Real-Time Systems

Lecture 5: Duration Calculus III

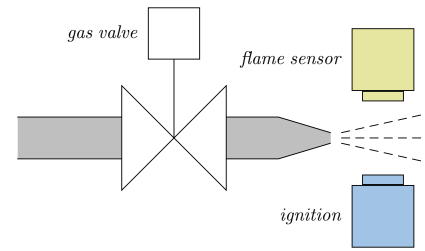
2017-11-09

Dr. Bernd Westphal

Albert-Ludwigs-Universität Freiburg, Germany

Duration Calculus: Preview

- Duration Calculus is an **interval logic**.
- Formulae are evaluated in an (**implicitly given**) interval.



- $G, F, I, H : \{0, 1\}$
- Define $L : \{0, 1\}$ as $G \wedge \neg F$.

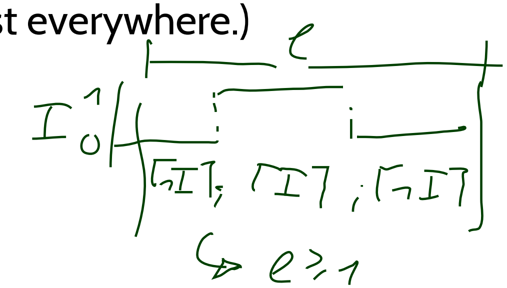
Strangest operators:

- **almost everywhere** – Example: $\lceil G \rceil$

  (Holds in a given interval $[b, e]$ iff the gas valve is open almost everywhere.)
- **chop** – Example: $(\lceil \neg I \rceil \,;\, \lceil I \rceil \,;\, \lceil \neg I \rceil) \implies \ell \geq 1$

  (Ignition phases last at least one time unit.)
- **integral** – Example: $\ell \geq 60 \implies \int L \leq \frac{\ell}{20}$

  (At most 5% leakage time within intervals of at least 60 time units.)



Content

Introduction

- **Observables and Evolutions**
- **Duration Calculus** (DC) ✓
- Semantical Correctness Proofs 5
- DC Decidability 6/7
- DC Implementables
- **PLC-Automata**

$obs : \text{Time} \to \mathscr{D}(obs)$

- **Timed Automata** (TA), Uppaal
- Networks of Timed Automata
- Region/Zone-Abstraction
- TA model-checking
- Extended Timed Automata
- Undecidability Results

$\langle obs_0, \nu_0 \rangle, t_0 \xrightarrow{\lambda_0} \langle obs_1, \nu_1 \rangle, t_1 \ldots$

- **Automatic Verification**...

  ...whether a TA satisfies a DC formula, observer-based
- **Recent Results**:
  - **Timed Sequence Diagrams**, or **Quasi-equal Clocks**, or **Automatic Code Generation**, or …

Content

- **Semantics-based Correctness Proofs**
  - Example: **Gas Burner Controller**
  - **Theorem 2.16**: Des-1 and Des-2 is a correct design wrt. Req
  - **Lemma 2.19**: Des-1 and Des-2 imply a simplified requirement Req-1
  - **Some Laws of the DC Integral Operator**
  - **Lemma 2.17**: Req-1 implies Req

- **Obstacles (in a Non-Ideal World)**
  - requirements may be **unrealisable** without considering plant assumptions
  - **intermediate** design levels
  - **different observables**
  - **proving correctness** may be difficult

- If time permits:
  A Calculus for DC

Specification and Semantics-based Correctness Proofs of Real-Time Systems with DC

Methodology (in an ideal world)

In order to **prove** a controller design **correct** wrt. a **specification**:

(i) Choose **observables** 'Obs'.

(ii) Formalise the **requirements** 'Req'
as a conjunction of DC formulae (over 'Obs').

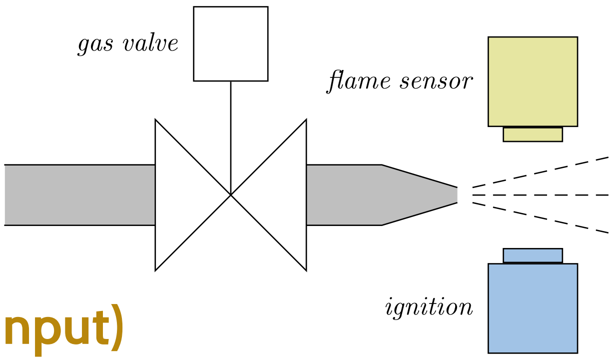
(iii) Formalise a **controller design** 'Ctrl'
as a conjunction of DC formulae (over 'Obs').

(iv) We say 'Ctrl' is **correct** (wrt. 'Req') iff

$$\models_0 \mathsf{Ctrl} \implies \mathsf{Req},$$

so "just" prove $\models_0 \mathsf{Ctrl} \implies \mathsf{Req}$.

Gas Burner Revisited



(i) Choose **observables**:

- $F : \{0,1\}$: value 1 models "flame sensed now" **(input)**
- $G : \{0,1\}$: value 1 models "gas valve is open now" **(output)**
- define $L := G \wedge \neg F$ to model **leakage**

(ii) Formalise the **requirement**:

$$\text{Req} := \Box(\ell \geq 60 \implies 20 \cdot \textstyle\int L \leq \ell)$$

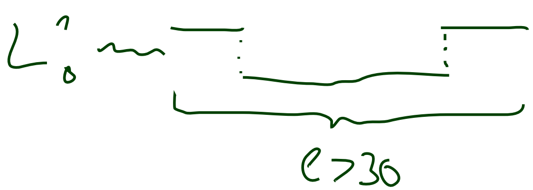
"in each interval of length at least 60 time units, at most 5% of the time leakage"

(iii) Formalise **controller design ideas**:

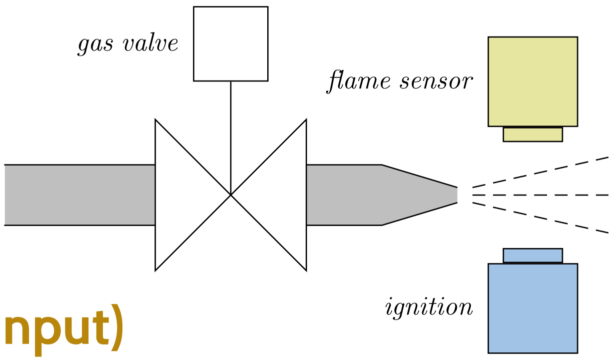
- Des-1 $:= \Box(\lceil L \rceil \implies \ell \leq 1)$

  "**make** leakage phases last for at most one time unit"

- Des-2 $:= \Box(\lceil L \rceil \,;\, \lceil \neg L \rceil \,;\, \lceil L \rceil \implies \ell > 30)$



Gas Burner Revisited



(i) Choose **observables**:

- $F : \{0,1\}$: value 1 models "flame sensed now" **(input)**
- $G : \{0,1\}$: value 1 models "gas valve is open now" **(output)**
- define $L := G \land \neg F$ to model **leakage**

(ii) Formalise the **requirement**:

$$\text{Req} := \Box(\ell \geq 60 \implies 20 \cdot \textstyle\int L \leq \ell)$$

"in each interval of length at least 60 time units, at most 5% of the time leakage"

(iii) Formalise **controller design ideas**:

- $\text{Des-1} := \Box(\lceil L \rceil \implies \ell \leq 1)$

  "**make** leakage phases last for at most one time unit"

- $\text{Des-2} := \Box(\lceil L \rceil \,;\, \lceil \neg L \rceil \,;\, \lceil L \rceil \implies \ell > 30)$

  "**ensure**: non-leakage phases between two leakage phases last at least 30 time units"

(iv) Prove **correctness**, i.e. prove $\models (\text{Des-1} \land \text{Des-2} \implies \text{Req})$.

(Or do we want "$\models_0$"...?)

A Correct Gas Burner Controller Design

$$\text{Req} := \Box(\ell \geq 60 \implies 20 \cdot \textstyle\int L \leq \ell)$$

$$\text{Des-1} := \Box(\lceil L \rceil \implies \ell \leq 1), \quad \text{Des-2} := \Box(\lceil L \rceil \,;\, \lceil \neg L \rceil \,;\, \lceil L \rceil \implies \ell > 30)$$

- A **controller for the gas burner** which guarantees Des-1 and Des-1 is **correct** wrt. Req if:

  $$\models (\text{Des-1} \land \text{Des-2} \implies \text{Req})$$

  (shown in **Theorem 2.16**)

- We do prove (in **Lemma 2.19**)

  $$\models (\text{Des-1} \land \text{Des-2}) \implies \text{Req-1}.$$

  for the the **simplified requirement**

  $$\text{Req-1} := \Box(\ell \leq 30 \implies \textstyle\int L \leq 1).$$

  ("intervals of length at most 30 time units have at most 1 time unit of accumulated leakage")

- Showing

  $$\models \text{Req-1} \implies \text{Req}$$

  (in **Lemma 2.17**) completes the overall proof.

Lemma 2.17

Claim:

$$\models \underbrace{\Box(\ell \leq 30 \implies \textstyle\int L \leq 1)}_{\textsf{Req-1}} \implies \underbrace{\Box(\ell \geq 60 \implies 20 \cdot \textstyle\int L \leq \ell)}_{\textsf{Req}}$$

Proof:

- Assume that ‘Req-1’ holds.
- Let $L_\mathcal{I}$ be any interpretation of L, and $[b, e]$ an interval with $e - b \geq 60$.
- We need to show that

$$20 \cdot \textstyle\int L \leq \ell$$

  evaluates to ‘tt’ on **interval** $[b, e]$ under **interpretation** $\mathcal{I}$ (and any **valuation** $\mathcal{V}$).

- We have

$$\mathcal{I}[\![20 \cdot \textstyle\int L \leq \ell]\!](\mathcal{V}, [b, e]) = \mathrm{tt}$$

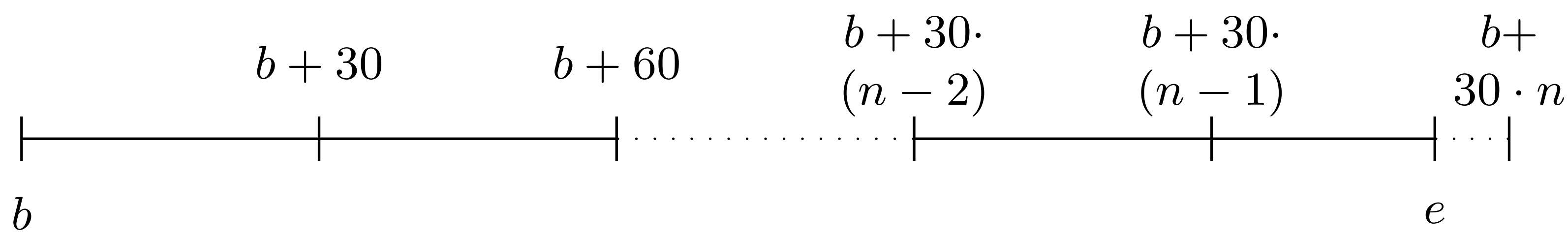
$\iff$ *(by DC semantics)*

$$\hat{20} \mathbin{\hat{\cdot}} \int_b^e L_\mathcal{I}(t)\, dt \mathrel{\hat{\leq}} (e - b)$$

Lemma 2.17 Cont'd

$\models \Box(\ell \le 30 \implies \int L \le 1)$ (Req-1) $\implies \Box(\ell \ge 60 \implies 20 \cdot \int L \le \ell)$ (Req)

- Set $n := \lceil \frac{e-b}{30} \rceil$, i.e. $n \in \mathbb{N}$ with $n - 1 < \frac{e-b}{30} \le n$, and split the interval as follows:



$$20 \cdot \int_b^e L_\mathcal{I}(t)\,dt$$

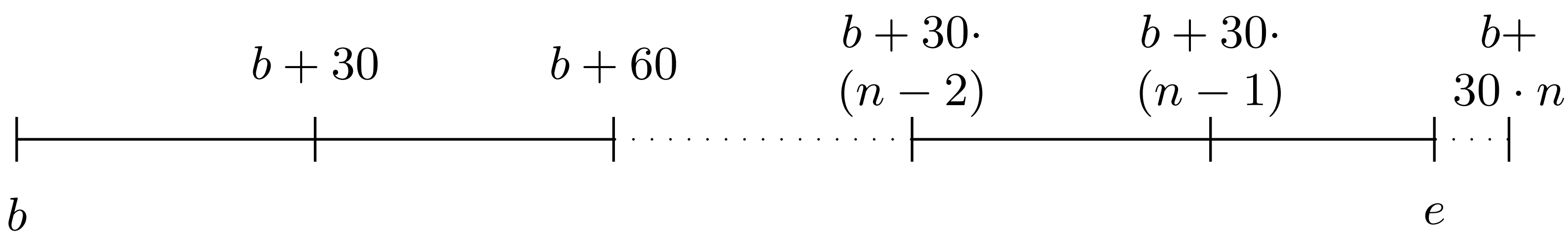
$$= 20 \left(\sum_{i=0}^{n-2} \underbrace{\int_{b+30i}^{b+30(i+1)} L_\mathcal{I}(t)\,dt}_{\le 1} + \underbrace{\int_{b+30(n-1)}^{e} L_\mathcal{I}(t)\,dt}_{\le 1} \right)$$

{Req-1} $\le \left(20 \cdot \sum_{i=0}^{n-2} 1\right) + \left(20 \cdot 1\right)$

Lemma 2.17 Cont'd

$$\models \Box(\ell \leq 30 \implies \textstyle\int L \leq 1)$$ (Req-1)
$$\implies \Box(\ell \geq 60 \implies 20 \cdot \textstyle\int L \leq \ell)$$ (Req)

- Set $n := \lceil \frac{e-b}{30} \rceil$, i.e. $n \in \mathbb{N}$ with $n - 1 < \frac{e-b}{30} \leq n$, $(*)$ and split the interval as follows:



$$
\begin{aligned}
& 20 \cdot \int_b^e L_{\mathcal{I}}(t)\,dt \\
& = 20 \left(\sum_{i=0}^{n-2} \int_{b+30i}^{b+30(i+1)} L_{\mathcal{I}}(t)\,dt + \int_{b+30(n-1)}^{e} L_{\mathcal{I}}(t)\,dt \right) \\
\{\text{Req-1}\} \quad & \leq 20 \cdot \sum_{i=0}^{n-2} 1 + 20 \cdot 1 = 20 \cdot n \\
\{(*)\} \quad & < 20 \cdot \left(\frac{e-b}{30} + 1 \right) = \frac{2}{3}(e-b) + 20 \\
\{e - b \geq 60\} \quad & \leq e - b
\end{aligned}
$$

□

Theorem 2.18.
For all state assertions P and all real numbers $r_1, r_2 \in \mathbb{R}$,

(i) $\models \int P \leq \ell$,

(ii) $\models (\int P = r_1) \,;\, (\int P = r_2) \implies \int P = r_1 + r_2$,

(iii) $\models \lceil \neg P \rceil \implies \int P = 0$,

(iv) $\models \lceil\rceil \implies \int P = 0$.

Lemma 2.19

(i) $\models \int P \leq \ell$, (iii) $\models \lceil \neg P \rceil \implies \int P = 0$,
(ii) $\models (\int P = r_1);(\int P = r_2) \implies \int P = r_1 + r_2$,
(iv) $\models \lceil\rceil \implies \int P = 0$.

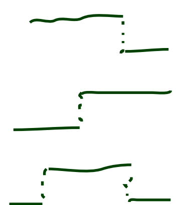
Claim:

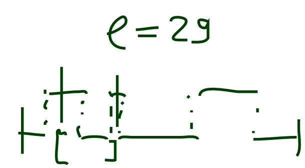
$$\models (\overbrace{\Box(\lceil L \rceil \implies \ell \leq 1)}^{\text{Des-1}} \wedge \overbrace{\Box(\lceil L \rceil ; \lceil \neg L \rceil ; \lceil L \rceil \implies \ell > 30)}^{\text{Des-2}}) \implies \overbrace{\Box(\ell \leq 30 \implies \int L \leq 1)}^{\text{Req-1}}$$

Proof:

$$\ell \leq 30$$

$$\{\text{Des-2}\} \implies \lceil\rceil$$
$$\vee \lceil L \rceil ; (\lceil\rceil \vee \lceil \neg L \rceil)$$
$$\vee \lceil \neg L \rceil ; (\lceil\rceil \vee \lceil L \rceil)$$
$$\vee \lceil \neg L \rceil ; \lceil L \rceil ; \lceil \neg L \rceil$$





Lemma 2.19

(i) $\models \int P \leq \ell$, (iii) $\models \lceil \neg P \rceil \implies \int P = 0$,
(ii) $\models (\int P = r_1);(\int P = r_2) \implies \int P = r_1 + r_2$,
(iv) $\models \lceil\rceil \implies \int P = 0$.

Claim:

$$\models (\overbrace{\Box(\lceil L \rceil \implies \ell \leq 1)}^{\text{Des-1}} \wedge \overbrace{\Box(\lceil L \rceil ; \lceil \neg L \rceil ; \lceil L \rceil \implies \ell > 30)}^{\text{Des-2}}) \implies \overbrace{\Box(\ell \leq 30 \implies \int L \leq 1)}^{\text{Req-1}}$$

Proof:

$$\ell \leq 30$$

$\{$Des-2$\}$ $\implies$ $\lceil\rceil$
$\vee \lceil L \rceil ; (\lceil\rceil \vee \lceil \neg L \rceil)$
$\vee \lceil \neg L \rceil ; (\lceil\rceil \vee \lceil L \rceil)$
$\vee \lceil \neg L \rceil ; \lceil L \rceil ; \lceil \neg L \rceil$

$\{$Des-1$\}$ $\implies$ $\lceil\rceil$
$\vee (\ell \leq 1) ; (\lceil\rceil \vee \lceil \neg L \rceil)$
$\vee \lceil \neg L \rceil ; (\lceil\rceil \vee (\ell \leq 1))$
$\vee \lceil \neg L \rceil ; (\ell \leq 1) ; \lceil \neg L \rceil$

Lemma 2.19

(i) $\models \int P \leq \ell$, (iii) $\models \lceil \neg P \rceil \implies \int P = 0$,
(ii) $\models (\int P = r_1) ; (\int P = r_2) \implies \int P = r_1 + r_2$,
(iv) $\models \lceil\rceil \implies \int P = 0$.

Claim:

$$\models (\overbrace{\Box(\lceil L \rceil \implies \ell \leq 1)}^{\text{Des-1}} \wedge \overbrace{\Box(\lceil L \rceil ; \lceil \neg L \rceil ; \lceil L \rceil \implies \ell > 30)}^{\text{Des-2}}) \implies \overbrace{\Box(\ell \leq 30 \implies \int L \leq 1)}^{\text{Req-1}}$$

Proof:

$$\begin{aligned}
& \ell \leq 30 \\
\{\text{Des-2}\} \implies & \lceil\rceil \\
& \vee \lceil L \rceil ; (\lceil\rceil \vee \lceil \neg L \rceil) \\
& \vee \lceil \neg L \rceil ; (\lceil\rceil \vee \lceil L \rceil) \\
& \vee \lceil \neg L \rceil ; \lceil L \rceil ; \lceil \neg L \rceil \\
\{\text{Des-1}\} \implies & \lceil\rceil \\
& \vee (\ell \leq 1) ; (\lceil\rceil \vee \lceil \neg L \rceil) \\
& \vee \lceil \neg L \rceil ; (\lceil\rceil \vee (\ell \leq 1)) \\
& \vee \lceil \neg L \rceil ; (\ell \leq 1) ; \lceil \neg L \rceil \\
\{(\text{i})\} \implies & \lceil\rceil \\
& \vee (\int L \leq 1) ; (\lceil\rceil \vee \lceil \neg L \rceil) \\
& \vee \lceil \neg L \rceil ; (\lceil\rceil \vee (\int L \leq 1)) \\
& \vee \lceil \neg L \rceil ; (\int L \leq 1) ; \lceil \neg L \rceil
\end{aligned}$$

Lemma 2.19

(i) $\models \int P \leq \ell$, (iii) $\models \lceil \neg P \rceil \implies \int P = 0$,
(ii) $\models \big((\int P = r_1);(\int P = r_2)\big) \implies \big(\int P = r_1 + r_2,\big)$
(iv) $\models \lceil\rceil \implies \int P = 0$.

Claim:

$$\models (\overbrace{\Box(\lceil L \rceil \implies \ell \leq 1)}^{\text{Des-1}} \wedge \overbrace{\Box(\lceil L \rceil \,;\, \lceil \neg L \rceil \,;\, \lceil L \rceil \implies \ell > 30)}^{\text{Des-2}}) \implies \overbrace{\Box(\ell \leq 30 \implies \int L \leq 1)}^{\text{Req-1}}$$

Proof:

$$\ell \leq 30$$

$\{$Des-2$\}$ $\implies$ $\lceil\rceil$
$\vee\ \lceil L \rceil \,;\, (\lceil\rceil \vee \lceil \neg L \rceil)$
$\vee\ \lceil \neg L \rceil \,;\, (\lceil\rceil \vee \lceil L \rceil)$
$\vee\ \lceil \neg L \rceil \,;\, \lceil L \rceil \,;\, \lceil \neg L \rceil$

$\{$Des-1$\}$ $\implies$ $\lceil\rceil$
$\vee\ (\ell \leq 1) \,;\, (\lceil\rceil \vee \lceil \neg L \rceil)$
$\vee\ \lceil \neg L \rceil \,;\, (\lceil\rceil \vee (\ell \leq 1))$
$\vee\ \lceil \neg L \rceil \,;\, (\ell \leq 1) \,;\, \lceil \neg L \rceil$

$\{$(i)$\}$ $\implies$ $\lceil\rceil$
$\vee\ (\int L \leq 1) \,;\, (\lceil\rceil \vee \lceil \neg L \rceil)$
$\vee\ \lceil \neg L \rceil \,;\, (\lceil\rceil \vee (\int L \leq 1))$
$\vee\ \lceil \neg L \rceil \,;\, (\int L \leq 1) \,;\, \lceil \neg L \rceil$

$\{$(iv), (iii)$\}$ $\implies$ $\int L = 0$
$\vee\ (\int L \leq 1) \,;\, (\int L = 0$ ~~$\vee \int L = 0$~~$)$
$\vee\ \int L = 0 \,;\, (\int L = 0 \vee (\int L \leq 1))$
$\vee\ \int L = 0 \,;\, (\int L \leq 1) \,;\, \int L = 0$

Lemma 2.19

(i) $\models \int P \leq \ell$, (iii) $\models \lceil \neg P \rceil \implies \int P = 0$,
(ii) $\models (\int P = r_1) ; (\int P = r_2) \implies \int P = r_1 + r_2$,
(iv) $\models \lceil\rceil \implies \int P = 0$.

Claim:

$$\models (\overbrace{\Box(\lceil L \rceil \implies \ell \leq 1)}^{\text{Des-1}} \wedge \overbrace{\Box(\lceil L \rceil ; \lceil \neg L \rceil ; \lceil L \rceil \implies \ell > 30)}^{\text{Des-2}}) \implies \overbrace{\Box(\ell \leq 30 \implies \int L \leq 1)}^{\text{Req-1}}$$

Proof:

$$\ell \leq 30$$

$$\{\text{Des-2}\} \implies \lceil\rceil \vee \lceil L \rceil ; (\lceil\rceil \vee \lceil \neg L \rceil) \vee \lceil \neg L \rceil ; (\lceil\rceil \vee \lceil L \rceil) \vee \lceil \neg L \rceil ; \lceil L \rceil ; \lceil \neg L \rceil$$

$$\{\text{Des-1}\} \implies \lceil\rceil \vee (\ell \leq 1) ; (\lceil\rceil \vee \lceil \neg L \rceil) \vee \lceil \neg L \rceil ; (\lceil\rceil \vee (\ell \leq 1)) \vee \lceil \neg L \rceil ; (\ell \leq 1) ; \lceil \neg L \rceil$$

$$\{(\text{i})\} \implies \lceil\rceil \vee (\textstyle\int L \leq 1) ; (\lceil\rceil \vee \lceil \neg L \rceil) \vee \lceil \neg L \rceil ; (\lceil\rceil \vee (\int L \leq 1)) \vee \lceil \neg L \rceil ; (\int L \leq 1) ; \lceil \neg L \rceil$$

$$\{(\text{iv}), (\text{iii})\} \implies \textstyle\int L = 0 \vee (\int L \leq 1) ; (\int L = 0 \vee \int L = 0) \vee \int L = 0 ; (\int L = 0 \vee (\int L \leq 1)) \vee \int L = 0 ; (\int L \leq 1) ; \int L = 0$$

$$\{(\text{ii})\} \implies \textstyle\int L = 0 \vee \int L \leq 1 + 0 \vee \int L \leq 0 + 1 \vee \int L \leq 0 + 1 + 0$$

Lemma 2.19

(i) $\models \int P \leq \ell$, (iii) $\models \lceil \neg P \rceil \implies \int P = 0$,
(ii) $\models (\int P = r_1) ; (\int P = r_2) \implies \int P = r_1 + r_2$,
(iv) $\models \lceil\rceil \implies \int P = 0$.

Claim:

$$\models (\overbrace{\Box(\lceil L \rceil \implies \ell \leq 1)}^{\text{Des-1}} \wedge \overbrace{\Box(\lceil L \rceil ; \lceil \neg L \rceil ; \lceil L \rceil \implies \ell > 30)}^{\text{Des-2}}) \implies \overbrace{\Box(\ell \leq 30 \implies \int L \leq 1)}^{\text{Req-1}}$$

Proof:

$$
\begin{aligned}
& \ell \leq 30 \\
\{\text{Des-2}\} \implies & \lceil\rceil \\
& \vee \lceil L \rceil ; (\lceil\rceil \vee \lceil \neg L \rceil) \\
& \vee \lceil \neg L \rceil ; (\lceil\rceil \vee \lceil L \rceil) \\
& \vee \lceil \neg L \rceil ; \lceil L \rceil ; \lceil \neg L \rceil \\
\{\text{Des-1}\} \implies & \lceil\rceil \\
& \vee (\ell \leq 1) ; (\lceil\rceil \vee \lceil \neg L \rceil) \\
& \vee \lceil \neg L \rceil ; (\lceil\rceil \vee (\ell \leq 1)) \\
& \vee \lceil \neg L \rceil ; (\ell \leq 1) ; \lceil \neg L \rceil \\
\{\text{(i)}\} \implies & \lceil\rceil \\
& \vee (\textstyle\int L \leq 1) ; (\lceil\rceil \vee \lceil \neg L \rceil) \\
& \vee \lceil \neg L \rceil ; (\lceil\rceil \vee (\textstyle\int L \leq 1)) \\
& \vee \lceil \neg L \rceil ; (\textstyle\int L \leq 1) ; \lceil \neg L \rceil
\end{aligned}
$$

$$
\begin{aligned}
\{\text{(iv), (iii)}\} \implies & \textstyle\int L = 0 \\
& \vee (\textstyle\int L \leq 1) ; (\textstyle\int L = 0 \vee \int L = 0) \\
& \vee \textstyle\int L = 0 ; (\int L = 0 \vee (\int L \leq 1)) \\
& \vee \textstyle\int L = 0 ; (\int L \leq 1) ; \int L = 0 \\
\{\text{(ii)}\} \implies & \textstyle\int L = 0 \\
& \vee \textstyle\int L \leq 1 + 0 \\
& \vee \textstyle\int L = 0 + 1 \\
& \vee \textstyle\int L \leq 0 + 1 + 0 \\
\implies & \textstyle\int L \leq 1 \qquad \Box
\end{aligned}
$$

Content

- **Semantics-based Correctness Proofs**
  - Example: **Gas Burner Controller**
  - **Theorem 2.16**: Des-1 and Des-2 is a correct design wrt. Req
  - **Lemma 2.19**: Des-1 and Des-2 imply a simplified requirement Req-1
  - **Some Laws of the DC Integral Operator**
  - **Lemma 2.17**: Req-1 implies Req
- **Obstacles (in a Non-Ideal World)**
  - requirements may be **unrealisable** without considering plant assumptions
  - **intermediate** design levels
  - **different observables**
  - **proving correctness** may be difficult
- If time permits:
  A Calculus for DC

Obstacles in Non-Ideal World

Methodology: The World is Not Ideal...

(i) Choose a collection of **observables** 'Obs'.

(ii) Provide **specification** 'Req' (conjunction of DC formulae over 'Obs').

(iii) Provide a description 'Ctrl' of the **controller** (DC formula over 'Obs').

(iv) Prove 'Ctrl' **correct** (wrt. 'Req'), i.e. prove $\models \mathrm{Ctrl} \implies \mathrm{Req}$.

That looks **too simple to be practical**.

Typical **obstacles**:

(i) It may be **impossible** to realise 'Req'
if it doesn't consider properties of the plant.

(ii) There are typically intermediate **design levels** between 'Req' and 'Ctrl'.

(iii) 'Req' and 'Ctrl' may use **different observables**.

(iv) Proving validity of the implication is **not trivial**.

(i) Assumptions As A Form of Plant Model

- Often the controller will (or can) operate correctly only under some **assumptions**.
- For instance, with a **level crossing**
  - we may assume an **upper bound** on the **speed of approaching trains**,
    (otherwise we'd need to close the gates arbitrarily fast)
  - we may assume that trains are **not arbitrarily slow** in the crossing,
    (otherwise we can't make promises to the road traffic)
- We shall **specify such assumptions** as a DC formula 'Asm'
  on the **input observables**
  and verify correctness of 'Ctrl' wrt. 'Req' by proving validity (from 0) of

$$\mathit{Ctrl} \wedge \mathit{Asm} \implies \mathit{Req}$$

- Shall we **care** whether 'Asm' is satisfiable?

(ii) Intermediate Design Levels

- A **top-down development approach** may involve
  - Req – **specification/requirements**
  - Des – **design**
  - Ctrl – **implementation**
- Then **correctness** is established by proving validity of

$$\mathsf{Ctrl} \implies \mathsf{Des} \qquad (1)$$

and

$$\mathsf{Des} \implies \mathsf{Req} \qquad (2)$$

(and then concluding ‘$\mathsf{Ctrl} \implies \mathsf{Req}$’ by transitivity).

- Any preference on the order (of (1) and (2))?

(iii): Different Observables

- Assume, 'Req' uses **more abstract observables** Obs_A and 'Ctrl' **more concrete observables** Obs_C.
- **For instance**:
  - in Obs_A: only consider **one gas valve**, **open or closed** – ($G : \{0, 1\}$)
  - in Obs_C: may consider **two valves** and **intermediate positions**, for instance, to react to different heating requests – $G_i : \{0, 1, 2, 3\}, i \in \{1, 2\}$
- To **prove correctness**,
  - we need information **how the observables are related**,
  - an **invariant** which **links** the data values of Obs_A and Obs_C.
- **If** we're given the linking invariant as a DC formula, say '$\mathsf{Link}_{C,A}$', **then** proving correctness of 'Ctrl' wrt. 'Req' amounts to proving

$$\models_0 \mathsf{Ctrl} \wedge \mathsf{Link}_{C,A} \implies \mathsf{Req}.$$

- For instance, $\mathsf{Link}_{C,A} := \lceil\rceil \vee \lceil G \iff (G_1 > 0 \vee G_2 > 0)\rceil$.

Obstacle (iv): How to Prove Correctness?

Main options:

- **by hand** on the basis of DC semantics (as demonstrated before),
- using **proof rules** from a calculus ($\rightarrow$ later),
- sometimes a **general theorem** may fit (e.g. cycle times of PLC automata),
- **algorithms** as in Uppaal ($\rightarrow$ later).

Tell Them What You've Told Them...

- **Design ideas** for the behaviour of real-time system controllers can also be described using DC formulae.
- The **correctness** of a design idea wrt. requirements can principally be proven "on foot" (using the DC semantics and analysis results).
- This approach is not limited to over-simplified (?) **gas burner** controllers:
  - Consider **plant assumptions**.
  - Use **intermediate designs** in a step-by-step development.
  - Link **different observables** by invariants.
  - Consider other **proof techniques**.

References

References

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