

Real-Time Systems

Lecture 7: DC Properties II

2013-05-14

Dr. Bernd Westphal

Albert-Ludwigs-Universität Freiburg, Germany

Contents & Goals

Last Lecture:

- RDC in discrete time
- Started: Satisfiability and realisability from 0 is decidable for RDC in discrete time

This Lecture:

- **Educational Objectives:** Capabilities for following tasks/questions.
 - Facts: (un)decidability properties of DC in discrete/continuous time.
 - What's the idea of the considered (un)decidability proofs?
- **Content:**
 - Complete: Satisfiability and realisability from 0 is decidable for RDC in discrete time
 - Undecidable problems of DC in continuous time

RDC in Discrete Time Cont'd

Recall: Decidability of Satisfiability/Realisability from 0

Theorem 3.6.

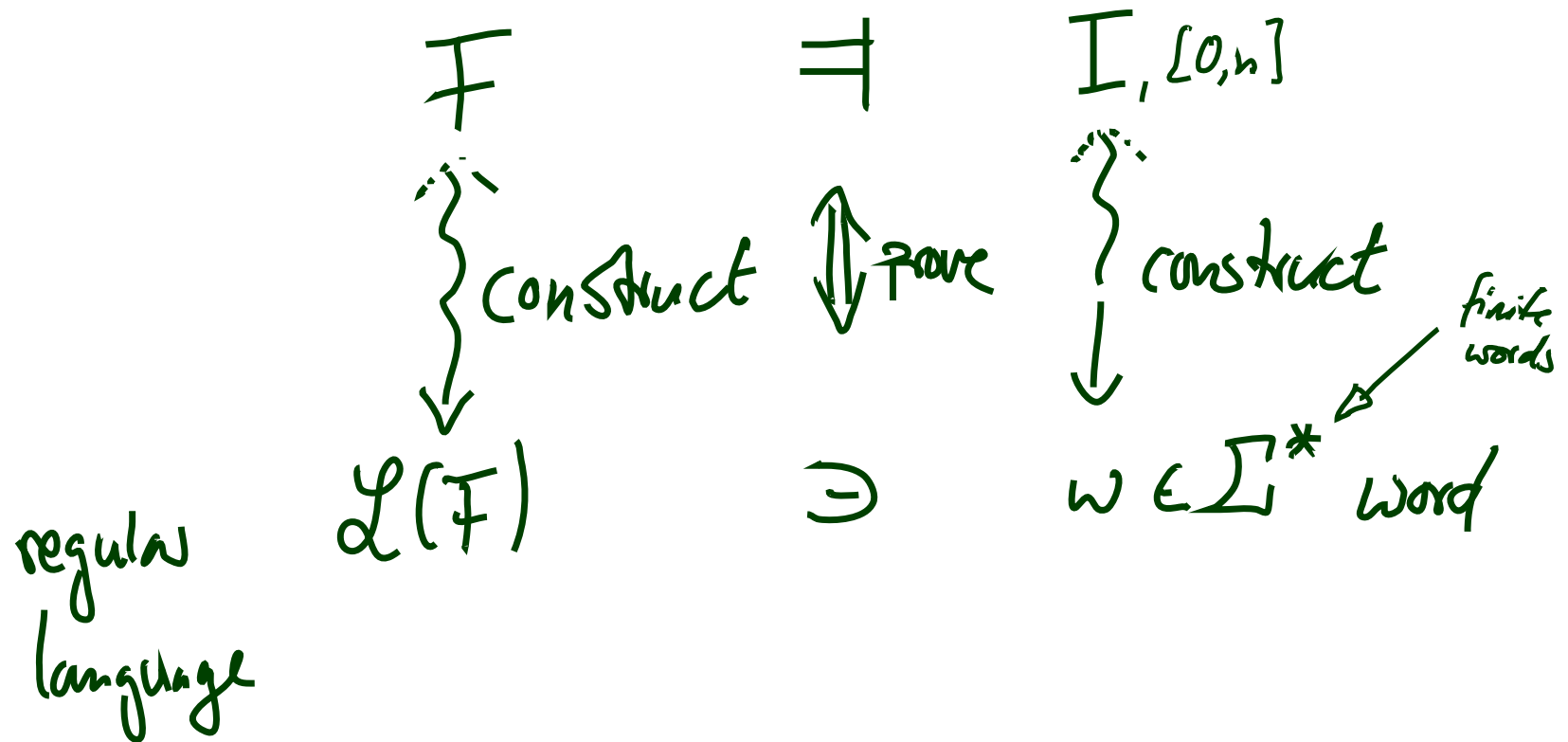
The satisfiability problem for RDC with discrete time is decidable.

Theorem 3.9.

The realisability problem for RDC with discrete time is decidable.

Recall: Proof Sketch

RDC formula F , DT/ Σ .



Sketch: Proof of Theorem 3.6

- give a procedure to construct, given a formula F , a **regular** language $\mathcal{L}(F)$ such that

$$\mathcal{I}, [0, n] \models F \text{ if and only if } w \in \mathcal{L}(F)$$

where word w describes \mathcal{I} on $[0, n]$
(suitability of the procedure: **Lemma 3.4**)

- then F is satisfiable in discrete time if and only if $\mathcal{L}(F)$ is not empty (**Lemma 3.5**)
- Theorem 3.6 follows because
 - $\mathcal{L}(F)$ can **effectively** be constructed,
 - the emptiness problem is **decidable** for regular languages.

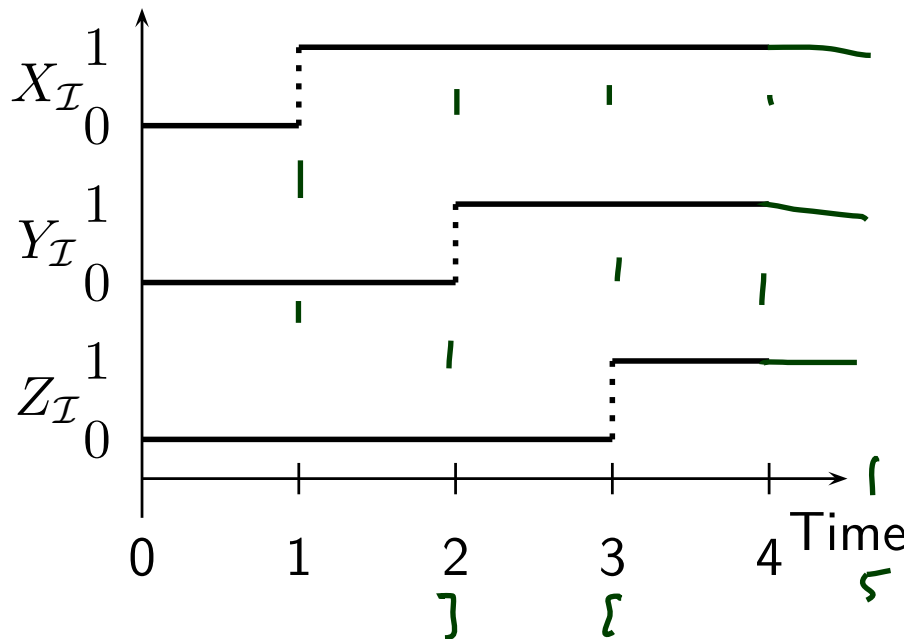
Construction of $\mathcal{L}(F)$

- Idea:**

- alphabet $\Sigma(F)$ consists of basic conjuncts of the state variables in F ,
- a letter corresponds to an interpretation on an interval of length 1,
- a word of length n describes an interpretation on interval $[0, n]$.

- Example:** Assume F contains exactly state variables X, Y, Z , then

$$\Sigma(F) = \{X \wedge Y \wedge Z, X \wedge Y \wedge \neg Z, X \wedge \neg Y \wedge Z, X \wedge \neg Y \wedge \neg Z, \\ \neg X \wedge Y \wedge Z, \neg X \wedge Y \wedge \neg Z, \neg X \wedge \neg Y \wedge Z, \neg X \wedge \neg Y \wedge \neg Z\}.$$



$$w = (\neg X \wedge \neg Y \wedge \neg Z) \\ \cdot (X \wedge \neg Y \wedge \neg Z) \\ * \cdot (X \wedge Y \wedge \neg Z) \\ \cdot (X \wedge Y \wedge Z) \in \Sigma(F)^* \\ \cdot (X \wedge Y \wedge Z)$$

Construction of $\mathcal{L}(F)$ more Formally

Definition 3.2. A word $w = a_1 \dots a_n \in \Sigma(F)^*$ with $n \geq 0$ describes a **discrete** interpretation \mathcal{I} on $[0, n]$ if and only if

$$\forall j \in \{1, \dots, n\} \forall t \in]j-1, j[: \mathcal{I}[[a_j]](t) = 1.$$

For $n = 0$ we put $w = \varepsilon$.

$$X \wedge Y \Leftrightarrow (X \wedge Y \wedge Z) \vee (X \wedge Y \wedge \bar{Z})$$

$$\begin{aligned} DNF(X \wedge Y) &= \\ &= \{ (X \wedge Y \wedge Z), \\ &\quad (X \wedge Y \wedge \bar{Z}) \} \end{aligned}$$

- Each state assertion P can be transformed into an equivalent **disjunctive normal form** $\bigvee_{i=1}^m a_i$ with $a_i \in \Sigma(F)$.
- Set $DNF(P) := \{a_1, \dots, a_m\} (\subseteq \Sigma(F))$.
- Define $\mathcal{L}(F)$ inductively:

finite words, length at least one

$$\mathcal{L}([P]) = DNF(P)^+, \quad (\text{regular language})$$

$$\mathcal{L}(\neg F_1) = \Sigma(F)^* \setminus \mathcal{L}(F_1) \quad (\text{again regular})$$

$$\mathcal{L}(F_1 \vee F_2) = \mathcal{L}(F_1) \cup \mathcal{L}(F_2), \quad (\text{---} \cup \text{---})$$

$$\mathcal{L}(F_1 ; F_2) = \mathcal{L}(F_1) \cdot \mathcal{L}(F_2). \quad (\text{---} \cdot \text{---})$$

concatenate

Lemma 3.4

Lemma 3.4. For all RDC formulae F , discrete interpretations \mathcal{I} , $n \geq 0$, and all words $w \in \Sigma(F)^*$ which **describe** \mathcal{I} on $[0, n]$,

$$\mathcal{I}, [0, n] \models F \text{ if and only if } w \in \mathcal{L}(F).$$

Proof: Structural induction

Base $F = \top$: assume $w = a_1 \dots a_n$ describes \mathcal{I} on $[0, n]$

$$\mathcal{I}, [0, n] \models \top \Leftrightarrow \mathcal{I}, [0, n] \models \top \text{ and } n \geq 1$$

$$\Leftrightarrow n \geq 1 \text{ and } \forall 1 \leq j \leq n \bullet \mathcal{I}, [j-1, j] \models \top$$

$$\Leftrightarrow n \geq 1 \text{ and } \forall 1 \leq j \leq n \bullet \mathcal{I}, [j-1, j] \models (\top \wedge [a_j]) \text{ and } a_j \in \text{DNF}(\top)$$

"describes" ↗

$$\Leftrightarrow n \geq 1 \text{ and } \forall 1 \leq j \leq n \bullet a_j \in \text{DNF}(\top) \quad \swarrow \text{clear}$$

$$\Leftrightarrow w \in \text{DNF}(\top)^+$$

$$\Leftrightarrow w \in \mathcal{L}(\top)$$

- Steps:
- $\neg F_1$
 - $F_1 \vee F_2$
 - $F_1 \wedge F_2$

Sketch: Proof of Theorem 3.9

Theorem 3.9.

The realisability problem for RDC with discrete time is decidable.

- $kern(L)$ contains all words of L whose prefixes are again in L .
- If L is regular, then $kern(L)$ is also regular.
- $kern(\mathcal{L}(F))$ can effectively be constructed.
- We have

Lemma 3.8. For all RDC formulae F , F is realisable from 0 in discrete time if and only if $kern(\mathcal{L}(F))$ is infinite.

- Infinity of regular languages is decidable.

(Variants of) RDC in Continuous Time

Recall: Restricted DC (RDC)

$$F ::= [P] \mid \neg F_1 \mid F_1 \vee F_2 \mid F_1 ; F_2$$

where P is a state assertion, but with **boolean** observables **only**.

From now on: “RDC + $\ell = x, \forall x$ ”

$$F ::= [P] \mid \neg F_1 \mid F_1 \vee F_2 \mid F_1 ; F_2 \mid \ell = 1 \mid \ell = x \mid \forall x \bullet F_1$$

Undecidability of Satisfiability/Realisability from 0

Theorem 3.10.

The realisability from 0 problem for DC with **continuous time** is undecidable, not even semi-decidable.

Theorem 3.11.

The satisfiability problem for DC with continuous time is undecidable.

Sketch: Proof of Theorem 3.10

Reduce divergence of **two-counter machines** to realisability from 0:

- Given a two-counter machine \mathcal{M} with final state q_{fin} ,
- construct a DC formula $F(\mathcal{M}) := \text{encoding}(\mathcal{M})$
- such that

\mathcal{M} **diverges** **if and only if** the DC formula

$$F(\mathcal{M}) \wedge \neg \diamond [q_{fin}]$$

is **realisable from 0**.

- If realisability from 0 was (semi-)decidable, divergence of two-counter machines would be (which it isn't).

Recall: Two-counter machines

A **two-counter** machine is a structure

$$\mathcal{M} = (Q, q_0, q_{fin}, Prog)$$

where

- Q is a finite set of **states**,
- comprising the **initial state** q_0 and the **final state** q_{fin}
- $Prog$ is the **machine program**, i.e. a finite set of **commands** of the form

$$q : inc_1 : q' \quad \text{and} \quad q : dec_i : q', q'', \quad i \in \{1, 2\}.$$

$$q : inc_2 : q'$$

$$q, q', q'' \in Q$$

- We assume **deterministic** 2CM: for each $q \in Q$, at most one command starts in q , and q_{fin} is the only state where no command starts.

2CM Configurations and Computations

current state
values of counters 1,2

- a **configuration** of \mathcal{M} is a triple $K = (q, n_1, n_2) \in \mathcal{Q} \times \mathbb{N}_0 \times \mathbb{N}_0$.
- The **transition relation** “ \vdash ” on configurations is defined as follows:

Command	Semantics: $K \vdash K'$
$q : inc_1 : q'$	$(q, n_1, n_2) \vdash (q', n_1 + 1, n_2)$
$q : dec_1 : q', q''$	$(q, 0, n_2) \vdash (q', 0, n_2)$ $(q, n_1 + 1, n_2) \vdash (q'', n_1, n_2)$
$q : inc_2 : q'$	$(q, n_1, n_2) \vdash (q', n_1, n_2 + 1)$
$q : dec_2 : q', q''$	$(q, n_1, 0) \vdash (q', n_1, 0)$ $(q, n_1, n_2 + 1) \vdash (q'', n_1, n_2)$

- The (!) **computation** of \mathcal{M} is a finite sequence of the form (“ \mathcal{M} halts”)

$$K_0 = (q_0, 0, 0) \vdash K_1 \vdash K_2 \vdash \dots \vdash (q_{fin}, n_1, n_2)$$

or an infinite sequence of the form (“ \mathcal{M} diverges”)

$$K_0 = (q_0, 0, 0) \vdash K_1 \vdash K_2 \vdash \dots$$

2CM Example

- $\mathcal{M} = (\mathcal{Q}, q_0, q_{fin}, Prog)$
- commands of the form $q : inc_i : q'$ and $q : dec_i : q', q'', i \in \{1, 2\}$
- configuration $K = (q, n_1, n_2) \in \mathcal{Q} \times \mathbb{N}_0 \times \mathbb{N}_0$.

Command	Semantics: $K \vdash K'$
$q : inc_1 : q'$	$(q, n_1, n_2) \vdash (q', n_1 + 1, n_2)$
$q : dec_1 : q', q''$	$(q, 0, n_2) \vdash (q', 0, n_2)$ $(q, n_1 + 1, n_2) \vdash (q'', n_1, n_2)$
$q : inc_2 : q'$	$(q, n_1, n_2) \vdash (q', n_1, n_2 + 1)$
$q : dec_2 : q', q''$	$(q, n_1, 0) \vdash (q', n_1, 0)$ $(q, n_1, n_2 + 1) \vdash (q'', n_1, n_2)$

$$Q = \{q_0, q_1, q_{fin}\}$$

$$Prog = \{q_0 : inc_1 : q_1, q_1 : inc_1 : q_{fin}\}$$

$$\begin{array}{l} (q_0, 0, 0) \\ \vdash \\ (q_1, 1, 0) \\ \vdash \\ (q_{fin}, 2, 0) \end{array} \hookrightarrow \text{machine halts}$$

$$Q = \{q_0, q_{fin}\}$$

$$Prog = \{q_0 : inc_2 : q_0\}$$

$$(q_0, 0, 0)$$

$$\vdash (q_0, 1, 0)$$

$$\vdash (q_0, 2, 0)$$

$$\vdash \dots$$

\hookrightarrow machine diverges

Reducing Divergence to DC realisability: Idea In Pictures

ZCM \mathcal{M} diverges

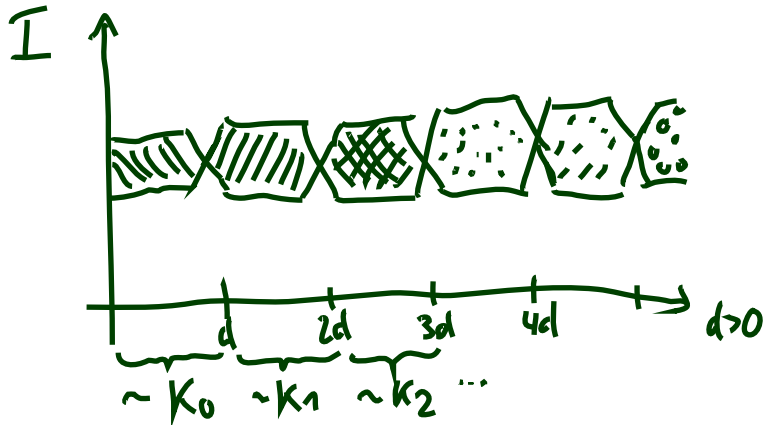
iff

q_{fin} does not occur

exists $\pi: k_0 \vdash k_1 \vdash k_2 \dots$

iff

exist



("I describes π ")

and

$$I \models_0 F(\mathcal{M}) \wedge \neg \Diamond [q_{fin}]$$

$F(\mathcal{M})$ intuitively requires:

- $[0, d]$ encodes $(q_0, 0, 0)$

- $[n \cdot d, (n+1) \cdot d]$ encodes a configuration

- $[n \cdot d, (n+1) \cdot d]$ and $[(n+1) \cdot d, (n+2) \cdot d]$ encode configurations which are in \vdash -relation

- if q_{fin} is reached, we stay there

Reducing Divergence to DC realisability: Idea

- A single configuration K of \mathcal{M} can be encoded in an interval of length 4; being an encoding interval can be **characterised** by a DC formula.
- An interpretation on 'Time' encodes **the** computation of \mathcal{M} if
 - each interval $[4n, 4(n + 1)]$, $n \in \mathbb{N}_0$, **encodes** a configuration K_n ,
 - each two subsequent intervals $[4n, 4(n + 1)]$ and $[4(n + 1), 4(n + 2)]$, $n \in \mathbb{N}_0$, encode configurations $K_n \vdash K_{n+1}$ **in transition relation**.
- Being encoding of the run can be **characterised** by DC formula $F(\mathcal{M})$.
- Then \mathcal{M} **diverges** if and only if $F(\mathcal{M}) \wedge \neg \diamond [q_{fin}]$ is realisable from 0.

Encoding Configurations

- We use $\text{Obs} = \{\text{obs}\}$ with $D(\text{obs}) = Q_{\mathcal{M}} \dot{\cup} \{C_1, C_2, B, X\}$.

set of states of \mathcal{M}

disjoint union

abbrev. for $\tau_{\text{obs}} = q$?

Examples:

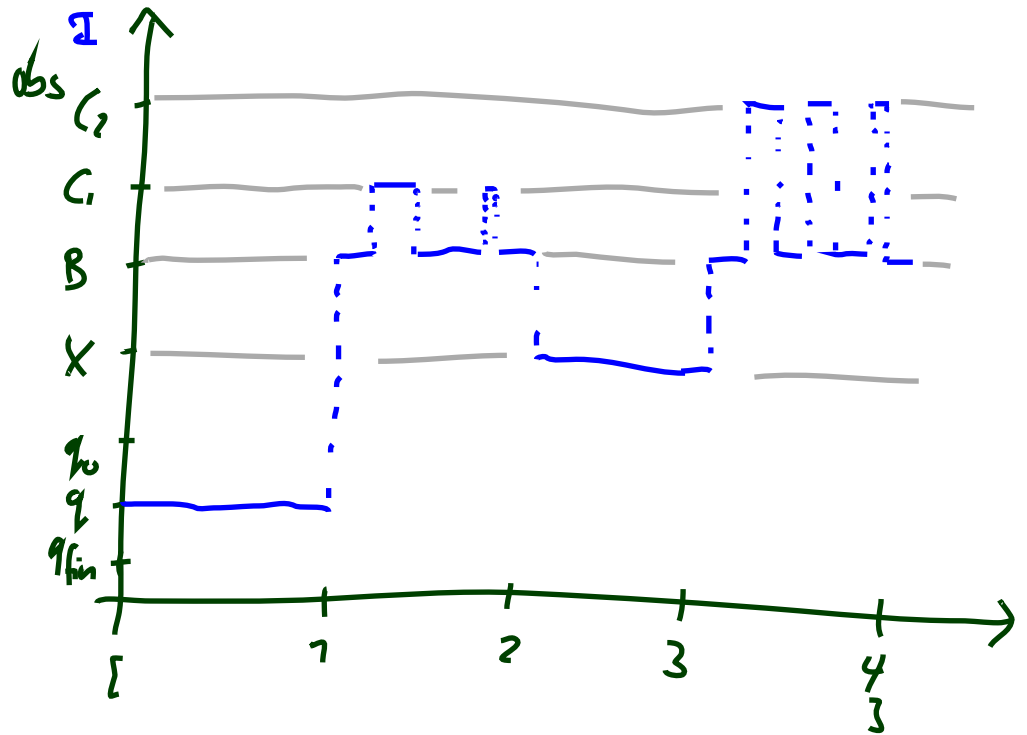
- $K = (q, 2, 3)$

$$\left(\begin{array}{c} [q] \\ \wedge \\ \ell = 1 \end{array} \right); \left(\begin{array}{c} [B]; [C_1]; [B]; [C_1]; [B] \\ \wedge \\ \ell = 1 \end{array} \right); \left(\begin{array}{c} [X] \\ \wedge \\ \ell = 1 \end{array} \right); \left(\begin{array}{c} [B]; [C_2]; [B]; [C_2]; [B]; [C_2]; [B] \\ \wedge \\ \ell = 1 \end{array} \right)$$

- $K_0 = (q_0, 0, 0)$

$$\left(\begin{array}{c} [q_0] \\ \wedge \\ \ell = 1 \end{array} \right); \left(\begin{array}{c} [B] \\ \wedge \\ \ell = 1 \end{array} \right); \left(\begin{array}{c} [X] \\ \wedge \\ \ell = 1 \end{array} \right); \left(\begin{array}{c} [B] \\ \wedge \\ \ell = 1 \end{array} \right)$$

or, using abbreviations, $[q_0]^1; [B]^1; [X]^1; [B]^1$.



Construction of $F(\mathcal{M})$

In the following, we give DC formulae describing

- the initial configuration,
- the general form of configurations,
- the transitions between configurations,
- the handling of the final state.

$F(\mathcal{M})$ is the conjunction of all these formulae.

$$F(\mathcal{M}) = \text{init} \wedge \text{keep} \wedge \dots$$

$$\wedge \bigwedge_{q: \text{inc}; q' \in \text{ProgM}} F(q: \text{inc}; q')$$

$$\wedge \bigwedge_{q: \text{dec}; q', q'' \in \text{ProgM}} F(q: \text{dec}; q', q'')$$

Initial and General Configurations

$$\text{init} : \iff (\ell \geq 4 \implies [q_0]^1 ; [B]^1 ; [X]^1 ; [B]^1 ; \text{true})$$

$$\begin{aligned} \text{keep} : \iff & \Box([Q]^1 ; [B \vee C_1]^1 ; [X]^1 ; [B \vee C_2]^1 ; \ell = 4 \\ & \implies \ell = 4 ; [Q]^1 ; [B \vee C_1]^1 ; [X]^1 ; [B \vee C_2]^1) \end{aligned}$$

where $Q := \neg(X \vee C_1 \vee C_2 \vee B)$.

$$\begin{aligned} & \Box \left(\begin{array}{cccc|c} \Gamma Q \uparrow & \Gamma B \vee C_1 \uparrow & \Gamma X \uparrow & \Gamma B \vee C_2 \uparrow & \ell = 4 \\ \ell = 1 & \ell = 1 & \ell = 1 & \ell = 1 & \end{array} \right) \\ & \implies \left(\begin{array}{c|cccc} \ell = 4 & \Gamma Q \uparrow & \Gamma B \vee C_1 \uparrow & \Gamma X \uparrow & \Gamma B \vee C_2 \uparrow \\ \ell = 1 & \ell = 1 & \ell = 1 & \ell = 1 & \ell = 1 \end{array} \right) \end{aligned}$$

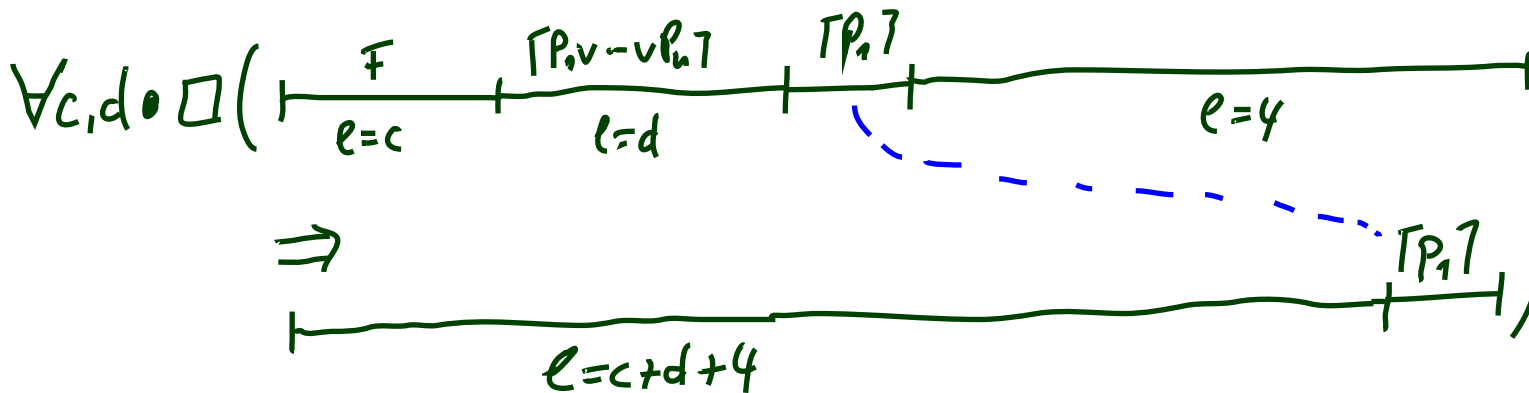
Auxiliary Formula Pattern copy

\swarrow formula \swarrow state assertions
 $copy(F, \{P_1, \dots, P_n\}) : \iff$

$$\forall c, d \bullet \Box((F \wedge \ell = c); ([P_1 \vee \dots \vee P_n] \wedge \ell = d); [P_1]; \ell = 4 \implies \ell = c + d + 4; [P_1])$$

$\wedge \dots$

$$\wedge \forall c, d \bullet \Box((F \wedge \ell = c); ([P_1 \vee \dots \vee P_n] \wedge \ell = d); [P_n]; \ell = 4 \implies \ell = c + d + 4; [P_n])$$



$q : inc_1 : q'$ (Increment)

(i) Keep rest of first counter \overline{F} $\{P_1, P_2\}$
 $copy(\overbrace{[q]^1 ; [B \vee C_1] ; [C_1]}^{\overline{F}}, \underbrace{\{B, C_1\}}_{\{P_1, P_2\}})$

(ii) Leave second counter unchanged

$copy(\underbrace{[q]^1 ; [B \vee C_1] ; [X]^1}_{\overline{F}}, \underbrace{\{B, C_2\}}_{\{P_1, P_2\}})$

$q : dec_1 : q', q''$ (Decrement)

(i) If zero

$$\Box([\![q]\!]^1 ; [\![B]\!]^1 ; [\![X]\!]^1 ; [\![B \vee C_2]\!]^1 ; \ell = 4 \implies \ell = 4 ; [\![q']\!]^1 ; [\![B]\!]^1 ; true)$$

(ii) Decrement counter

$$\forall d \bullet \Box([\![q]\!]^1 ; ([\![B]\!] ; [\![C_1]\!] \wedge \ell = d) ; [\![B]\!] ; [\![B \vee C_1]\!] ; [\![X]\!]^1 ; [\![B \vee C_2]\!]^1 ; \ell = 4 \\ \implies \ell = 4 ; [\![q'']\!]^1 ; [\![B]\!]^d ; true)$$

(iii) Keep rest of first counter

$$copy([\![q]\!]^1 ; [\![B]\!] ; [\![C_1]\!] ; [\![B_1]\!], \{B, C_1\})$$

(iv) Leave second counter unchanged

$$copy([\![q]\!]^1 ; [\![B \vee C_1]\!] ; [\![X]\!]^1, \{B, C_2\})$$

Final State

$copy(\lceil q_{fin} \rceil^1 ; \lceil B \vee C_1 \rceil^1 ; \lceil X \rceil ; \lceil B \vee C_2 \rceil^1, \{q_{fin}, B, X, C_1, C_2\})$

Satisfiability

- Following [Chaochen and Hansen, 2004] we can observe that \mathcal{M} **halts if and only if** the DC formula $F(\mathcal{M}) \wedge \diamond[q_{fin}]$ is **satisfiable**.

This yields

Theorem 3.11. The satisfiability problem for DC with continuous time is undecidable.

(It is semi-decidable.)

- Furthermore, by taking the contraposition, we see
 \mathcal{M} **diverges if and only if** \mathcal{M} does not **halt**
if and only if $F(\mathcal{M}) \wedge \neg \diamond[q_{fin}]$ is **not** satisfiable.
- Thus whether a DC formula is **not satisfiable** is not decidable, not even semi-decidable.

- By Remark 2.13, F is valid iff $\neg F$ is not satisfiable, so

Corollary 3.12. The validity problem for DC with continuous time is undecidable, not even semi-decidable.

- This provides us with an alternative proof of Theorem 2.23 (“there is no sound and complete proof system for DC”):
 - **Suppose** there were such a calculus \mathcal{C} .
 - By Lemma 2.22 it is semi-decidable whether a given DC formula F is a theorem in \mathcal{C} .
 - By the soundness and completeness of \mathcal{C} , F is a theorem in \mathcal{C} **if and only if** F is valid.
 - Thus it is semi-decidable whether F is valid. **Contradiction.**

Discussion

- Note: the DC fragment defined by the following grammar is **sufficient** for the reduction

$$F ::= [P] \mid \neg F_1 \mid F_1 \vee F_2 \mid F_1 ; F_2 \mid \ell = 1 \mid \ell = x \mid \forall x \bullet F_1,$$

P a state assertion, x a global variable.

- Formulae used in the reduction are abbreviations:

$$\ell = 4 \iff \ell = 1 ; \ell = 1 ; \ell = 1 ; \ell = 1$$

$$\ell \geq 4 \iff \ell = 4 ; \text{true}$$

$$\ell = x + y + 4 \iff \ell = x ; \ell = y ; \ell = 4$$

- Length 1 is not necessary — we can use $\ell = z$ instead, with fresh z .
- This is RDC augmented by “ $\ell = x$ ” and “ $\forall x$ ”, which we denote by **RDC** + $\ell = x, \forall x$.

References

References

- [Chaochen and Hansen, 2004] Chaochen, Z. and Hansen, M. R. (2004). *Duration Calculus: A Formal Approach to Real-Time Systems*. Monographs in Theoretical Computer Science. Springer-Verlag. An EATCS Series.
- [Olderog and Dierks, 2008] Olderog, E.-R. and Dierks, H. (2008). *Real-Time Systems - Formal Specification and Automatic Verification*. Cambridge University Press.