Contents & Goals

Last Lecture:
• PLC, PLC automata

This Lecture:
• Educational Objectives:
  what's notable about TAs 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 behavior?

Content:
• Timed automata syntax
• TA operational semantics

Recall: Tying It All Together

abstraction
formal description
language I
semantic
integration
automatic
verification
formal descr.
language II
requirements
duration calculus
constraint diagrams
dc timed automata
live seq. charts
PLC-Automata
PLC-code
code
PLC-code
C-code
code
logical semantics
logical semantics
compiler
equiv.
equiv.
equiv.
operationalsemantics
operationalsemantics

Example:

off light bright

User:

ℓ₀
ℓ₁
ℓ₂
ℓ₃
ℓ₄

y := 0

y < 2

y := 0

y > 3

press
press
press
press
press
press
press

Example

User:

ℓ₀
ℓ₁
ℓ₂
ℓ₃
ℓ₄

y := 0

y < 2

y := 0

y > 3

press
press
press
press
press
press
press

Example

off light bright
Example

Channel Names and Actions

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

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

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

• \( \tau / \in \text{Chan} \) 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.

• A alphabet \( B \subseteq \text{Chan} \) 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

• \( x, y \in X \),

• \( c \in \mathbb{Q}^{+0} \), and

• \( \sim \in \{<, >, \leq, \geq\} \).

• Clock constraints of the form \( x - y \sim c \) are called difference constraints.
In that case we write $\phi = \nu$ if and only if $\phi = \nu \sim c$. If $\phi$ is a clock constraint, then the set of clock assignments $\text{ClockValuations}(\phi)$ is defined inductively: if $\phi$ is an action, denoted by $\phi \sim \ell$, then $\text{ClockValuations}(\phi) = \{(X, I) \mid X \sim \ell, I \sim \nu \sim c\}$, where $X$ is a clock variable and $I$ is a clock valuation. If $\phi$ is a reset, then $\text{ClockValuations}(\phi) = \{(X, I) \mid X \sim \ell, I \sim \nu \sim c\}$, where $X$ is a clock variable and $I$ is a clock valuation. If $\phi$ is a guard, then $\text{ClockValuations}(\phi) = \{(X, I) \mid X \sim \ell, I \sim \nu \sim c\}$, where $X$ is a clock variable and $I$ is a clock valuation. If $\phi$ is a control state, then $\text{ClockValuations}(\phi) = \{(X, I) \mid X \sim \ell, I \sim \nu \sim c\}$, where $X$ is a clock variable and $I$ is a clock valuation. If $\phi$ is a location, then $\text{ClockValuations}(\phi) = \{(X, I) \mid X \sim \ell, I \sim \nu \sim c\}$, where $X$ is a clock variable and $I$ is a clock valuation. If $\phi$ is a timed automaton, then $\text{ClockValuations}(\phi) = \{(X, I) \mid X \sim \ell, I \sim \nu \sim c\}$, where $X$ is a clock variable and $I$ is a clock valuation. If $\phi$ is a timed graph, then $\text{ClockValuations}(\phi) = \{(X, I) \mid X \sim \ell, I \sim \nu \sim c\}$, where $X$ is a clock variable and $I$ is a clock valuation. If $\phi$ is a timed automaton, then $\text{ClockValuations}(\phi) = \{(X, I) \mid X \sim \ell, I \sim \nu \sim c\}$, where $X$ is a clock variable and $I$ is a clock valuation. If $\phi$ is a timed graph, then $\text{ClockValuations}(\phi) = \{(X, I) \mid X \sim \ell, I \sim \nu \sim c\}$, where $X$ is a clock variable and $I$ is a clock valuation.
Recall the user model for our light controller:

Discussion: Set of Configurations

A configuration is a tuple $\langle \lambda, \alpha, \phi, Y, \ell \rangle$, where $\lambda$, $\alpha$, $\phi$, and $Y$ are the light state, action, time, and location, respectively, and $\ell$ is the clock valuation.

An action occurs, location may change, some clocks may be reset, or time elapses.

Operational Semantics of TA

Example

Operations on Clock Valuations

Definition Semantics of TA

Operational Semantics of TA

Example

Operations on Clock Valuations

Definition Semantics of TA
Example

- A sequence of time-stamped configurations

\[ \pi = (\ell, \nu) \]

- Example behaviour

- Non-Zenobehaviour

- Zeno

\[ \exists \forall \alpha \]

- Time for run

\[ t \]

- Definition 4.9.
Example

\[ \ell_0 \leq \ell_1 \]

\( s, x < 10 \), \( x = 0 \)

\( a! \geq 10 \)

References
