Contents & Goals

- Last Lecture:
  • DC (un)decidability

- This Lecture:
  • Educational Objectives:
    • what's notable about TA syntax? What's simple clock constraint?
    • what's a configuration of a TA? When are two in transition relation?
    • what's the difference between guard and invariant? Why have both?
    • what's a computation path? A run? Zeno behaviour?

- Content:
  • Timed automata syntax
  • TA operational semantics

Introduction

- First-order Logic
- Duration Calculus (DC)
  - Semantical Correctness
  - Proofs with DC
  - DC Decidability
  - Implementables
  - PLC-Automata
  - Timed Automata (TA), Uppaal
  - Networks of Timed Automata
  - Region/Zone-Abstraction
  - Extended Timed Automata
  - Undecidability Results

Example: Off/Light/Bright

\[
\begin{align*}
\text{off} & \quad \text{light} \\
\text{press} & \quad ? \\
\text{press} & \quad ? \\
\text{press} & \quad ? \\
\text{press} & \quad ? \\
\text{press} & \quad ? \\
\end{align*}
\]

Example
Channel Names and Actions

To define timed automata formally, we need the following sets of symbols:

• A set \((a, b \in \text{Chan})\) of channel names or channels.

• For each channel \(a \in \text{Chan}\), two visible actions:
  - \(a?\) denotes input on the channel \(a\).
  - \(a!\) denotes output on the channel \(a\).

• \(\tau\) represents an internal action, not visible from outside.

• \((\alpha, \beta \in \text{Act}) := \{a? | a \in \text{Chan}\} \cup \{a! | a \in \text{Chan}\} \cup \{\tau\}\) is the set of actions.

• An alphabet \(B\) is a set of channels, i.e. \(B \subseteq \text{Chan}\).

• For each alphabet \(B\), we define the corresponding action set \(B?! := \{a? | a \in B\} \cup \{a! | a \in B\} \cup \{\tau\}\).

• Note: \(\text{Chan}?! = \text{Act}\).
Simple Clock Constraints

Let \((x, y) \in X\) be a set of clock variables (or clocks).

The set \(\{\phi \in \Phi(X)\}\) of (simple) clock constraints (over \(X\)) is defined by the following grammar:

\[\phi ::= x \sim c \mid x - y \sim c \mid \phi_1 \land \phi_2\]

where

- \(x, y \in X\),
- \(c \in Q^{+}_0\), and
- \(\sim \in \{<, >, \leq, \geq\}\).

Clock constraints of the form \(x - y \sim c\) are called difference constraints.

Example

- \(x := 0\)
- \(x \leq 3\)
- \(x > 3\)
- \(x \leq 3\)
- \(x > 3\)
- \(x \leq 3\)
- \(x > 3\)

Timed Automaton

Definition 4.3. [Timed automaton]

A (pure) timed automaton \(A\) is a structure \(A = (L, B, X, I, E, \ell_{ini})\) where

- \(\ell \in L\) is a finite set of locations (or control states),
- \(B \subseteq \text{Chan},\)
- \(X\) is a finite set of clocks,
- \(I : L \to \Phi(X)\) assigns to each location a clock constraint, its invariant,
- \(E \subseteq L \times B \times \Phi(X)\times 2^X \times L\) a finite set of directed edges.
- Edges \((\ell, \alpha, \phi, Y, \ell')\) from location \(\ell\) to \(\ell'\) are labelled with an action \(\alpha\), a guard \(\phi\), and a set \(Y\) of clocks that will be reset.
- \(\ell_{ini}\) is the initial location.

Graphical Representation of Timed Automata

Clock V aluations

Let \(X\) be a set of clocks. A valuation \(\nu\) of clocks in \(X\) is a mapping \(\nu : X \to \text{Time}\) assigning each clock \(x \in X\) the current time \(\nu(x)\).

Let \(\phi\) be a clock constraint. The satisfaction relation between clock valuations \(\nu\) and clock constraints \(\phi\), denoted by \(\nu \models \phi\), is defined inductively:

- \(\nu \models x \sim c\) iff \(\nu(x) \sim c\)
- \(\nu \models x - y \sim c\) iff \(\nu(x) - \nu(y) \sim c\)
- \(\nu \models \phi_1 \land \phi_2\) iff \(\nu \models \phi_1\) and \(\nu \models \phi_2\)
The clock valuation $\nu$ is a mapping $\nu : X \rightarrow \mathbb{R}$ defined inductively:

- $\nu(x^0) = 0$ for all $x^0 \in X$.
- If $x = \neg y$ for some $y$, then $\nu(x) = \nu(y)$.
- If $x = y(z)$ for some $y$ and $z$, then $\nu(x) = \nu(y) + \nu(z)$.

The transition relation $\xrightarrow{\ell}$ is a relation between clock valuations $\nu$ and time valuations $\lambda$, such that there exists a valuation $\nu'$ of the form $(x, \nu')$ such that $\nu' \equiv \nu$ and $\lambda' \equiv \lambda$, where $\nu'$ and $\lambda'$ are the clock and time valuations respectively.

A transition sequence of the form $\tau \xrightarrow{\ell_1} \tau' \xrightarrow{\ell_2} \tau''$ is called an $A$-transition if $\forall x \in X$ there are $\nu, \nu'$ such that $\nu(x) = 0$ and $\nu'(x) = \nu(x) + \ell_1(x)$.

Operational Semantics of TA

Example

Operational Semantics of TA

Operational Semantics of TA

Clock Valuations

Temporal Logic and Clock Constraints

Operational Semantics of TA

Operational Semantics of TA
Two Approaches to Exclude "Bad" Configurations

**Approach 1:**
- Rule out configurations not in the step from
- Reclass Definition 4.4
- Recall the user model for our light controller:
- Discussion: Set of Configurations

**Approach 2:**
- Compute paths that are reachable
- Bad configurations are not even configurations!
Definition 4.9. An infinite sequence $t_0, t_1, t_2, \ldots$ of values $t_i \in \text{Time}$ for $i \in \mathbb{N}_0$ is called a real-time sequence if and only if it has the following properties:

- **Monotonicity:** $\forall i \in \mathbb{N}_0: t_i \leq t_{i+1}$
- **Non-Zeno behaviour** (or unboundedness or progress):
  $\forall t \in \text{Time} \exists i \in \mathbb{N}_0: t < t_i$

Definition 4.10. A run of $A$ starting in the time-stamped configuration $\langle \ell_0, \nu_0 \rangle$, $t_0$ is an infinite computation path of $A$:

$\xi = \langle \ell_0, \nu_0 \rangle, t_0 \lambda_1 \rightarrow \langle \ell_1, \nu_1 \rangle, t_1 \lambda_2 \rightarrow \langle \ell_2, \nu_2 \rangle, t_2 \lambda_3 \rightarrow \ldots$

where $(t_i)_{i \in \mathbb{N}_0}$ is a real-time sequence.

If $\langle \ell_0, \nu_0 \rangle \in \mathcal{C}_{\text{ini}}$ and $t_0 = 0$, then we call $\xi$ a run of $A$.

Example:

$$\ell \leq 2 \quad s, \quad x < 10, \quad x := 0$$

$$a! \quad x \geq 10$$

References:
