Real-Time Systems

Lecture 18: Automatic Verification of DC Properties for TA II

2013-07-10

Dr. Bernd Westphal

Albert-Ludwigs-Universität Freiburg, Germany
Contents & Goals

Last Lecture:
- Completed Undecidability Results for TBA
- Started to relate TA and DC

This Lecture:
- **Educational Objectives:** Capabilities for following tasks/questions.
  - How can we relate TA and DC formulae? What’s a bit tricky about that?
  - Can we use Uppaal to check whether a TA satisfies a DC formula?

- **Content:**
  - An evolution-of-observables semantics of TA
  - A satisfaction relation between TA and DC
  - Model-checking DC properties with Uppaal
Observing Timed Automata
**DC Properties of Timed Automata**

**Wanted:** A satisfaction relation between networks of timed automata and DC formulae, a notion of $\mathcal{N}$ satisfies $F$, denoted by $\mathcal{N} \models F$.

**Plan:**

- Consider network $\mathcal{N}$ consisting of TA
  
  $$\mathcal{A}_{e,i} = (L_i, C_i, B_i, U_i, X_i, V_i, I_i, E_i, \ell_{ini,i})$$

- Define observables $\text{Obs}(\mathcal{N})$ of $\mathcal{N}$.

- Define evolution $\mathcal{I}_\xi$ of $\text{Obs}(\mathcal{N})$ induced by computation path $\xi \in \text{CompPaths}(\mathcal{N})$ of $\mathcal{N}$,
  
  $$\text{CompPaths}(\mathcal{N}) = \{\xi \mid \xi \text{ is a computation path of } \mathcal{N}\}$$

- Say $\mathcal{N} \models F$ if and only if $\forall \xi \in \text{CompPaths}(\mathcal{N}) : \mathcal{I}_\xi \models_0 F$. 
Let $\mathcal{N}$ be a network of $n$ extended timed automata

$$A_{e,i} = (L_i, C_i, B_i, U_i, X_i, V_i, I_i, E_i, \ell_{ini,i})$$

For simplicity: assume that the $L_i$ and $X_i$ are pairwise disjoint and that each $V_i$ is pairwise disjoint to every $L_i$ and $X_i$ (otherwise rename).

- **Definition**: The observables $\text{Obs}(\mathcal{N})$ of $\mathcal{N}$ are

$$\{\ell_1, \ldots, \ell_n\} \cup \bigcup_{1 \leq i \leq n} V_i$$

with
- $D(\ell_i) = L_i$,
- $D(v)$ as given, $v \in V_i$. 

- current location of $A_{e,i}$
Observables of TA Network: Example

\[ A_{e,i} = (L_i, C_i, B_i, U_i, X_i, V_i, I_i, E_i, \ell_{ini,i}). \]

The observables \( \text{Obs}(\mathcal{N}) \) of \( \mathcal{N} \) are \( \{\ell_1, \ldots, \ell_n\} \cup \bigcup_{1 \leq i \leq n} V_i \) with

- \( \mathcal{D}(\ell_i) = L_i, \)
- \( \mathcal{D}(v) \) as given, \( v \in V_i. \)

\[
\begin{align*}
\text{Obs}(\mathcal{N}) &= \{\ell_1, \ell_2\} \cup \{a\} \\
\mathcal{D}(\ell_1) &= \{\text{off}, \text{light}, \text{bright}\} \\
\mathcal{D}(\ell_2) &= \{\ell_0\} \\
\mathcal{D}(a) &= \{0, \ldots, 5\}
\end{align*}
\]
Evolutions of TA Network

Recall: computation path

\[ \xi = \langle \mathbf{l}_0, \nu_0 \rangle, t_0 \xrightarrow{\lambda_1} \langle \mathbf{l}_1, \nu_1 \rangle, t_1 \xrightarrow{\lambda_2} \langle \mathbf{l}_2, \nu_2 \rangle, t_2 \xrightarrow{\lambda_3} \ldots \]

of \( \mathcal{N} \), \( \mathbf{l}_j \) denotes a tuple \( \langle \ell_1^j, \ldots, \ell_n^j \rangle \in L_1 \times \cdots \times L_n \).

Recall: Given \( \xi \) and \( t \in \text{Time} \), we use \( \xi(t) \) to denote the set

\[ \{ \langle \mathbf{l}, \nu \rangle \mid \exists i \in \mathbb{N}_0 : t_i \leq t \leq t_{i+1} \land \mathbf{l} = \mathbf{l}_i \land \nu = \nu_i + t - t_i \} \]

of configurations at time \( t \).

New: \( \bar{\xi}(t) \) denotes \( \langle \mathbf{l}_j, \nu_j + t - t_j \rangle \) where \( j = \max\{i \in \mathbb{N}_0 \mid t_i \leq t \land \mathbf{l} = \mathbf{l}_i \} \).

Our choice:

- **Ignore** configurations assumed for 0-time only.
- **Extend** finite computation paths to infinite length, staying in last configuration.

Yet clocks advance – see later. \((\text{Assume no time clock.})\)
Evolutions of TA Network: Example

\( \tilde{\xi}(t) \) denotes \( \langle \ell_j, \nu_j + t - t_j \rangle \) where \( j = \max \{ i \in \mathbb{N}_0 \mid t_i \leq t \} \).

Example:

\[
\xi = \langle \text{off}, 0 \rangle \xrightarrow{2.5}{t_0} \langle \text{off}, 2.5 \rangle \xrightarrow{2.5}{t_0} \langle \text{light}, 0 \rangle \xrightarrow{2.5}{t_2} \langle \text{bright}, 0 \rangle \xrightarrow{2.5}{t_3} \langle \text{off}, 0 \rangle \xrightarrow{2.5}{t_4} \langle \text{off}, 1 \rangle \xrightarrow{1.0}{t_5} \langle \text{off}, 3.5 \rangle \xrightarrow{\tau}{t_5} \cdots
\]

- \( \tilde{\xi}(0) = \langle \text{off}, x=0 \rangle \)
- \( \tilde{\xi}(1.0) = \langle \text{off}, x=0+(1.0-0) \rangle \)
- \( \tilde{\xi}(2.5) = \langle \text{off}, x=2.5 \rangle \)

\( \{ i \mid t_i \leq 2.5 \} = \{ 4, 3, 2, 1 \} \).
\( \tilde{\xi} \) induces the unique interpretation

\[
\mathcal{I}_\xi : \text{Obs}(\mathcal{N}) \rightarrow (\text{Time} \rightarrow \mathcal{D})
\]

of \( \text{Obs}(\mathcal{N}) \) defined pointwise as follows:

\[
\mathcal{I}_\xi(a)(t) = \begin{cases} 
\ell^i, & \text{if } a = \ell_i, \tilde{\xi}(t) = \langle \ell^1, \ell^i, \ell^n \rangle, \nu \\
\nu(a), & \text{if } a \in V_i, \tilde{\xi}(t) = \langle \ell, \nu \rangle
\end{cases}
\]

**Example:** \( \mathcal{D}(\ell_1) = \{ \text{off}, \text{light}, \text{bright} \} \)

\( \xi = \langle \text{off} \rangle, 0 \xrightarrow{2.5} \langle \text{off} \rangle, 2.5 \xrightarrow{\tau} \langle \text{light} \rangle, 2.5 \xrightarrow{\tau} \langle \text{bright} \rangle, 2.5 \xrightarrow{\tau} \langle \text{off} \rangle, 2.5 \xrightarrow{1.0} \langle \text{off} \rangle, 3.5 \xrightarrow{\tau} \ldots \)

\[
\mathcal{I}_\xi
\]

```
off
light
bright
```

```
0 1 2 3 4 5 6 7
```

```
Time
```
Evolutions of TA Network Cont’d

\[ \xi = \langle \text{off}_0 \rangle, 0 \xrightarrow{2.5} \langle \text{off}_{2.5} \rangle, 2.5 \xrightarrow{\tau} \langle \text{light}_0 \rangle, 2.5 \xrightarrow{\tau} \langle \text{bright}_0 \rangle, 2.5 \xrightarrow{\tau} \langle \text{off}_0 \rangle, 2.5 \xrightarrow{1.0} \langle \text{off}_1 \rangle, 3.5 \xrightarrow{\tau} \ldots \]

Abbreviations as usual:

- \( I_\xi(\ell_1)(0) = \text{off} \)
- \( I(\ell_1 = \text{off})(0) = \text{Iff}(I_\xi(\ell_1)(0) = I(\text{off})) = \text{off} \)
- \( I(\text{off})(1.0) = I(\ell_1 = \text{off})(1.0) \)

if \( L_i \) pairwise disjoint.
But **what about clocks**? Why not $x \in \text{Obs}(\mathcal{N})$ for $x \in X_i$?

We would know how to define $I_\xi(x)(t)$, namely

$$I_\xi(x)(t) = \nu_\xi(x) + (t - t_\xi(t)).$$

But... $I_\xi(x)(t)$ changes too often.

**Better** (if wanted):

- add $\Phi(X_1 \cup \cdots \cup X_i)$ to $\text{Obs}(\mathcal{N})$,
  with $\mathcal{D}(\varphi) = \{0, 1\}$ for $\varphi \in \Phi(X_1 \cup \cdots \cup X_i)$.

- set

$$I_\xi(\varphi)(t) = \begin{cases} 1, & \text{if } \nu(x) \models \varphi, \bar{\xi}(t) = \langle \vec{l}, \nu \rangle \\ 0, & \text{otherwise} \end{cases}$$

The truth value of constraint $\varphi$ can endure over non-point intervals.
Some Checkable Properties
First Answer: $\mathcal{N} \models F$ if and only if $\forall \xi \in \text{CompPaths}(\mathcal{N}) : \mathcal{I}_\xi \models_0 F$.

Second Question: what kinds of DC formulae can we check with Uppaal?

- Clear: Not every DC formula. (Otherwise contradicting undecidability results.)

- Quite clear: $F = \square \lceil \text{off} \rceil$ or $F = \neg \lozenge \lceil \text{light} \rceil$
  
  (Use Uppaal’s fragment of TCTL, something like $\forall \square \text{off}$, but not exactly (see later).)

- Maybe: $F = \ell > 5 \implies \lozenge \lceil \text{off} \rceil^5$

- Not so clear: $F = \neg \lozenge (\lceil \text{bright} \rceil ; \lceil \text{light} \rceil)$
Model-Checking DC Properties with Uppaal

- **Second Question**: what kinds of DC formulae can we check with Uppaal?

**Wanted**:
- a function $f$ mapping DC formulae to Uppaal DC formulae and
- a transformation $\widetilde{\cdot}$ of networks of TA

such that

$$\widetilde{\mathcal{N}} \models_{\text{Uppaal}} f(F) \iff \mathcal{N} \models F \left( \iff \forall \mathcal{E} \in \text{Comp}(\mathcal{N}) \exists x \mathcal{E} x F \right)$$

One step more general: an additional **observer** construction $\mathcal{O}(\cdot)$ such that

$$\mathcal{N} \parallel \mathcal{O}(F) \models_{\text{Uppaal}} f_{\mathcal{O}}(F) \iff \mathcal{N} \models F$$

may use components of the observer.
**Model-Checking Invariants with Uppaal**

- **Quite clear:** $F = \square [P]$.
  - Unfortunately, we have in general not
    \[ N \models \square [P] \implies N \models \forall \square P, \]
    but in general not
    \[ N \models \forall \square P \implies N \models \square [P] \]
    because Uppaal also considers $P$ without duration.
  - Possible fix: measure duration explicitly, transform
    \[
    \begin{align*}
    \ell &\quad \ell \\
    \varphi &\quad \varphi
    \end{align*}
    \]
    Then check for $N \models \forall \square (P \land z > 0). \quad \text{if } P \in \ell$. 

Testable DC Properties
A More Systematic Approach

We have seen $f_{\mathcal{O}}$, $\sim$, and $\mathcal{O}(\cdot)$ with

$$\tilde{\mathcal{N}} \parallel \mathcal{O}(F) \models_{\text{Uppaal}} f_{\mathcal{O}}(F) \iff \mathcal{N} \models F \quad (\star)$$

for some particular $F$. **Tedious**: always have to prove ($\star$).

**Better**:
- characterise a subset of DC,
- give procedures to construct $f_{\mathcal{O}}(\cdot)$, $\sim$, and $\mathcal{O}(\cdot)$
- prove once and for all that, if $F$ is in this fragment, then

$$\tilde{\mathcal{N}} \parallel \mathcal{O}(F) \models_{\text{Uppaal}} f_{\mathcal{O}}(F) \iff \mathcal{N} \models F$$

**Even better**: exact (syntactic) characterisation of the DC fragment that is testable (not in the lecture).
**Testability**

**Definition 6.1.** A DC formula $F$ is called **testable** if an observer (or test automaton (or monitor)) $A_F$ exists such that for all networks $N = C(A_1, \ldots, A_n)$ it holds that

$$N \models F \iff C(A'_1, \ldots, A'_n, A_F) \models \forall \square \neg (A_F.q_{bad})$$

Otherwise it’s called **untestable**.

**Proposition 6.3.** There exist untestable DC formulae.

**Theorem 6.4.** DC implementables are testable.
“Whenever we observe a change from $A$ to $\neg A$ at time $t_A$, the system has to produce a change from $B$ to $\neg B$ at some time $t_B \in [t_A, t_A + 1]$ and a change from $C$ to $\neg C$ at time $t_B + 1$.

**Sketch of Proof:** Assume there is $A_F$ such that, for all networks $\mathcal{N}$, we have

$$\mathcal{N} \models F \iff C(A'_1, \ldots, A'_n, A_F) \models \forall \Box \neg (A_F.q_{bad})$$

Assume the number of clocks in $A_F$ is $n \in \mathbb{N}_0$. 
Consider the following time points:

- \( t_A := 1 \)
- \( t_i^B := t_A + \frac{2i - 1}{2(n+1)} \) for \( i = 1, \ldots, n + 1 \)
- \( t_i^C \in \left[ t_i^B + 1 - \frac{1}{4(n+1)}, t_i^B + 1 + \frac{1}{4(n+1)} \right] \) for \( i = 1, \ldots, n + 1 \)
  with \( t_i^C - t_i^B \neq 1 \) for \( 1 \leq i \leq n + 1 \).

**Example:** \( n = 3 \)
Example: \( n = 3 \)

- The shown interpretation \( \mathcal{I} \) satisfies assumption of property.
- It has \( n + 1 \) candidates to satisfy commitment.
- By choice of \( t^i_C \), the commitment is not satisfied; so \( F \) not satisfied.
- Because \( A_F \) is a test automaton for \( F \), is has a computation path to \( q_{bad} \).
- Because \( n = 3 \), \( A_F \) can not save all \( n + 1 \) time points \( t^i_B \).
- Thus there is \( 1 \leq i_0 \leq n \) such that all clocks of \( A_F \) have a valuation which is not in \( 2 - t^{i_0}_B + \left( -\frac{1}{4(n+1)}, \frac{1}{4(n+1)} \right) \).
Example: $n = 3$

- Because $A_F$ is a test automaton for $F$, it has a computation path to $q_{bad}$.
- Thus there is $1 \leq i_0 \leq n$ such that all clocks of $A_F$ have a valuation which is not in $2 - t_B^{i_0} + \left( -\frac{1}{4(n+1)}, \frac{1}{4(n+1)} \right)$

- Modify the computation to $\mathcal{T}'$ such that $t_C^{i_0} := t_B^{i_0} + 1$.
- Then $\mathcal{T}' \models F$, but $A_F$ reaches $q_{bad}$ via the same path.
- That is: $A_F$ claims $\mathcal{T}' \not\models F$.
- Thus $A_F$ is not a test automaton. **Contradiction.**
Theorem 6.4. DC implementables are testable.

- **Initialisation**: \([\square \lor [\pi] ; \text{true}]\)
- **Sequencing**: \([\pi] \rightarrow [\pi \lor \pi_1 \lor \cdots \lor \pi_n]\)
- **Progress**: \([\pi] \xrightarrow{\theta} [\neg \pi]\)
- **Synchronisation**: \([\pi \land \varphi] \xrightarrow{\theta} [\neg \pi]\)
- **Bounded Stability**: \([-\pi] ; [\pi \land \varphi] \xrightarrow{\leq \theta} [\pi \lor \pi_1 \lor \cdots \lor \pi_n]\)
- **Unbounded Stability**: \([-\pi] ; [\pi \land \varphi] \rightarrow [\pi \lor \pi_1 \lor \cdots \lor \pi_n]\)
- **Bounded initial stability**: \([\pi \land \varphi] \xrightarrow{\leq \theta}_0 [\pi \lor \pi_1 \lor \cdots \lor \pi_n]\)
- **Unbounded initial stability**: \([\pi \land \varphi] \rightarrow_0 [\pi \lor \pi_1 \lor \cdots \lor \pi_n]\)

**Proof Sketch**:
- For each implementable \(F\), construct \(A_F\).
- Prove that \(A_F\) is a test automaton.
Proof of Theorem 6.4: Preliminaries

- **Note**: DC does not refer to communication between TA in the network, but only to data variables and locations.

**Example**:  
\[ \ddagger = \Diamond ([v = 0] ; [v = 1]) \]

- **Recall**: transitions of TA are only triggered by synchronisation, not by changes of data-variables.
Proof of Theorem 6.4: Preliminaries

- **Note**: DC does not refer to communication between TA in the network, but only to data variables and locations.

  **Example**:

  \[ \Diamond ([v = 0] ; [v = 1]) \]

  

- **Recall**: transitions of TA are only triggered by synchronisation, not by changes of data-variables.

- **Approach**: have auxiliary *step* action.

  Technically, replace each

  by

  Note: the observer sees the data variables *after* the update.
Proof of Theorem 6.4: Sketch

- Example: $[\pi] \xrightarrow{\theta} [\neg \pi]$
Definition 6.5.

- A **counterexample formula** (CE for short) is a DC formula of the form:

\[
true ; ([\pi_1] \land \ell \in I_1) ; \ldots ; ([\pi_k] \land \ell \in I_k) ; true
\]

where for \(1 \leq i \leq k\),

- \(\pi_i\) are state assertions,
- \(I_i\) are non-empty, and open, half-open, or closed time intervals of the form
  - \((b, e)\) or \([b, e)\) with \(b \in Q_0^+\) and \(e \in Q_0^+ \cup \{\infty\}\),
  - \((b, e]\) or \([b, e]\) with \(b, e \in Q_0^+\).

\((b, \infty)\) and \([b, \infty)\) denote unbounded sets.

- Let \(F\) be a DC formula. A DC formula \(F_{CE}\) is called **counterexample formula for** \(F\) if \(\models F \iff \neg (F_{CE})\) holds.

**Theorem 6.7.** CE formulae are testable.
References