Real-Time Systems

Lecture 07: DC Implementables

2014-06-03

Dr. Bernd Westphal

Albert-Ludwigs-Universität Freiburg, Germany

Contents & Goals

Last Lectures:
- Semantical Correctness Proof

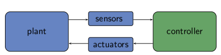
This Lecture:
- **Educational Objectives:** Capabilities for following tasks/questions.
  - What does this standard forms mean? Give a satisfying interpretation.
  - What are implementables? What is a control automaton?
  - Please specify (and prove correct) a controller which satisfies this requirement.
- **Content:**
  - DC Standard Forms
  - Control Automata
  - DC Implementables
  - Example

DC Implementables

Requirements vs. Implementations

- **Problem:** in general, a DC requirement doesn't tell **how** to achieve it, how to build a controller/write a program which ensures it.
- What a controller (clearly) can do is:
  - consider inputs now,
  - change (local) state, or
  - wait,
  - set outputs now.

  (But not, e.g., consider future inputs now.)

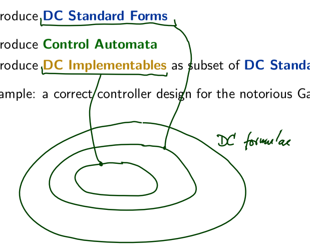


- So, if we have
  - a DC requirement 'Req',
  - a description 'Impl' in DC, which "uses" **just these** operations,

  then
  - proving correctness amounts to proving $\models_0$ Impl $\implies$ Req (**in DC**)
  - and we (more or less) know how to program (the correct) 'Impl' in a PLC language, or in C on a real-time OS, or or or...

Approach: Control Automata and DC Impl'bles

Plan:
- Introduce **DC Standard Forms**
- Introduce **Control Automata**
- Introduce **DC Implementables** as subset of **DC Standard Forms**
- Example: a correct controller design for the notorious Gas Burner



DC Standard Forms: Followed-by

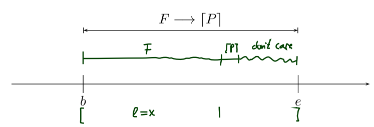
In the following: F is a DC **formula**, P a **state assertion**, θ a **rigid term**.

- **Followed-by:**

$$F \longrightarrow \lceil P \rceil :\iff \neg\Diamond(F\,;\lceil\neg P\rceil) \iff \Box\neg(F\,;\lceil\neg P\rceil)$$

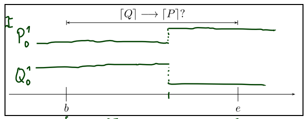
in other symbols

$$\forall x \bullet \Box\big((F \land \ell = x)\,;\ell > 0\big) \implies \big((F \land \ell = x)\,;\lceil P\rceil\,;true\big)$$



DC Standard Forms: Followed-by Examples

$$\forall x \bullet \Box((F \wedge \ell = x) \,;\, \ell > 0 \implies (F \wedge \ell = x) \,;\, \lceil P \rceil \,;\, true)$$



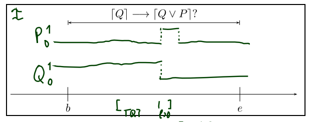
[⌈Q⌉ ℓ=x | ℓ>0 ⌈P⌉;true]

[⌈Q⌉ ℓ=x ·|···] ℓ>0 ¬(⌈P⌉;true)

↳ $\mathcal{I}$ does not satisfy ⌈Q⌉ ⟶ ⌈P⌉

DC Standard Forms: Followed-by Examples

$$\forall x \bullet \Box((F \wedge \ell = x) \,;\, \ell > 0 \implies (F \wedge \ell = x) \,;\, \lceil P \rceil \,;\, true)$$



⌈P∨Q⌉ ⇐ ⌈P⌉ ✓

[⌈Q⌉ | ℓ>0] ⌈P⌉(⇒⌈P∨Q⌉) ✓

↳ $\mathcal{I}$ satisfies ⌈Q⌉ ⟶ ⌈Q∨P⌉

DC Standard Forms: Followed-by Examples

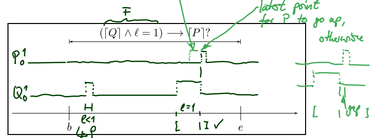
$$\forall x \bullet \Box((F \wedge \ell = x) \,;\, \ell > 0 \implies (F \wedge \ell = x) \,;\, \lceil P \rceil \,;\, true)$$

~ ⌈Q⌉ ∧ ℓ=1 ∧ ℓ=x

"let ∫P be as small as possible":

ok, but not smallest ∫P

latest point for P to go up, otherwise



ℓ<1 ↳ P need not change

[-|-] ↳ not ⌈Q⌉ and ℓ=1

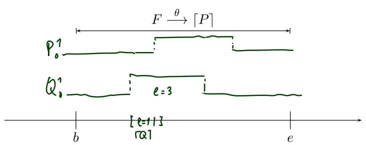
[∄ ↳ not ; ℓ>0

↳ trivially satisfied

DC Standard Forms: (Timed) leads-to

- **(Timed) leads-to:**

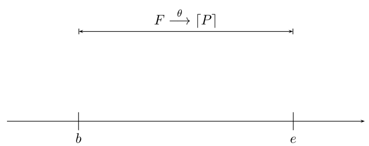
$$F \xrightarrow{\theta} \lceil P \rceil :\iff (F \wedge \ell = \theta) \longrightarrow \lceil P \rceil$$



DC Standard Forms: (Timed) up-to

- **(Timed) up-to:**

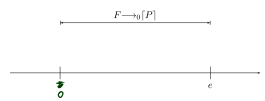
$$F \xrightarrow{\leq \theta} \lceil P \rceil :\iff (F \wedge \ell \leq \theta) \longrightarrow \lceil P \rceil$$



DC Standard Forms: Initialisation

- **Followed-by-initially:**

$$F \longrightarrow_0 \lceil P \rceil :\iff \neg(F \,;\, \lceil \neg P \rceil)$$



- **(Timed) up-to-initially:**

$$F \xrightarrow{\leq \theta}_0 \lceil P \rceil :\iff (F \wedge \ell \leq \theta) \longrightarrow_0 \lceil P \rceil$$

- **Initialisation:**

$$\sqcap \vee \lceil P \rceil \,;\, true$$

Control Automata

Examples: $T = y \vee T = g$ basic phase, phase

- Let $X_1, \ldots, X_k$ be k state variables ranging over **finite** domains $\mathcal{D}(X_1), \ldots, \mathcal{D}(X_k)$.
- With a DC formula 'Impl' ranging over $X_1, \ldots, X_k$ we have a **system of k control automata**.
- 'Impl' is typically a conjunction of **DC implementables**.

$T = g \wedge B = p$ basic phase, not a phase, different observables!

- A state assertion of the form

$$X_i = d_i, \quad d_i \in \mathcal{D}(X_i),$$

which constrains the values of X_i, is called **basic phase** of X_i.
- A **phase** of X_i is a Boolean combination of basic phases of X_i.
- **Abbreviations:**
  - Write X_i instead of $X_i = 1$, if X_i is Boolean.
  - Write d_i instead of $X_i = d_i$, if $\mathcal{D}(X_i)$ is disjoint from $\mathcal{D}(X_j)$, $i \neq j$.

Control Automata: Example

Model of Gas Burner controller as a system of four control automata:

- H Boolean, representing **heat request**, (input)
- F Boolean, representing **flame**, (input)
- C with $\mathcal{D}(C) = \{\text{idle}, \text{purge}, \text{ignite}, \text{burn}\}$, representing the (status of the) **controller**, (local) new!
- G Boolean, representing **gas valve**. (output)

controller reads inputs, sets its local status accordingly and writes outputs

- **Basic phase** of C:

$$C = \text{purge} \qquad \text{(or only: purge)}$$

- **Phase** of C:

$$\text{purge} \vee \text{idle}$$

DC Implementables

- DC Implementables are special patterns of DC Standard Forms (due to A.P. Ravn).
- Within one pattern,
  - $\pi, \pi_1, \ldots, \pi_n$, $n \geq 0$, denote **phases** of **the same** state variable X_i,
  - φ denotes a state assertion not depending on X_i.
- θ denotes a **rigid** term.
- **Initialisation**:

$$\lceil\rceil \vee \lceil \pi \rceil \,;\, true$$

- **Sequencing**:

$$\lceil \pi \rceil \longrightarrow \lceil \pi \vee \pi_1 \vee \cdots \vee \pi_n \rceil$$

- **Progress**:

$$\lceil \pi \rceil \xrightarrow{\theta} \lceil \neg \pi \rceil$$

DC Implementables Cont'd

- **Bounded Stability**:

$$\lceil \neg \pi \rceil \,;\, \lceil \pi \wedge \varphi \rceil \xrightarrow{\leq \theta} \lceil \pi \vee \pi_1 \vee \cdots \vee \pi_n \rceil$$

- **Unbounded Stability**:

$$\lceil \neg \pi \rceil \,;\, \lceil \pi \wedge \varphi \rceil \longrightarrow \lceil \pi \vee \pi_1 \vee \cdots \vee \pi_n \rceil$$

- **Bounded initial stability**:

$$\lceil \pi \wedge \varphi \rceil \xrightarrow{\leq \theta}_0 \lceil \pi \vee \pi_1 \vee \cdots \vee \pi_n \rceil$$

- **Unbounded initial stability**:

$$\lceil \pi \wedge \varphi \rceil \longrightarrow_0 \lceil \pi \vee \pi_1 \vee \cdots \vee \pi_n \rceil$$

Specification by DC Implementables

- Let $X_1, \ldots, X_k$ be a system of k control automata.
- Let 'Impl' be a conjunction of **DC implementables**.
- Then 'Impl' **specifies** all interpretations $\mathcal{I}$ of $X_1, \ldots, X_k$ and all valuations $\mathcal{V}$ such that

$$\mathcal{I}, \mathcal{V} \models_0 \text{Impl}$$

- Hmm: And what does this have to do with controllers...?

Example: Gas Burner

Recall: Control Automata

Model of Gas Burner controller as a system of four control automata:

- H : Boolean, representing **heat request**, (input)
- F : Boolean, representing **flame**, (input)
- C with $\mathcal{D}(C) = \{\text{idle}, \text{purge}, \text{ignite}, \text{burn}\}$, representing the **controller**, (local)
- G : Boolean, representing **gas valve**. (output)

Gas Burner Controller Specification

$\lceil\rceil \vee \lceil \text{idle} \rceil ; true, \quad \lceil\rceil \vee \lceil \neg H \rceil ; true, \quad \lceil\rceil \vee \lceil \neg F \rceil ; true, \quad \lceil\rceil \vee \lceil \neg G \rceil ; true$ (Init-1 - 4)

$\lceil \text{idle} \rceil \longrightarrow \lceil \text{idle} \vee \text{purge} \rceil$ (Seq-1)

$\lceil \text{purge} \rceil \longrightarrow \lceil \text{purge} \vee \text{ignite} \rceil$ (Seq-2)

$\lceil \text{ignite} \rceil \longrightarrow \lceil \text{ignite} \vee \text{burn} \rceil$ (Seq-3)

$\lceil \text{burn} \rceil \longrightarrow \lceil \text{burn} \vee \text{idle} \rceil$ (Seq-4)

timing

$\lceil \text{purge} \rceil \xrightarrow{30+\varepsilon} \lceil \neg\text{purge} \rceil$ (Prog-1)

$\lceil \text{ignite} \rceil \xrightarrow{0.5+\varepsilon} \lceil \neg\text{ignite} \rceil$ (Prog-2)

input effect

$\lceil \text{idle} \wedge H \rceil \xrightarrow{\varepsilon} \lceil \neg\text{idle} \rceil$ (Syn-1)

$\lceil \text{burn} \wedge (\neg H \vee \neg F) \rceil \xrightarrow{\varepsilon} \lceil \neg\text{burn} \rceil$ (Syn-2)

$\lceil G \wedge (\text{idle} \vee \text{purge}) \rceil \xrightarrow{\varepsilon} \lceil \neg G \rceil$ (Syn-3)

$\lceil \neg G \wedge (\text{ignite} \vee \text{burn}) \rceil \xrightarrow{\varepsilon} \lceil G \rceil$ (Syn-4)

link local states to output

$\lceil \neg\text{idle} \rceil ; \lceil \text{idle} \wedge \neg H \rceil \longrightarrow \lceil \text{idle} \rceil$ (Stab-1)

$\lceil \text{idle} \wedge \neg H \rceil \longrightarrow_0 \lceil \text{idle} \rceil$ (Stab-1-init)

$\lceil \neg\text{purge} \rceil ; \lceil \text{purge} \rceil \xrightarrow{\leq 30} \lceil \text{purge} \rceil$ (Stab-2)

$\lceil \neg\text{ignite} \rceil ; \lceil \text{ignite} \rceil \xrightarrow{\leq 0.5} \lceil \text{ignite} \rceil$ (Stab-3)

$\lceil \neg\text{burn} \rceil ; \lceil \text{burn} \wedge H \wedge F \rceil \longrightarrow \lceil \text{burn} \rceil$ (Stab-4)

env. assumption

$\lceil F \rceil ; \lceil \neg F \wedge \neg\text{ignite} \rceil \longrightarrow \lceil \neg F \rceil$ (Stab-5)

$\lceil \neg F \wedge \neg\text{ignite} \rceil \longrightarrow_0 \lceil \neg F \rceil$ (Stab-5-init)

$\lceil G \rceil ; \lceil \neg G \wedge (\text{idle} \vee \text{purge}) \rceil \longrightarrow \lceil \neg G \rceil$ (Stab-6)

$\lceil \neg G \wedge (\text{idle} \vee \text{purge}) \rceil \longrightarrow_0 \lceil \neg G \rceil$ (Stab-6-init)

$\lceil \neg G \rceil ; \lceil G \wedge (\text{ignite} \vee \text{burn}) \rceil \longrightarrow \lceil G \rceil$ (Stab-7)

Gas Burner Controller Specification: Untimed

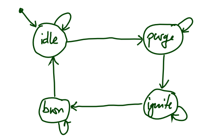
$\lceil\rceil \vee \lceil \text{idle} \rceil ; true$ (Init-1)

$\lceil \text{idle} \rceil \longrightarrow \lceil \text{idle} \vee \text{purge} \rceil$ (Seq-1)

$\lceil \text{purge} \rceil \longrightarrow \lceil \text{purge} \vee \text{ignite} \rceil$ (Seq-2)

$\lceil \text{ignite} \rceil \longrightarrow \lceil \text{ignite} \vee \text{burn} \rceil$ (Seq-3)

$\lceil \text{burn} \rceil \longrightarrow \lceil \text{burn} \vee \text{idle} \rceil$ (Seq-4)



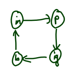
Gas Burner Controller Specification: Timing

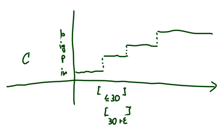
$\lceil \text{purge} \rceil \xrightarrow{30+\varepsilon} \lceil \neg\text{purge} \rceil$ (Prog-1)

$\lceil \text{ignite} \rceil \xrightarrow{0.5+\varepsilon} \lceil \neg\text{ignite} \rceil$ (Prog-2)

$\lceil \neg\text{purge} \rceil ; \lceil \text{purge} \rceil \xrightarrow{\leq 30} \lceil \text{purge} \rceil$ (Stab-2)

$\lceil \neg\text{ignite} \rceil ; \lceil \text{ignite} \rceil \xrightarrow{\leq 0.5} \lceil \text{ignite} \rceil$ (Stab-3)





Gas Burner Controller Specification: Outputs

$\overbrace{\lceil G \wedge \underbrace{(\text{idle} \vee \text{purge})}_{\varphi} \rceil}^{\pi} \xrightarrow{\varepsilon} \overbrace{\lceil \neg G \rceil}^{\neg\pi}$ (Syn-3)

$\lceil \neg G \wedge (\text{ignite} \vee \text{burn}) \rceil \xrightarrow{\varepsilon} \lceil G \rceil$ (Syn-4)

$\lceil G \rceil ; \lceil \neg G \wedge (\text{idle} \vee \text{purge}) \rceil \longrightarrow \lceil \neg G \rceil$ (Stab-6)

$\lceil \neg G \wedge (\text{idle} \vee \text{purge}) \rceil \longrightarrow_0 \lceil \neg G \rceil$ (Stab-6-init)

$\lceil \neg G \rceil ; \lceil G \wedge (\text{ignite} \vee \text{burn}) \rceil \longrightarrow \lceil G \rceil$ (Stab-7)

$\lceil\rceil \vee \lceil \neg G \rceil ; true$

valve closes/opens after ε time units the latest

valve stays closed/open

Gas Burner Controller Specification: Inputs

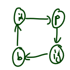
$\lceil \text{idle} \wedge H \rceil \xrightarrow{\varepsilon} \lceil \neg\text{idle} \rceil$ (Syn-1)

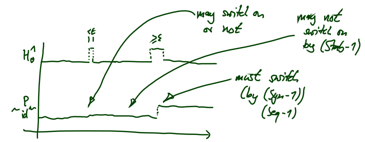
$\lceil \text{burn} \wedge (\neg H \vee \neg F) \rceil \xrightarrow{\varepsilon} \lceil \neg\text{burn} \rceil$ (Syn-2)

$\lceil \neg\text{idle} \rceil ; \lceil \text{idle} \wedge \neg H \rceil \longrightarrow \lceil \text{idle} \rceil$ (Stab-1)

$\lceil \text{idle} \wedge \neg H \rceil \longrightarrow_0 \lceil \text{idle} \rceil$ (Stab-1-init)

$\lceil \neg\text{burn} \rceil ; \lceil \text{burn} \wedge H \wedge F \rceil \longrightarrow \lceil \text{burn} \rceil$ (Stab-4)





Gas Burner Controller Specification: Assumptions

$$\lceil\rceil \vee \lceil \neg H \rceil \,;\, true \qquad \text{(Init-2)}$$

$$\lceil\rceil \vee \lceil \neg F \rceil \,;\, true \qquad \text{(Init-3)}$$

$$\lceil\rceil \vee \lceil \neg G \rceil \,;\, true \qquad \text{(Init-4)}$$

$$\lceil F \rceil \,;\, \lceil \neg F \wedge \neg \mathsf{ignite} \rceil \longrightarrow \lceil \neg F \rceil \qquad \text{(Stab-5)}$$

$$\lceil \neg F \wedge \neg \mathsf{ignite} \rceil \longrightarrow_0 \lceil \neg F \rceil \qquad \text{(Stab-5-init)}$$

no spontaneous flames

References

[Olderog and Dierks, 2008] Olderog, E.-R. and Dierks, H. (2008). *Real-Time Systems - Formal Specification and Automatic Verification*. Cambridge University Press.