Real-Time Systems

Lecture 7: DC Properties II

2012-06-05

Dr. Bernd Westphal

Albert-Ludwigs-Universität Freiburg, Germany

Contents & Goals

Last Lecture:

- RDC in discrete time
- 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 continuous time.
  - What's the idea of the considered (un)decidability proofs?
- **Content:**
  - Undecidable problems of DC in continuous time

(Variants of) RDC in Continuous Time

Recall: Restricted DC (RDC)

$\ell = 1 \equiv \lceil 1 \rceil \wedge \neg(\lceil 1 \rceil ; \lceil 1 \rceil)$

$$F ::= \lceil P \rceil \mid \neg F_1 \mid F_1 \vee F_2 \mid F_1 \,;\, F_2$$

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

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

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

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 $F(\mathcal{M}) := encoding(\mathcal{M})$
- such that

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

  $$F(\mathcal{M}) \wedge \neg\Diamond\lceil q_{fin}\rceil$$

  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

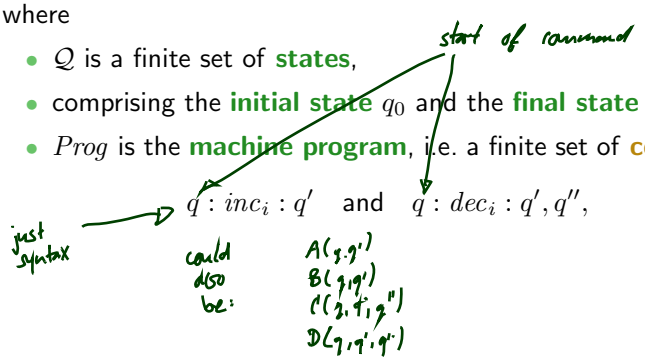
A **two-counter** machine is a structure

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

where

- $\mathcal{Q}$ 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

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



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

2CM Configurations and Computations

- a **configuration** of $\mathcal{M}$ is a triple $K = (q, n_1, n_2) \in \mathcal{Q} \times \mathbb{N}_0 \times \mathbb{N}_0$.

2CM Configurations and Computations

- 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 (!) **computation** of $\mathcal{M}$ is a finite sequence of the form ("$\mathcal{M}$ **halts**")

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

or an infinite sequence of the form ("$\mathcal{M}$ **diverges**")

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

2CM Configurations and Computations

- 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 (!) **computation** of $\mathcal{M}$ is a finite sequence of the form ("$\mathcal{M}$ **halts**")

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

or an infinite sequence of the form ("$\mathcal{M}$ **diverges**")

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

- The **transition relation** "$\vdash$" on configurations is defined as follows:

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

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 $K = (q, n_1, n_2) \in \mathcal{Q} \times \mathbb{N}_0 \times \mathbb{N}_0$.
-

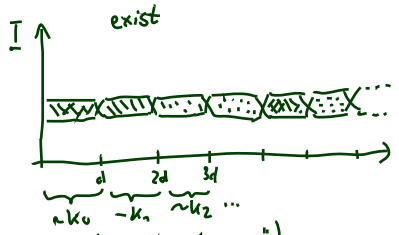
Command	Semantics: $K \vdash K'$
$q : inc_1 : q'$ $q : dec_1 : q', q''$	$(q, n_1, n_2) \vdash (q', n_1 + 1, n_2)$ $(q, 0, n_2) \vdash (q', 0, n_2)$ $(q, n_1 + 1, n_2) \vdash (q'', n_1, n_2)$
$q : inc_2 : q'$ $q : dec_2 : q', q''$	$(q, n_1, n_2) \vdash (q', n_1, n_2 + 1)$ $(q, n_1, 0) \vdash (q', n_1,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

$Q = \{q_0, q_1, q_{fin}\}$
$Prog = \{q_0 : inc_1 : q_1,$
$q_1 : inc_2 : q_{fin}\}$
$(q_0,0,0)$
$\vdash$
$(q_1,1,0)$
$\vdash$
$(q_{fin},1,1)$
↳ machine halts

$\hat{Q} = \{\hat{q}_0, \hat{q}_{fin}\}$, $\hat{Prog} = \{\hat{q}_0 : inc_1 : \hat{q}_0\}$
$(\hat{q}_0,0,0)$
$\vdash$
$(\hat{q}_0,1,0)$
$\vdash$
$(\hat{q}_0,2,0)$
$\vdots$
↳ machine diverges

Reducing Divergence to DC realisability: Idea In Pictures

2CM $\mathcal{M}$ diverges
iff
exists $\pi : K_0 \vdash K_1 \vdash K_2 \ldots$ never q_{fin}
iff
exist



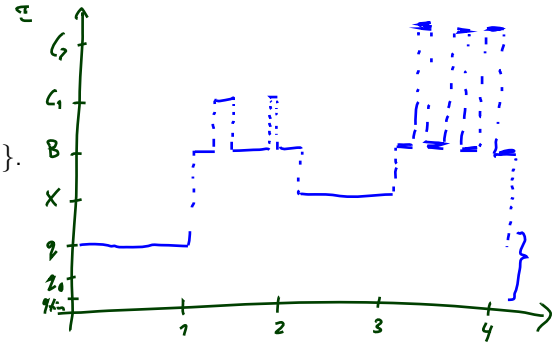
("I describes π")
and
$I \models_0 \mathcal{F}(\mathcal{M}) \wedge \neg \Diamond \lceil q_{fin} \rceil$

$\mathcal{F}(\mathcal{M})$ intuitively requires:
- $[n \cdot d, (n+1) \cdot d]$ encodes a configuration
- $[n \cdot d, (n+1) \cdot d]$ and $[(n+1) \cdot d, (n+2) \cdot d]$ are in $\vdash$-relation
- $[0, d]$ encodes $(q_0, 0, 0)$
- 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 $F(\mathcal{M}) \wedge \neg\Diamond\lceil q_{fin}\rceil$ is realisable from 0.

Encoding Configurations



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

states of $\mathcal{M}$ — disjoint union — abbreviates $\lceil \text{obs} = q \rceil$

Examples:

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

$$\begin{pmatrix} \lceil q\rceil \\ \wedge \\ \ell = 1 \end{pmatrix}; \begin{pmatrix} \lceil B\rceil ; \lceil C_1\rceil ; \lceil B\rceil ; \lceil C_1\rceil ; \lceil B\rceil \\ \wedge \\ \ell = 1 \end{pmatrix}; \begin{pmatrix} \lceil X\rceil \\ \wedge \\ \ell = 1 \end{pmatrix}; \begin{pmatrix} \lceil B\rceil ; \lceil C_2\rceil ; \lceil B\rceil ; \lceil C_2\rceil ; \lceil B\rceil ; \lceil C_2\rceil ; \lceil B\rceil \\ \wedge \\ \ell = 1 \end{pmatrix}$$

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

$$\begin{pmatrix} \lceil q_0\rceil \\ \wedge \\ \ell = 1 \end{pmatrix}; \begin{pmatrix} \lceil B\rceil \\ \wedge \\ \ell = 1 \end{pmatrix}; \begin{pmatrix} \lceil X\rceil \\ \wedge \\ \ell = 1 \end{pmatrix}; \begin{pmatrix} \lceil B\rceil \\ \wedge \\ \ell = 1 \end{pmatrix}$$

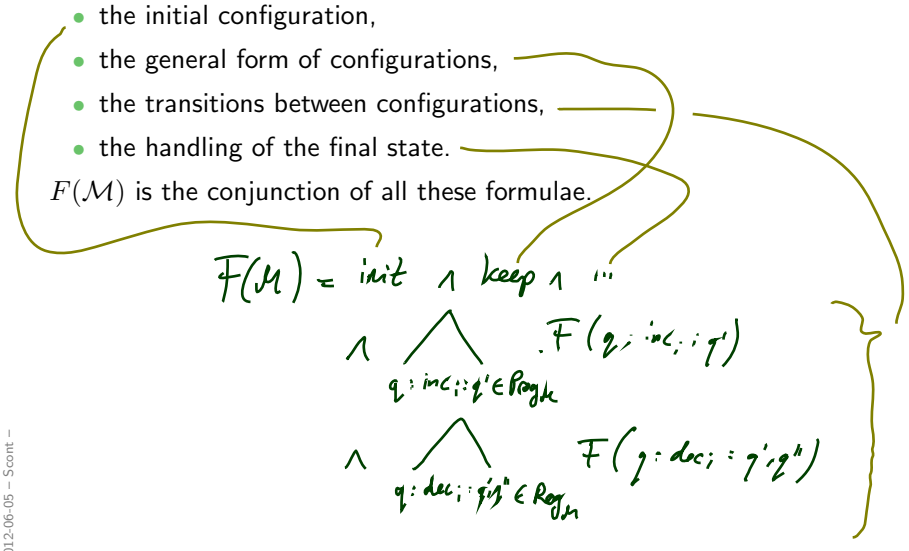
or, using abbreviations, $\lceil q_0\rceil^1 ; \lceil B\rceil^1 ; \lceil X\rceil^1 ; \lceil B\rceil^1$.

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.

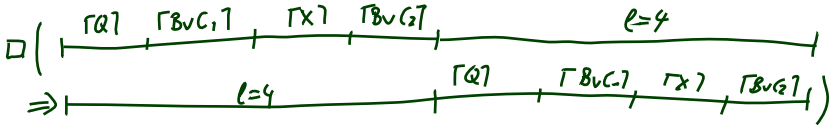


Initial and General Configurations

$$init :\Longleftrightarrow (\ell \geq 4 \implies \lceil q_0 \rceil^1 ; \lceil B \rceil^1 ; \lceil X \rceil^1 ; \lceil B \rceil^1 ; true)$$

$$keep :\Longleftrightarrow \Box(\lceil Q \rceil^1 ; \lceil B \vee C_1 \rceil^1 ; \lceil X \rceil^1 ; \lceil B \vee C_2 \rceil^1 ; \ell = 4 \implies \ell = 4 ; \lceil Q \rceil^1 ; \lceil B \vee C_1 \rceil^1 ; \lceil X \rceil^1 ; \lceil B \vee C_2 \rceil^1)$$

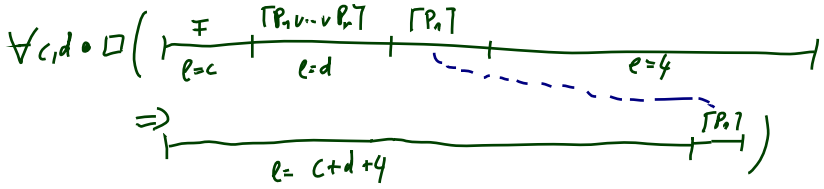
where $Q := \neg(X \vee C_1 \vee C_2 \vee B)$.



Auxiliary Formula Pattern copy

formula state assertions

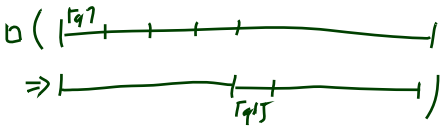
$$copy(F, \{P_1, \ldots, P_n\}) :\Longleftrightarrow$$
$$\forall\, c, d \bullet \Box((F \wedge \ell = c)\,;\,(\lceil P_1 \vee \cdots \vee P_n \rceil \wedge \ell = d)\,;\,\lceil P_1 \rceil\,;\,\ell = 4$$
$$\Longrightarrow\ \ell = c + d + 4\,;\,\lceil P_1 \rceil$$
$$\wedge\ \ldots$$
$$\wedge\ \forall\, c, d \bullet \Box((F \wedge \ell = c)\,;\,(\lceil P_1 \vee \cdots \vee P_n \rceil \wedge \ell = d)\,;\,\lceil P_n \rceil\,;\,\ell = 4$$
$$\Longrightarrow\ \ell = c + d + 4\,;\,\lceil P_n \rceil$$



q : *inc*$_1$: *q'* (Increment)

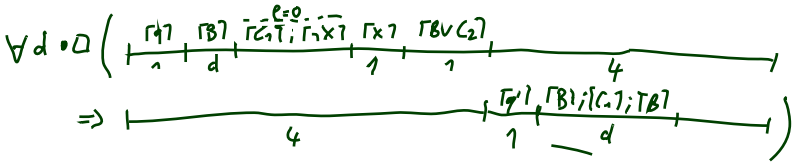
(i) Change state

$$\Box(\lceil q \rceil^1\,;\,\lceil B \vee C_1 \rceil^1\,;\,\lceil X \rceil^1\,;\,\lceil B \vee C_2 \rceil^1\,;\,\ell = 4\ \Longrightarrow\ \ell = 4\,;\,\lceil q' \rceil^1\,;\,true)$$



(ii) Increment counter

$$\forall\, d \bullet \Box(\lceil q \rceil^1\,;\,\lceil B \rceil^d\,;\,(\ell = 0 \vee \lceil C_1 \rceil\,;\,\lceil \neg X \rceil)\,;\,\lceil X \rceil^1\,;\,\lceil B \vee C_2 \rceil^1\,;\,\ell = 4$$
$$\Longrightarrow\ \ell = 4\,;\,\lceil q' \rceil^1\,;\,(\lceil B \rceil\,;\,\lceil C_1 \rceil\,;\,\lceil B \rceil \wedge \ell = d)\,;\,true$$



$q : inc_1 : q'$ (Increment)

(i) Keep rest of first counter

$$copy(\lceil q \rceil^1 ; \lceil B \vee C_1 \rceil ; \lceil C_1 \rceil, \{B, C_1\})$$

(ii) Leave second counter unchanged

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

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

(i) If zero

$$\Box(\lceil q \rceil^1 ; \lceil B \rceil^1 ; \lceil X \rceil^1 ; \lceil B \vee C_2 \rceil^1 ; \ell = 4 \implies \ell = 4 ; \lceil q' \rceil^1 ; \lceil B \rceil^1 ; true)$$

(ii) Decrement counter

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

(iii) Keep rest of first counter

$$copy(\lceil q \rceil^1 ; \lceil B \rceil ; \lceil C_1 \rceil ; \lceil B_1 \rceil, \{B, C_1\})$$

(iv) Leave second counter unchanged

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

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 $F(\mathcal{M}) \wedge \Diamond \lceil q_{fin} \rceil$ 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 $F(\mathcal{M}) \wedge \neg \Diamond \lceil q_{fin} \rceil$ is **not** satisfiable.

- Thus whether a DC formula is **not satisfiable** is not decidable, not even semi-decidable.

Validity

- 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

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

  P a state assertion, x a global variable.

- Formulae used in the reduction are abbreviations:

$$\begin{aligned} \ell = 4 &\iff \ell = 1 \,;\, \ell = 1 \,;\, \ell = 1 \,;\, \ell = 1 \\ \ell \geq 4 &\iff \ell = 4 \,;\, \mathit{true} \\ \ell = x + y + 4 &\iff \ell = x \,;\, \ell = y \,;\, \ell = 4 \end{aligned}$$

- 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.