Contents & Goals

Last Lecture:
- DC Syntax and Semantics: Terms, Formulae

This Lecture:
- Educational Objectives: Capabilities for following tasks/questions.
  - Read (and at best also write) Duration Calculus formulae – including abbreviations.
  - What is Validity/Satisfiability/Realisability for DC formulae?
  - How can we prove a design correct?

- Content:
  - Duration Calculus Abbreviations
  - Basic Properties
  - Validity, Satisfiability, Realisability
  - A correctness proof for a gas busier design
Duration Calculus: Overview

We will introduce three (or five) syntactical “levels”:

(i) **Symbols:**

\[ f, g, \text{true, false, } =, <, \leq, \geq, \text{ x, y, z, X, Y, Z, } d \]

(ii) **State Assertions:**

\[ P ::= 0 \mid 1 \mid X = d \mid \neg P_1 \mid P_1 \land P_2 \]

(iii) **Terms:**

\[ \theta ::= x \mid \ell \mid \int P \mid f(\theta_1, \ldots, \theta_n) \]

(iv) **Formulae:**

\[ F ::= p(\theta_1, \ldots, \theta_n) \mid \neg F_1 \mid F_1 \land F_2 \mid \forall x \bullet F_1 \mid F_1 ; F_2 \]

(v) **Abbreviations:**

\[ [ ] , \ [ P ] , \ [ P ]^i , \ [ P ]^{\leq i} , \ \Diamond F , \ \Box F \]
Remark 2.10. [Rigid and chop-free] Let $F$ be a duration formula, $I$ an interpretation, $V$ a valuation, and $[b, e] \in \text{Intv}.$

- If $F$ is rigid, then
  \[
  \forall [b', e'] \in \text{Intv} : I[\llbracket F \rrbracket](V, [b, e]) = I[\llbracket F \rrbracket](V, [b', e']).
  \]

- If $F$ is chop-free or $\theta$ is rigid, then in the calculation of the semantics of $F$, every occurrence of $\theta$ denotes the same value.

**Example:**
\[
\begin{align*}
\ell = 3; \theta = 5 & \quad \theta = 3; \theta = 5 \\
\ell > 0 \land \ell > 1 & \quad \ell > 1 \land \ell > 2 \quad \text{chop-free}
\end{align*}
\]

Substitution Lemma

Lemma 2.11. [Substitution]
Consider a formula $F$, a global variable $x$, and a term $\theta$ such that $F$ is chop-free or $\theta$ is rigid.

Then for all interpretations $I$, valuations $V$, and intervals $[b, e]$,
\[
I[\llbracket F[x := \theta] \rrbracket](V, [b, e]) = I[\llbracket F \rrbracket](V[x := d], [b, e])
\]
where $d = I[\llbracket \theta \rrbracket](V, [b, e]).$
**Duration Calculus: Overview**

We will introduce three (or five) syntactical "levels":

(i) **Symbols:**

\[ f, g, \text{true}, \text{false}, =, <, >, \leq, \geq, \ x, y, z, \ X, Y, Z, \ d \]

(ii) **State Assertions:**

\[ P ::= 0 \ | \ 1 \ | \ X = d \ | \ \neg P_1 \ | \ P_1 \land P_2 \]

(iii) **Terms:**

\[ \theta ::= x \ | \ \ell \ | \ \int P \ | \ f(\theta_1, \ldots, \theta_n) \]

(iv) **Formulae:**

\[ F ::= p(\theta_1, \ldots, \theta_n) \ | \ \neg F_1 \ | \ F_1 \land F_2 \ | \ \forall x \bullet F_1 \ | \ F_1 ; F_2 \]

(v) **Abbreviations:**

\[ [\ ], \ [P], \ [P]^t, \ [P]^{\leq t}, \ \Diamond F, \ \Box F \]

---

**Duration Calculus Abbreviations**
Abbreviations

- $[\ell] := \ell = 0$ (point interval)
- $[P] := \{P = 0\} \land \ell > 0$ (almost everywhere)
- $[P]^t := [P] \land \ell = t$ (for time $t$)
- $[P]^{\leq t} := [P] \land \ell \leq t$ (up to time $t$)

- $\Diamond F := \text{true} \land F \land \text{true}$ (for some subinterval)
- $\Box F := \neg \Diamond \neg F$ (for all subintervals)

Abbreviations: Examples
Duration Calculus: Looking back

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

Back to our gas burner:
- \( G, F, I, H, \quad \mathcal{D}(G) = \cdots = \mathcal{D}(H) = \{0, 1\} \)
- Define \( L \) as \( G \land \neg F \).

Strangest operators:
- **everywhere** — Example: \([G]\)
  (Holds in a given interval \([b, e]\) iff the gas valve is open almost everywhere.)
- **chop** — Example: \([\neg I; I; \neg I]\) \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.)

DC Validity, Satisfiability, Realisability
Validation, Satisfiability, Realisability

Let $\mathcal{I}$ be an interpretation, $\mathcal{V}$ a valuation, $[b, e]$ an interval, and $F$ a DC formula.

- $\mathcal{I}, \mathcal{V}, [b, e] \models F$ ("$F$ holds in $\mathcal{I}, \mathcal{V}, [b, e]$") iff $\mathcal{I}[F](\mathcal{V}, [b, e]) = \text{tt}.$
- $F$ is called satisfiable iff it holds in some $\mathcal{I}, \mathcal{V}, [b, e]$.
- $\mathcal{I}, \mathcal{V} \models F$ ("$\mathcal{I}$ and $\mathcal{V}$ realise $F$") iff $\forall [b, e] \in \text{Intv} : \mathcal{I}, \mathcal{V}, [b, e] \models F$.
- $F$ is called realisable iff some $\mathcal{I}$ and $\mathcal{V}$ realise $F$.
- $\mathcal{I} \models F$ ("$\mathcal{I}$ realises $F$") iff $\forall \mathcal{V} \in \text{Val} : \mathcal{I}, \mathcal{V} \models F$.
- $\models F$ ("$F$ is valid") iff $\forall$ interpretation $\mathcal{I} : \mathcal{I} \models F$.

Remark 2.13. For all DC formulae $F$,

- $F$ is satisfiable iff $\neg F$ is not valid,
  $F$ is valid iff $\neg F$ is not satisfiable.

- If $F$ is valid then $F$ is realisable, but not vice versa.

- If $F$ is realisable then $F$ is satisfiable, but not vice versa.
**Examples: Valid? Realisable? Satisfiable?**

- $\mathcal{I}, \mathcal{V}, [b, e] \models F$ (**F holds** in $\mathcal{I}, \mathcal{V}, [b, e]$) iff $\mathcal{I} [F](\mathcal{V}, [b, e]) = \text{tt}$.
- $F$ is called **satisfiable** iff it holds in some $\mathcal{I}, \mathcal{V}, [b, e]$.
- $\mathcal{I}, \mathcal{V} \models F$ (**$\mathcal{I}$ and $\mathcal{V}$ realise $F$**) iff $\forall [b, e] \in \text{Intv} : \mathcal{I}, \mathcal{V}, [b, e] \models F$.
- $F$ is called **realisable** iff some $\mathcal{I}$ and $\mathcal{V}$ realise $F$.
- $\models F$ (**$\mathcal{I}$ realises** $F$) iff $\forall \mathcal{V} \in \text{Val} : \mathcal{I}, \mathcal{V} \models F$.
- $\models F$ (**$F$ is valid** ) iff $\forall$ interpretation $\mathcal{I} : \mathcal{I} \models F$.

<table>
<thead>
<tr>
<th>$\ell \geq 0$</th>
<th>Satisfiable</th>
<th>Realisable</th>
<th>Valid</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\ell = 1$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>$\ell = 30$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>$\ell = 20$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>$\ell &lt; 0$</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
</tbody>
</table>

**Initial Values**

- $\mathcal{I}, \mathcal{V} \models_0 F$ (**$\mathcal{I}$ and $\mathcal{V}$ realise $F$ from 0**) iff $\forall t \in \text{Time} : \mathcal{I}, \mathcal{V}, [0, t] \models F$.
- $F$ is called **realisable from 0** iff some $\mathcal{I}$ and $\mathcal{V}$ realise $F$ from 0.

- Intervals of the form $[0, t]$ are called **initial intervals**.

- $\mathcal{I} \models_0 F$ (**$\mathcal{I}$ realises** $F$ from 0) iff $\forall \mathcal{V} \in \text{Val} : \mathcal{I}, \mathcal{V} \models_0 F$.
- $\models_0 F$ (**$F$ is valid from 0**) iff $\forall$ interpretation $\mathcal{I} : \mathcal{I} \models_0 F$. 
Initial or not Initial...

For all interpretations $\mathcal{I}$, valuations $\mathcal{V}$, and DC formulae $F$,

(i) $\mathcal{I}, \mathcal{V} \models F$ implies $\mathcal{I}, \mathcal{V} \models^0 F$, but not vice versa,

(ii) if $F$ is realisable then $F$ is realisable from 0, but not vice versa,

(iii) $F$ is valid iff $F$ is valid from 0.

Specification and Semantics-based Correctness Proofs of
Real-Time Systems with DC
Methodology: Ideal World...

(i) Choose a collection of observables 'Obs'.
(ii) Provide the requirement/specification 'Spec' as a conjunction of DC formulae (over 'Obs').
(iii) Provide a description 'Ctrl' of the controller in form of a DC formula (over 'Obs').
(iv) We say 'Ctrl' is correct (wrt. 'Spec') iff

\[ \models_0 \text{Ctrl} \implies \text{Spec}. \]

---

Gas Burner Revisited

(i) Choose observables:
- two boolean observables \( G \) and \( F \)
  (i.e. \( \text{Obs} = \{G, F\}, \mathcal{D}(G) = \mathcal{D}(F) = \{0, 1\} \))
- \( G = 1 \): gas valve open (output)
- \( F = 1 \): have flame (input)
- define \( L := G \land \neg F \) (leakage)

(ii) Provide the requirement:

\[ \text{Req} : \iff \Box (\ell \geq 60) \implies \forall \ell \left( L \leq \ell \right) \]
Gas Burner Revisited

(iii) Provide a description ‘Ctrl’ of the controller in form of a DC formula (over ‘Obs’).

Here, firstly consider a design:

- Des-1 : \( \iff \Box([L] \implies \ell \leq 1) \)
- Des-2 : \( \iff \Box([L]; [-L]; [L] \implies \ell > 30) \)

(iv) Prove correctness:
- We want (or do we want \( \models_0 \) ...?):
  \( \models (\text{Des}-1 \land \text{Des}-2 \implies \text{Req}) \)

     \[
     \text{Thm. 2.16}
     \]

- We do show
  \( \models \text{Req}-1 \implies \text{Req} \)

     \[
     \text{Lem. 2.17}
     \]

  with the simplified requirement
  \( \text{Req}-1 := \Box(\ell \leq 30 \implies \ell \leq 1), \)

- and we show
  \( \models (\text{Des}-1 \land \text{Des}-2) \implies \text{Req}-1. \)

     \[
     \text{Lem. 2.19}
     \]
References