1 \wedge \psi^\frown\chi_2)$, and some simple *ITL* transformations. To understand this replacement, note that the satisfaction of formulas of the kind $(\chi_1{}^\frown\psi \wedge \chi_2)$ at an interval σ in a model M depends on the set of those $\tau \in \sigma$ for which $M, [\tau, \max\sigma] \models \psi$. However, such a set of time points τ can be defined by a condition of the form $M, [\min\sigma, \tau] \models \psi'$ for some appropriate ψ' as well. The move from $(\chi_1{}^\frown\psi \wedge \chi_2)$ to $(\chi_1 \wedge \psi^\frown\chi_2)$ facilitated the proof of the completeness of a system for DC^* with DC^*3 in it for subsets of DC^* in [Gue98, Gue00c] and the precursor of this paper [DG99]. To achieve deductive characterisation, the benefit from *choosing* interpretations as allowed on the frame level can be supplied by *finding* formulas with appropriate meanings. In the case of DC^*3 the scope of the result from [DG99] is limited by our ability to find formulas ψ which approximate φ^* for φ in the considered subset and argue that the respective instances of DC^*3 force φ^* to have the right truth value.

The axioms DC^*1 and DC^*2 can be obtained by the same translation from the *PDL* axioms $[\alpha^*]\varphi \Rightarrow \varphi$ and $[\alpha^*]\varphi \Rightarrow [\alpha^*][\alpha]\varphi$, respectively.

6 Some more examples of the use of DC^*1–DC^*5

The interderivability proofs from Section 4 illustrate the working of our axioms and proof rule. In this section we give derivations for a couple of DC^* theorems of general interest and use one of them in a proof about our introductory gas-burner example in order to give some such illustration with a practical flavour.

Here are two derivations of the monotonicity of iteration. One of them involves DC^*3:

$$\begin{array}{lll} \alpha^* \wedge \neg\beta^* & \Rightarrow (\neg\beta^* \wedge \ell = 0^\frown\top) \vee (((\alpha^* \wedge \beta^{*\frown}\alpha) \wedge \neg\beta^*)^\frown\top) & \text{by } DC^*3 \\ & \Rightarrow (((\alpha^* \wedge \beta^{*\frown}\alpha) \wedge \neg\beta^*)^\frown\top) & \text{by } DC^*1 \\ & \Rightarrow (\neg\beta^* \wedge \beta^*)^\frown\top & \text{by } DC_2^* \\ & & \text{and } \alpha \Rightarrow \beta \\ & \Rightarrow \bot & \end{array}$$

The other involves the proof rules ω and DC^*5:

$$\begin{array}{lll} 1 & (\lceil P\rceil \vee \lceil\neg P\rceil)^k \Rightarrow \left(\mathsf{g}(\alpha,P) \Rightarrow \left(\Box(\varphi \Rightarrow \beta) \Rightarrow \bigvee\limits_{l\le k} \beta^l\right)\right) & k<\omega,\ DC \\ 2 & \beta^m \Rightarrow \beta^* & m<\omega,\ DC^*1,\ DC^*2,\ DC \\ 3 & (\lceil P\rceil \vee \lceil\neg P\rceil)^k \Rightarrow (\mathsf{g}(\alpha,P) \Rightarrow (\Box(\varphi \Rightarrow \beta) \Rightarrow \beta^*)) & k<\omega,\ 1,\ 2,\ DC \\ 4 & \mathsf{g}(\alpha,P) \Rightarrow (\Box(\varphi \Rightarrow \beta) \Rightarrow \beta^*) & 3,\ \omega \\ 5 & \alpha^* \Rightarrow (\Box(\varphi \Rightarrow \beta) \Rightarrow \beta^*) & 4,\ DC^*5 \\ 6 & \Box(\varphi \Rightarrow \beta) \Rightarrow (\alpha^* \Rightarrow \beta^*) & \end{array}$$

Here follows another useful DC^* theorem:

$$\begin{array}{ll} \vdash_{DC^*} & \Box(\varphi \Rightarrow \neg(\top^\frown\neg\alpha) \wedge \neg(\neg\beta^\frown\top))\wedge \\ & \Box(\ell = 0 \Rightarrow \alpha \wedge \beta) \wedge \Box(\beta \Rightarrow \neg(\top^\frown\neg\gamma)) \Rightarrow \\ & \Rightarrow \varphi^* \Rightarrow \Box(\gamma \vee (\alpha^\frown\varphi^{*\frown}\beta))\,. \end{array} \tag{9}$$

To prove it in our system, below we give a derivation of $\varphi^* \Rightarrow \Box(\gamma \vee (\alpha^\frown\varphi^{*\frown}\beta))$ using

$$\varphi \Rightarrow \neg(\top^\frown\neg\alpha), \varphi \Rightarrow \neg(\neg\beta^\frown\top), \beta \Rightarrow \neg(\top^\frown\neg\gamma) \text{ and } \ell = 0 \Rightarrow \alpha, \ell = 0 \Rightarrow \beta$$

as assumptions. Then (9) will follow by the deduction theorem for DC [HZ97].

1	$\varphi \Rightarrow \neg(\top\frown\neg\alpha)$	assumption
2	$\varphi \Rightarrow \neg(\top\frown\neg(\alpha\frown\ell = 0))$	by 1
3	$\ell = 0 \Rightarrow \varphi^*$	by DC_1^*
4	$\varphi \Rightarrow \neg(\top\frown\neg(\alpha\frown\varphi^*))$	by 2, 3, $Mono_r$
5	$\ell = 0 \Rightarrow \alpha$	asumption
6	$\ell = 0 \Rightarrow (\ell = 0\frown\ell = 0)$	$L2$
7	$\ell = 0 \Rightarrow (\alpha\frown\varphi^*))$	by 5, 6, DC_1^*, $Mono_l$, $Mono_r$
8	$(\top\frown\neg(\alpha\frown\varphi^*)) \Rightarrow \neg\ell = 0$	by 7, DC
9	$\neg((\top\frown\neg(\alpha\frown\varphi^*)) \wedge \ell = 0\frown\top)$	by 8, N_l
10	$(\varphi^* \wedge (\top\frown\neg(\alpha\frown\varphi^*))) \Rightarrow$ $((\top\frown\neg(\alpha\frown\varphi^*)) \wedge \ell = 0\frown\top)\vee$ $((\varphi^* \wedge \neg(\top\frown\neg(\alpha\frown\varphi^*))\frown\varphi)\wedge$ $(\top\frown\neg(\alpha\frown\varphi^*))\frown\top)$	by DC^*3
11	$(\varphi^* \wedge \neg(\top\frown\neg(\alpha\frown\varphi^*))\frown\varphi)\wedge$ $(\top\frown\neg(\alpha\frown\varphi^*)) \Rightarrow$ $(\top\frown(\alpha\frown\varphi^*\frown\varphi) \wedge \neg(\alpha\frown\varphi^*))\vee$ $(\varphi \wedge (\top\frown\neg(\alpha\frown\varphi^*))))$	DC
12	$(\alpha\frown\varphi^*\frown\varphi) \Rightarrow (\alpha\frown\varphi^*)$	by DC_2^*, $Mono_r$
13	$\neg((\alpha\frown\varphi^*\frown\varphi) \wedge \neg(\alpha\frown\varphi^*))$ $\vee(\varphi \wedge (\top\frown\neg(\alpha\frown\varphi^*)))$	by 4, $Mono_r$, 14
14	$\neg(\top\frown((\alpha\frown\varphi^*\frown\varphi) \wedge \neg(\alpha\frown\varphi^*))$ $\vee(\varphi \wedge (\top\frown\neg(\alpha\frown\varphi^*))))$	by 13, N_r
15	$\neg((\varphi^* \wedge \neg(\top\frown\neg(\alpha\frown\varphi^*))\frown\varphi)\wedge$ $(\top\frown\neg(\alpha\frown\varphi^*)))$	by 11, 14
16	$\varphi^* \Rightarrow \neg(\top\frown\neg(\alpha\frown\varphi^*)))$	by 9, 10, 15, $Mono_l$
17	$\varphi^* \wedge (\neg(\varphi^*\frown\beta)\frown\top) \Rightarrow$ $((\neg(\varphi^*\frown\beta)\frown\top) \wedge \ell = 0\frown\top)\vee$ $(((\varphi^* \wedge \neg(\neg(\varphi^*\frown\beta)\frown\top))\frown\varphi) \wedge (\neg(\varphi^*\frown\beta)\frown\top)\frown\top)$	DC_3^*
18	$\ell = 0 \Rightarrow \beta$	assumption
19	$\ell = 0 \Rightarrow \varphi^*$	DC_1^*
20	$\ell = 0 \Rightarrow \neg(\neg(\varphi^*\frown\beta)\frown\top)$	18, 19, DC
21	$\neg((\neg(\varphi^*\frown\beta)\frown\top) \wedge \ell = 0\frown\top)$	20, DC
22	$\neg(((\varphi^* \wedge \neg(\neg(\varphi^*\frown\beta)\frown\top))\frown\varphi) \wedge (\neg(\varphi^*\frown\beta)\frown\top)\frown\top)$	DC
23	$\varphi^* \Rightarrow \neg(\neg(\varphi^*\frown\beta)\frown\top)$	17, 20, 22, DC
24	$\beta \Rightarrow \neg(\top\frown\neg\gamma)$	assumption
25	$\varphi^* \Rightarrow \Box(\gamma \vee (\alpha\frown\varphi^*\frown\beta))$	16, 23, 24, DC

Now let us prove the correctness of the gas-burner design from the introduction as a last example of the working of our DC^* axioms and rule. We have to give a derivation for

$$((\lceil\!\lceil leak\rceil\!\rceil \wedge \ell \leq 1)\frown(\lceil\!\lceil nonleak\rceil\!\rceil \wedge \ell \geq 30))^* \Rightarrow \Box(\ell \geq 60 \Rightarrow 20 \textstyle\int leak \leq \ell)\,.$$

Let

$$\begin{array}{rcl}
\varphi & \rightleftharpoons & \lceil\!\lceil leak\rceil\!\rceil \wedge \ell \leq 1\frown\lceil\!\lceil \neg leak\rceil\!\rceil \wedge \ell \geq 30, \\
\alpha & \rightleftharpoons & \ell = 0 \vee \lceil\!\lceil \neg leak\rceil\!\rceil \vee (\lceil\!\lceil leak\rceil\!\rceil \wedge \ell \leq 1\frown\lceil\!\lceil \neg leak\rceil\!\rceil \wedge \ell \geq 30)\,, \\
\beta, \gamma & \rightleftharpoons & \ell = 0 \vee (\ell \leq 1 \wedge \lceil\!\lceil leak\rceil\!\rceil\frown\ell = 0 \vee \lceil\!\lceil \neg leak\rceil\!\rceil)\,.
\end{array}$$

The formulas

$$\Box(\varphi \Rightarrow \neg(\top^\frown\neg\alpha) \wedge \neg(\neg\beta^\frown\top)),\ \Box(\ell = 0 \Rightarrow \alpha \wedge \beta) \text{ and } \beta \Rightarrow \neg(\top^\frown\neg\gamma)$$

are valid in DC without iteration. Therefore we can complete our derivation using (9), provided we can derive

$$\gamma \Rightarrow 20\int leak \leq \ell \text{ and } (\alpha^\frown\varphi^{*\frown}\beta) \wedge \ell \geq 60 \Rightarrow 20\int leak \leq \ell.$$

The first formula is straightforward to derive without DC^*-specific axioms. Here follows a derivation for the second formula:

1	$\alpha \Rightarrow 31\int leak \leq \ell$	DC
2	$\varphi^* \wedge 31\int leak > \ell \Rightarrow (\varphi^* \wedge 31\int leak > \ell^\frown\top)$	DC
3	$\varphi \Rightarrow 31\int leak \leq \ell$	DC
4	$(\varphi^* \wedge 31\int leak > \ell^\frown\top) \Rightarrow$ $(\ell = 0 \wedge 31\int leak > \ell^\frown\top)\vee$ $(((\varphi^* \wedge 31\int leak \leq \ell^\frown\varphi) \wedge 31\int leak > \ell)^\frown\top)$	by DC^*3
5	$\ell = 0 \Rightarrow 31\int leak \leq \ell$	DC
6	$(\varphi^* \wedge 31\int leak > \ell^\frown\top) \Rightarrow$ $(((\varphi^* \wedge 31\int leak \leq \ell^\frown\varphi) \wedge 31\int leak > \ell)^\frown\top)$	by 4, 5, $Mono_r$
7	$(\varphi^* \wedge 31\int leak \leq \ell^\frown\varphi) \Rightarrow 31\int leak \leq \ell$	by 2, 3, DC
8	$\varphi^* \Rightarrow 31\int leak \leq \ell$	by 6, 7, $Mono_r$
9	$(\alpha^\frown\varphi^*) \Rightarrow 31\int leak \leq \ell$	by 1, 8, DC
10	$\beta \Rightarrow \int leak \leq 1$	DC
11	$(\alpha^\frown\varphi^{*\frown}\beta) \wedge \ell \geq 60 \Rightarrow 20\int leak \leq \ell$	by 9, 10, DC

7 Related work on the axiomatisation of iteration

Iteration is known *chop-star* in Moszkowski's original discrete-time *ITL*, where it is regarded as part of the basic system. Unlike real- and abstract-time *DC*, finite variability is a trivial property in bounded discrete time intervals and therefore discrete-time *ITL* is recursively axiomatisable, whereas in *DC* one has to settle for relative completeness or bring in ω-rules like in this paper. Apart from that, the axioms DC^*1–DC^*4 are valid in discrete-time *ITL* too and can be derived in its proof system. The rule DC^*5, however, is new and *DC*-specific. An analogous rule can, in principle, be put together for discrete-time *ITL* too by writing a formula with a meaning like that of $\mathsf{f}(\varphi, P)$, but we do not know this to have been worked on.

Another difference with discrete-time *ITL* is the adoption of the *locality principle* there. This means that the truth values of propositional temporal letters depend only on the beginning point of the reference interval. The locality principle makes propositional discrete-time *ITL* equivalent to (untimed) regular expressions in both expressive power and complexity. We should note the recent advances in both the axiomatisation and the decision procedures for discrete-time *ITL* from [Mos03] where both issues are elegantly handled using a proposed *hierarchical* complete proof system. The role of propositional temporal letters with the locality principle can largely be taken by DC state variables, which means that the decidability results about the duration-free subset of simple DC known as the $\lceil P \rceil$*-subset of* DC from [ZHS93] look akin to some earlier results on propositional discrete-time ITL. By contrast, adding propositional temporal letters without the locality principle to the $\lceil P \rceil$-subset of DC renders it non-recursively axiomatisable [Gue04].

8 Discussion on the decidability results for simple DC^*

As said in the introduction of the paper, representing the repetitive behaviour of real-time systems is the main motivation for introducing the iteration operator into DC. An important subset of DC^* for this purpose has been introduced in [DW96] as the class of so-called *simple* DC^* formulas, whose syntax can be defined by the BNF

$$\varphi ::= \lceil S \rceil \mid a \leq \ell \mid \ell \leq a \mid (\varphi \vee \varphi) \mid (\varphi \wedge \varphi) \mid (\varphi \frown \varphi) \mid \varphi^* \qquad (10)$$

In this section, we discuss the decidability of the satisfiability of simple DC^* formulas at the real-time frame, i.e. the frame in which the time domain is $\langle \mathbf{R}, \leq \rangle$ and the duration domain is $\langle \mathbf{R}_+, +, 0 \rangle$.

One of the notions in the literature that are close to our notion of simple DC^* is the notion of *timed regular expressions* introduced by Asarin et al in [ACM97], a subset of which has been introduced by us earlier in [LD96]. Simple DC^* formulas syntactically correspond exactly to timed regular expression, and their semantics coincide. Therefore, a simple DC^* formula can be viewed as a timed regular expression. It was shown in how from a timed regular expression E one can build a timed automaton A which recognises exactly the models of E and has only constants from E in the constraints for its clock variables (guards, tests and invariants). It is well known that emptiness is decidable for timed automata with integer constants in their guards and tests [AD94]. This entails the following theorem:

Theorem 3 *The satisfiability of formulas with the syntax (10) in which a stands for integer constants is decidable.*

The complexity of the decidability procedure, however, is exponential in the size of the constants occurring in the clock constraints (see e.g. [AD94]).

In [ACM97] it is also shown how given a timed automaton A, one can build a timed regular expression E and a renaming of the locations of A such that each model of E is the renaming of a behaviour of A. In this sense simple DC^* formulas and timed automata have the same expressive power.

If we restrict ourselves to the class of *sequential* simple DC^* formulas, which can be defined by the BNF

$$\varphi ::= \ell = 0 \mid \lceil S \rceil \mid \varphi \vee \varphi \mid (\varphi \frown \varphi) \mid \varphi^* \mid \varphi \wedge a \leq \ell \mid \varphi \wedge \ell \leq a \ ,$$

then we can have a very simple decision procedure for satisfiability and some interesting results. Since the operators $^\frown$ and $\wedge$ distribute over $\vee$, and because of the equivalence $(\varphi \vee \psi)^* \Leftrightarrow (\varphi^{*\frown}\psi^*)^*$, each sequential simple DC^* formula φ is equivalent to a disjunction of $\vee$-free simple formulas. Such a φ is satisfiable iff at least one of its disjunctive members is satisfiable. The satisfiability of $\vee$-free sequential simple DC^* formulas is very easy to decide indeed. Let $\min(\varphi), \max(\varphi) \in \mathbf{R}$ be defined for such φ by the clauses

$\min(\ell = 0) = \max(\ell = 0) = 0$
$\min(\llbracket S \rrbracket) = 0$, $\max(\llbracket S \rrbracket) = \infty$
$\min(\varphi_1{}^\frown\varphi_2) = \min(\varphi_1) + \min(\varphi_2)$, $\max(\varphi_1{}^\frown\varphi_2) = \max(\varphi_1) + \max(\varphi_2)$
$\min(\varphi^*) = 0$, If $\max(\varphi) > 0$ then $\max(\varphi^*) = \infty$ otherwise $\max(\varphi^*) = 0$.
$\min(\varphi \wedge a \leq \ell) = \max\{\min(\varphi), a\}$, $\max(\varphi \wedge a \leq \ell) = \max(\varphi)$
$\min(\varphi \wedge \ell \leq a) = \min(\varphi)$, $\max(\varphi \wedge \ell \leq a) = \min\{\max(\varphi), a\}$

Obviously φ is satisfiable iff $\min(\varphi) \leq \max(\varphi)$.

In [LD96, LDZ97], we have developed some simple algorithms for checking a real-time system whose behaviour is described by a 'sequential' timed regular expression for a linear duration invariants of the form

$$\Box(a \leq \ell \leq b \Rightarrow \textstyle\sum_{S\in\mathbf{S}} c_S \int S \leq M)\,,$$

where $\mathbf{S}$ is a finite set of state variables. Because of the obvious correspondence between sequential simple DC^* formulas and sequential timed regular expressions, these algorithms can be used for proving automatically the implication from a sequential simple DC^* formula to a linear duration invariant. An advantage of the method is that it reduces the problem to well-understood linear programming problems. Because of this advantage, in [DP98], we tried to generalise the method for the general simple DC^* formulas, and showed that in most cases, the method can still be used for checking the implication from a simple DC^* formula to a linear duration invariant.

9 Concluding remarks

The contribution of this paper is to show that iteration can be defined and/or axiomatised (relatively) completely in *quantified systems* of DC with some support from the notion of finite variability of state which is more deeply seated in DC. This approach helps to identify and restrict finite variability *of state* as DC's only source of recursive unaxiomatisability, which can be regarded as *imcompleteness* in the sense of Gödel. Having the infinitary rule about the finite variability of state, the class of duration domains targetted by our completeness theorem can be narrowed down to each of the practically important domains $\langle \mathbf{R}_+, 0, + \rangle$ and $\langle \mathbf{N}, 0, + \rangle$ by enforcing other principles which defy recursive axiomatisation with no further infinitary additions. For instance,

$$x = 0 \vee (\ell \leq x)^*$$

means that there are no intervals with "infinitely small" (*non-standard*) durations, which is one possible form of the "the missing part" in the relatively complete axiomatisation with respect to real time from [HZ92]. By extending arithmetic with multiplication and division and the real-closed field axioms about them (cf. e.g. [Sho67]) this axiom can be shown to entail the *principle of Archimedes* which rules out "infinitely large" real numbers. As for discrete time,

$$(\ell = 1 \land \neg(\ell \neq 0 ^\frown \ell \neq 0))^*$$

means that every interval is a finite union of unit intervals which themselves have no internal points.

Finite variability of state does not seem to be helpful enough for the axiomatisation of the general least-fixed-point operator μ in *DC* as known from [Pan95]. For the time being, the method from [Gue00a] which is similar to that from the precursor [DG99] of this paper, allows us to do only subsets of *HDC* with μ.

References

[ACM97] Eugene Asarin, Paul Caspi, and Oded Maler. A Kleene Theorem for Timed Automata. In G. Winskel, editor, *Proceedings of IEEE International Symposium on Logics in Computer Science LICS'97*, pages 160–171. IEEE Computer Society Press, 1997.

[AD94] Rajeev Alur and David L. Dill. A Theory of Timed Automata. *Theoretical Computer Science*, 126:183–235, 1994.

[AGM92] Samson Abramsky, Dov Gabbay, and T.S.E. Maibaum, editors. *Handbook of Logic in Computer Science*. Clarendon Press, 1992.

[Dan98] Dang Van Hung. Modelling and Verification of Biphase Mark Protocols in Duration Calculus Using PVS/DC$^-$. In *Proceedings of the 1998 International Conference on Application of Concurrency to System Design (CSD'98)*, pages 88–98. IEEE Computer Society Press, March 1998.

[DG99] Dang Van Hung and Dimitar P. Guelev. Completeness and Decidability of a Fragment of Duration Calculus with Iteration. In P.S. Thiagarajan and R. Yap, editors, *Advances in Computing Science - ASIAN'99*, volume 1742 of *LNCS*, pages 139–150. Springer, 1999.

[DP98] Dang Van Hung and Pham Hong Thai. On Checking Parallel Real-Time Systems for Linear Duration Invariants. In B. Kramer, N. Uchihita, P. Croll, and S. Russo, editors, *Proceedings of the International Symposium of Software Engineering for Parallel and Distributed Systems (PDSE'98)*, pages 61–71. IEEE Computer Society Press, April 1998.

[Dut95a] Bruno Dutertre. On First-order Interval Temporal Logic. Report CSD-TR-94-3, Department of Computer Science, Royal Holloway, University of London, Egham, Surrey TW20 0EX, England, 1995. A short version appeared as [Dut95b].

[Dut95b] Bruno Dutertre. On First Order Interval Temporal Logic. In *Proceedings of LICS'95*, pages 36–43. IEEE Computer Society Press, 1995.

[DW96] Dang Van Hung and Wang Ji. On The Design of Hybrid Control Systems Using Automata Models. In *Proceedings of FST TCS 1996*, volume 1180 of *LNCS*, pages 156–167. Springer, 1996.

[Gue98] Dimitar P. Guelev. A Calculus of Durations on Abstract Domains: Completeness and Extensions. Technical Report 139, UNU/IIST, P.O.Box 3058, Macau, May 1998.

[Gue00a] Dimitar P. Guelev. A Complete Fragment of Higher-order Duration μ-calculus. In *Proceedings of FST TCS 2000*, volume 1974 of *LNCS*, pages 264–276. Springer, 2000.

[Gue00b] Dimitar P. Guelev. Interpolation and Related Results on the $\lceil P \rceil$-fragment of *DC* with Iteration. Technical Report 203, UNU/IIST, P.O. Box 3058, Macau, June 2000.

[Gue00c] Dimitar P. Guelev. *Probabilistic and Temporal Modal Logics*. Ph.D. thesis, Sofia University, 2000. (In Bulgarian).

[Gue04] Dimitar P. Guelev. A Complete Proof System for First-order Interval Temporal Logic with Projection. *Journal of Logic and Computation*, 14(2):215–249, 2004.

[Gue05] Dimitar P. Guelev. Sharpening the Incompleteness of the Duration Calculus. In Irek Ulidowski, editor, *Proceedings of ARTS 2004*, volume 139(1) of *ENTCS*, pages 91–104. Elsevier Science, 2005. Presented at ARTS 2004, Stirling, UK.

[He 99a] He Jifeng. A Behavioral Model for Co-design. In *Proceedings of FM'99*, volume 1709 of *LNCS*, pages 1420–1438. Springer, 1999.

[He 99b] He Jifeng. Integrating Variants of *DC*. Research Report 172, UNU/IIST, P.O.Box 3058, Macau, August 1999.

[HZ92] Michael R. Hansen and Zhou Chaochen. Semantics and Completeness of Duration Calculus. In *Real-Time: Theory and Practice*, volume 600 of *LNCS*, pages 209–225. Springer, 1992.

[HZ95] He Weidong and Zhou Chaochen. A Case Study of Optimization. *The Computer Journal*, 38(9):734–746, 1995.

[HZ96] Michael R. Hansen and Zhou Chaochen. Chopping a Point. In *BCS-FACS 7th refinement workshop*, Electronic Workshops in Computing. Springer, 1996.

[HZ97] Michael R. Hansen and Zhou Chaochen. Duration Calculus: Logical Foundations. *Formal Aspects of Computing*, 9:283–330, 1997.

[Koz83] Dexter Kozen. Results on the propositional μ-calculus. *Theoretical Computer Science*, 27:333–354, 1983.

[LD96] Li Xuan Dong and Dang Van Hung. Checking Linear Duration Invariants by Linear Programming. In J. Jaffar and R. H. C. Yap, editors, *Concurrency and Palalellism, Programming, Networking, and Security*, volume 1179 of *LNCS*, pages 321–332. Springer, 1996.

[LDZ97] Li Xuan Dong, Dang Van Hung, and Zheng Tao. Checking Hybrid Automata for Linear Duration Invariants. In R. K. Shamasundar and K. Ueda, editors, *Advances in Computing Science*, volume 1345 of *LNCS*, pages 166–180. Springer, 1997.

[Mos85] Ben Moszkowski. Temporal Logic For Multilevel Reasoning About Hardware. *IEEE Computer*, 18(2):10–19, 1985.

[Mos86] Ben Moszkowski. *Executing Temporal Logic Programs.* Cambridge University Press, 1986. URL: `http://www.cse.dmu.ac.uk/ cau/papers/tempura-book.pdf`.

[Mos03] Ben Moszkowski. A Hierarchical Completeness Proof for Propositional Temporal Logic . In *Verification: Theory and Practice. Essays Dedicated to Zohar Manna on the Occasion of His 64th Birthday*, volume 2772 of *LNCS*, pages 480–523. Springer, 2003.

[Pan95] Paritosh K. Pandya. Some extensions to Mean-Value Calculus: Expressiveness and Decidability. In *Proceedings of CSL'95*, volume 1092 of *LNCS*, pages 434–451. Springer, 1995.

[PD98] Paritosh K. Pandya and Dang Van Hung. Duration Calculus of Weakly Monotonic Time. In *Proceedings of FTRTFT'98*, volume 1486 of *LNCS*, pages 55–64. Springer, 1998.

[PD99] Ekaterina Pavlova and Dang Van Hung. A Formal Specification of the Concurrency Control in Real-Time Databases. Technical Report 152, UNU/IIST, P.O.Box 3058, Macau, January 1999.

[Sho67] Joseph Shoenfield. *Mathematical Logic.* Addison-Wesley, Reading, Massachusetts, 1967.

[Sko00] Dimiter Skordev. On the Duration Domains for Interval Temporal Logic. *Annuaire de l'universite de Sofia "St. Kliment Ochriski"*, 94:27–33, 2000.

[vB83] J. A. F. K. van Benthem. *Modal Logic and Classical Logic.* Bibliopolis, 1983.

[WHCZ96] B. H. Widjaja, He Weidong, Chen Zongji, and Zhou Chaochen. A Cooperative Design for Hybrid Systems. In A. Pnueli and H. Lin, editors, *Logic and Software Engineering. International Workshop in Honor of Chih-Sung Tang*, pages 127–150. World Scientific, 1996.

[XH95] Xu Qiwen and He Weidong. Hierarchical Design of a Chemical Concentration Control System. In *Proceedings of* Hybrid Systems III: Verification and Control, volume 1066 of *LNCS*, pages 270–281. Springer, 1995.

[YWZP94] Yu Xinyao, Wang Ji, Zhou Chaochen, and Paritosh K. Pandya. Specification of an Adaptive Control System. In *Proceedings of FTRTFT'94*, volume 863 of *LNCS*, pages 738–755. Springer, 1994.

[ZGZ00] Zhou Chaochen, Dimitar P. Guelev, and Zhan Naijun. A Higher-order Duration Calculus. In *Millennial Perspectives in Computer Science*, pages 407–416. Palgrave, 2000.

[ZH04] Zhou Chaochen and Michael R. Hansen. *Duration Calculus. A Formal Approach to Real-Time Systems.* Springer, 2004.

[ZHR91] Zhou Chaochen, C. A. R. Hoare, and Anders P. Ravn. A Calculus of Durations. *Information Processing Letters*, 40(5):269–276, 1991.

[ZHS93] Zhou Chaochen, Michael R. Hansen, and P. Sestoft. Decidability and Undecidability Results for Duration Calculus. In *Proceedings of STACS'93*, volume 665 of *LNCS*, pages 58–68. Springer, 1993.

[ZL94] Zhou Chaochen and Li Xiaoshan. *A Mean Value Calculus of Durations*, pages 431–451. Prentice Hall, 1994.

[ZZ94] Zheng Yuhua and Zhou Chaochen. A Formal Proof of a Deadline Driven Scheduler. In *Proceedings of FTRTFT'94*, volume 863 of *LNCS*, pages 756–775. Springer, 1994.

[ZZYL94] Zhou Chaochen, Zhang Jingzhong, Yang Lu, and Li Xiaoshan. Linear Duration Invariants. In *Proceedings of FTRTFT'94*, volume 863 of *LNCS*, pages 86–109. Springer, 1994.