Contents & Goals

Last Lecture:
- Decidability of the location reachability problem: zones.
- Extended TA: data variables.

This Lecture:
- Educational Objectives: Capabilities for following tasks/questions.
  - By what are TA extended? Why is that useful?
  - What’s an urgent/committed location? What’s the difference?
  - What’s an urgent channel?
  - Where has the notion of “input action” and “output action” correspondences in the formal semantics?

- Content:
  - Extended TA:
    - Structuring Facilities
    - Restriction of Non-Determinism
  - The Logic of Uppaal
Recall: Example (Partly Already Seen in Uppaal Demo)

Templates:

- $L$: $l$: off $x := 0$ press? $x > 3$
  - light $x \leq 3$ press? $x > 3$
  - bright

- $U$: $u$: $v := 0$ press!
  - $v = 1$
  - $y := 0$ press!
  - $y < 2$
  - $v := 1$

System:

$\langle x \text{ press}? \text{ chan press} \rangle$ $\langle y \text{ press!} \rangle$

Extensions:

- Data Variables (Expressions, Constraints, Updates)
- Structuring
- Urgent/Committed Location, Urgent Channel
Recall: Data Variables and Expressions

• Let \((v, w \in V)\) be a set of (integer) variables.
  \((\psi_{\text{int}} \in \Psi(V))\): integer expressions over \(V\) using func. symb. \(+, -, \ldots\)
  \((\varphi_{\text{int}} \in \Phi(V))\): integer (or data) constraints over \(V\)
  using integer expressions, predicate symbols \(=, <, \leq, \ldots\), and
  boolean logical connectives. (incl. \(\land, \lor, \lnot\), \(\vdash V, V, \ldots\))

• Let \((x, y \in X)\) be a set of clocks.
  \((\varphi \in \Phi(X, V))\): (extended) guards, defined by

\[
\varphi := \varphi_{\text{clk}} \mid \varphi_{\text{int}} \mid \varphi_1 \land \varphi_2
\]

where \(\varphi_{\text{clk}} \in \Phi(X)\) is a simple clock constraint (as defined before)
and \(\varphi_{\text{int}} \in \Phi(V)\) an integer (or data) constraint.

Examples: Extended guard or not extended guard? Why?

(a) \(x < y \land v > 2\), (b) \(x < y \lor v > 2\), (c) \(v < 1 \lor v > 2\), (d) \(x < v\)

Modification or Reset Operation

• New: a modification or reset operation is

\[
x := 0, \quad x \in X,
\]

or

\[
v := \psi_{\text{int}}, \quad v \in V, \quad \psi_{\text{int}} \in \Psi(V).
\]

• By \(R(X, V)\) we denote the set of all resets.

• By \(\vec{r}\) we denote a finite list \(\langle r_1, \ldots, r_n \rangle\), \(n \in \mathbb{N}_0\),
of reset operations \(r_i \in R(X, V)\);
  \(\langle \rangle\) is the empty list.

• By \(R(X, V)^*\) we denote the set of all such lists of reset operations.
**Modification or Reset Operation**

- **New**: a modification or reset (operation) is
  \[ x := 0, \quad x \in X, \]
  or
  \[ v := \psi_{\text{int}}, \quad v \in V, \quad \psi_{\text{int}} \in \Psi(V). \]

By \( R(X, V) \) we denote the set of all resets.

By \( \mathcal{R} \) we denote a finite list \( \langle r_1, \ldots, r_n \rangle, \ n \in \mathbb{N}_0 \), of reset operations \( r_i \in R(X, V) \);

\[ \langle \rangle \] is the empty list.

By \( R(X, V)^* \) we denote the set of all such lists of reset operations.

**Examples**: Modification or not? Why?

(a) \( x := y \),
(b) \( x := v \),
(c) \( v := x \),
(d) \( v := w \),
(e) \( v := 0 \)

\begin{align*}
\checkmark & \quad \checkmark & \quad \times & \quad \times & \quad \checkmark & \quad \checkmark
\end{align*}

---

**Structuring Facilities**

- Global declarations of clocks, data variables, channels, and constants.
- Binary and broadcast channels: chan \( c \) and broadcast chan \( b \).
- Templates of timed automata.
- Instantiation of templates (instances are called process).
- System definition: list of processes.
Restricting Non-determinism

- **Urgent locations** — enforce local immediate progress.

![U]

- **Committed locations** — enforce atomic immediate progress.

![C]

- **Urgent channels** — enforce cooperative immediate progress.

```plaintext
urgent chan press;
```

Urgent Locations: Only an Abbreviation...

Replace

![Urgent Location Diagram]

with

![Updated Location Diagram]

where \( z \) is a fresh clock:
- reset \( z \) on all in-going edges,
- add \( z = 0 \) to invariant.

**Question:** How many fresh clocks do we need in the worst case for a network of \( N \) extended timed automata?
**Extended Timed Automata**

**Definition 4.39.** An extended timed automaton is a structure

\[ \mathcal{A}_e = (L, C, B, U, X, V, E, \ell_{ini}) \]

where \( L, B, X, I, \ell_{ini} \) are as in Def. 4.3, except that location invariants in \( I \) are downward closed, and where

- \( C \subseteq L \): committed locations,
- \( U \subseteq B \): urgent channels,
- \( V \): a set of data variables,
- \( E \subseteq L \times B \times \Phi(X, V) \times R(X, V)^* \times L \): a set of directed edges such that

\[ (\ell, \alpha, \varphi, \vec{r}, \ell') \in E \land \text{chan}(\alpha) \in U \implies \varphi = \text{true}. \]

Edges \((\ell, \alpha, \varphi, \vec{r}, \ell')\) from location \( \ell \) to \( \ell' \) are labelled with an action \( \alpha \), a guard \( \varphi \), and a list \( \vec{r} \) of reset operations.

**Operational Semantics of Networks**

**Definition 4.40.** Let \( \mathcal{A}_{e,i} = (L_i, C_i, B_i, U_i, X_i, V_i, I_i, E_i, \ell_{ini,i}) \), \( 1 \leq i \leq n \), be extended timed automata with pairwise disjoint sets of clocks \( X_i \).

The operational semantics of \( \mathcal{C}(\mathcal{A}_{e,1}, \ldots, \mathcal{A}_{e,n}) \) (closed!) is the labelled transition system

\[ \mathcal{T}_e(\mathcal{C}(\mathcal{A}_{e,1}, \ldots, \mathcal{A}_{e,n})) \]

\[ = (\text{Conf}, \text{Time} \cup \{\tau\}, \lambda \xrightarrow{\lambda} \lambda \in \text{Time} \cup \{\tau\}, C_{ini}) \]

where

- \( X = \bigcup_{i=1}^{n} X_i \) and \( V = \bigcup_{i=1}^{n} V_i \),
- \( \text{Conf} = \{(\vec{\ell}, \nu) \mid \ell_i \in L_i, \nu : X \cup V \rightarrow \text{Time}, \nu \models \bigwedge_{k=1}^{n} I_k(\ell_k) \} \)
- \( C_{ini} = \{(\ell_{ini,i}, \nu_{ini,i}) \} \cap \text{Conf} \)

and the transition relation consists of transitions of the following three types.
Helpers: Extended Valuations and Timeshift

- **Now:** \( \nu : X \cup V \to \text{Time} \cup \mathcal{D}(V) \)
- Canonically extends to \( \nu : \Psi(V) \to \mathcal{D} \) (valuation of expression).
- “|=” extends canonically to expressions from \( \Phi(X, V) \).

\[
\begin{align*}
\eta &:= \nu \mid \{ (v_1, \ldots, v_n) \} \\
& \quad \forall v \in V, \ f \in \mathcal{F} \\
\mathcal{D}(\Phi) &\subseteq \mathcal{Z} \\
\nu : \Phi \to \mathcal{D}(\Phi) \\
\nu[\langle v_1, \ldots, v_n \rangle] &:= \text{inter}(\nu(v_1), \ldots, \nu(v_n))
\end{align*}
\]

- Extended timeshift \( \nu + t, t \in \text{Time} \), applies to clocks only:
  - \((\nu + t)(x) := \nu(x) + t, x \in X\)
  - \((\nu + t)(v) := \nu(v), v \in V\).

- Effect of modification \( r \in R(X, V) \) on \( \nu \), denoted by \( \nu[r] \):
  \[
  \nu[x := 0](a) := \begin{cases} 
  0, & \text{if } a = x, \\
  \nu(a), & \text{otherwise}
  \end{cases}
  \]
  \[
  \nu[v := \psi_{\text{int}}](a) := \begin{cases} 
  \nu(\psi_{\text{int}}), & \text{if } a = v, \\
  \nu(a), & \text{otherwise}
  \end{cases}
  \]

- We set \( \nu[\langle r_1, \ldots, r_n \rangle] := \nu[r_1] \ldots [r_n] = (((\nu[r_1])[r_2]) \ldots)[r_n] \).
Operational Semantics of Networks: Internal Transitions

• An internal transition \((\ell, \nu) \xrightarrow{\tau} (\ell', \nu')\) occurs if there is \(i \in \{1, \ldots, n\}\) such that
  1. there is a \(\tau\)-edge \((\ell_i, \tau, \varphi, \vec{r}, \ell'_i) \in E_i\),
  2. \(\nu |\ = \varphi\),
  3. \(\vec{\ell} = \ell_i[\ell_i := \ell'_i]\),
  4. \(\nu' = \nu[\vec{r}]\),
  5. \(\nu' |\ = I_i(\ell'_i)\),
  6. (\(\spadesuit\)) if \(\ell_k \in C_k\) for some \(k \in \{1, \ldots, n\}\) then \(\ell_i \in C_i\).

Operational Semantics of Networks: Synchronisation Transition

• A synchronisation transition \((\vec{\ell}, \nu) \xrightarrow{\tau} (\vec{\ell'}, \nu')\) occurs if there are \(i, j \in \{1, \ldots, n\}\) with \(i \neq j\) such that
  1. there are edges \((\ell_i, b!, \varphi_i, \vec{r}_i, \ell'_i) \in E_i\) and \((\ell_j, b?, \varphi_j, \vec{r}_j, \ell'_j) \in E_j\),
  2. \(\nu |\ = \varphi_i \land \varphi_j\),
  3. \(\vec{\ell} = \ell_i[\ell_i := \ell'_i][\ell_j := \ell'_j]\),
  4. \(\nu' = \nu[\vec{r}_i][\vec{r}_j]\),
  5. \(\nu' |\ = I_i(\ell'_i) \land I_j(\ell'_j)\),
  6. (\(\spadesuit\)) if \(\ell_k \in C_k\) for some \(k \in \{1, \ldots, n\}\) then \(\ell_i \in C_i\) or \(\ell_j \in C_j\).
Operational Semantics of Networks: Delay Transitions

- A delay transition \( (\vec{\ell}, \nu) \xrightarrow{t} (\vec{\ell}, \nu + t) \) occurs if
  - \( \nu + t \models \bigwedge_{k=1}^{n} I_k(\ell_k) \),
  - there are no \( i, j \in \{1, \ldots, n\} \) and \( b \in U \) with \( (\ell_i, b', \varphi_i, \vec{r}_i, \ell_i') \in E_i \) and \( (\ell_j, b', \varphi_j, \vec{r}_j, \ell_j') \in E_j \),
  - there is no \( i \in \{1, \ldots, n\} \) such that \( \ell_i \in C_i \).

Restricting Non-determinism: Example

- Property 1
  \[ \exists W \models W = 1 \]

- Property 2
  \[ \forall Q. q_1 \implies y \leq 0 \]

- Property 3
  \[ \forall Q. p_1 \wedge Q. q_1 \implies (x \geq y \implies y \leq 0) \]

| \( \mathcal{N} := P \parallel Q \parallel R \) | ✓ | ✓ | ✓ |
| \( \mathcal{N}, q_1 \) urgent | | | |
| \( \mathcal{N}, q_1 \) comm. | | | |
| \( \mathcal{N}, b \) urgent | | | |
### Restricting Non-determinism: Urgent Location

<table>
<thead>
<tr>
<th>Property 1</th>
<th>Property 2</th>
<th>Property 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\exists \diamond w = 1$</td>
<td>$\forall \square Q.q_1 \implies y \leq 0$</td>
<td>$\forall \square (P.p_1 \land Q.q_1 \implies (x \geq y \implies y \leq 0))$</td>
</tr>
<tr>
<td>$\mathcal{N}$</td>
<td>✔</td>
<td>☒</td>
</tr>
<tr>
<td>$\mathcal{N}, q_1 \text{ urgent}$</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>$\mathcal{N}, q_1 \text{ comm.}$</td>
<td>☒</td>
<td>✔</td>
</tr>
<tr>
<td>$\mathcal{N}, b \text{ urgent}$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Restricting Non-determinism: Committed Location

<table>
<thead>
<tr>
<th>Property 1</th>
<th>Property 2</th>
<th>Property 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\exists \diamond w = 1$</td>
<td>$\forall \square Q.q_1 \implies y \leq 0$</td>
<td>$\forall \square (P.p_1 \land Q.q_1 \implies (x \geq y \implies y \leq 0))$</td>
</tr>
<tr>
<td>$\mathcal{N}$</td>
<td>✔</td>
<td>☒</td>
</tr>
<tr>
<td>$\mathcal{N}, q_1 \text{ urgent}$</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>$\mathcal{N}, q_1 \text{ comm.}$</td>
<td>☒</td>
<td>✔</td>
</tr>
<tr>
<td>$\mathcal{N}, b \text{ urgent}$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Restricting Non-determinism: Urgent Channel

\[ P \]
\[ x := 0 \]
\[ p_0 \]
\[ p_1 \]
\[ p_2 \]
\[ b_w := 2 \]
\[ p_3 \]

\[ Q \]
\[ y := 0, v := 1 \]
\[ q_0 \]
\[ q_1 \]
\[ q_2 \]
\[ q_3 \]
\[ r_0 \]
\[ w := v \]

Property 1
\[ \exists \Diamond w = 1 \]

Property 2
\[ \forall \Box Q, q_1 \implies y \leq 0 \]

Property 3
\[ \forall \Box (P, p_1 \land Q, q_1 \implies (x \geq y \implies y \leq 0)) \]

<table>
<thead>
<tr>
<th></th>
<th>Property 1</th>
<th>Property 2</th>
<th>Property 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>✔</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>N, q_1 urgent</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>N, q_1 comm.</td>
<td>✗</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>N, b urgent</td>
<td>✔</td>
<td>✗</td>
<td>✔</td>
</tr>
</tbody>
</table>

Extended vs. Pure Timed Automata
**Extended vs. Pure Timed Automata**

\[ A_e = (L, C, B, U, X, V, I, E, \ell_{ini}) \]
\[ \ell, \alpha, \varphi, \vec{r}, \ell' \in L \times B_1 \times \Phi(X, V) \times R(X, V)^* \times L \]

vs.

\[ A = (L, B, X, I, E, \ell_{ini}) \]
\[ \ell, \alpha, \varphi, Y, \ell' \in E \subseteq L \times B_n \times \Phi(X) \times 2^X \times L \]

- \( A_e \) is in fact (or specialises to) a **pure** timed automaton if
  - \( C = \emptyset \),
  - \( U = \emptyset \),
  - \( V = \emptyset \),
  - for each \( \vec{r} = \langle r_1, \ldots, r_n \rangle \), every \( r_i \) is of the form \( x := 0 \) with \( x \in X \).
  - \( I(\ell), \varphi \in \Phi(X) \) is then a consequence of \( V = \emptyset \).

**Operational Semantics of Extended TA**

**Theorem 4.41.** If \( A_1, \ldots, A_n \) specialise to pure timed automata, then the operational semantics of

\[ \mathcal{C}(A_1, \ldots, A_n) \]

and

\[ \text{chan } b_1, \ldots, b_m \bullet (A_1 \parallel \ldots \parallel A_n), \]

where \( \{b_1, \ldots, b_m\} = \bigcup_{i=1}^n B_i \), **coincide**, i.e.

\[ \mathcal{T}_p(\mathcal{C}(A_1, \ldots, A_n)) = \mathcal{T}(\text{chan } b_1, \ldots, b_m \bullet (A_1 \parallel \ldots \parallel A_n)). \]
Recall

Theorem 4.33. [Location Reachability] The location reachability problem for pure timed automata is **decidable**.

Theorem 4.34. [Constraint Reachability] The constraint reachability problem for pure timed automata is **decidable**.

- And what about tea “extended” timed automata?
What About Extended Timed Automata?

Extended Timed Automata add the following features:

- **Data-Variables**
  - As long as the domains of all variables in $V$ are finite, adding data variables doesn’t hurt.
  - If they’re infinite, we’ve got a problem (encode two-counter machine).

- **Structuring Facilities**
  - Don’t hurt — they’re merely abbreviations.

- **Restricting Non-determinism**
  - Restricting non-determinism doesn’t affect the configuration space.
  - Restricting non-determinism only removes certain transitions, so makes region automaton even smaller.

The Logic of Uppaal
The Uppaal Fragment of Timed Computation Tree Logic

Consider $\mathcal{N} = \mathcal{C}(A_1, \ldots, A_n)$ over data variables $V$.

- basic formula:
  \[
  \text{atom} ::= A_i.\ell \mid \varphi
  \]
where $\ell \in L_i$ is a location and $\varphi$ a constraint over $X_i$ and $V$.

- configuration formulae:
  \[
  \text{term} ::= \text{atom} \mid \neg \text{term} \mid \text{term}_1 \land \text{term}_2
  \]

- existential path formulae:
  \[
  \text{e-formula} ::= \exists \Diamond \text{term} \mid \exists \Box \text{term}
  \]

- universal path formulae:
  \[
  \text{a-formula} ::= \forall \Diamond \text{term} \mid \forall \Box \text{term} \mid \text{term}_1 \rightarrow \text{term}_2
  \]

- formulae:
  \[
  F ::= \text{e-formula} \mid \text{a-formula}
  \]

Configurations at Time $t$

- Recall: computation path (or path) starting in $(\vec{\ell}_0, \nu_0), t_0$:
  \[
  \xi = (t_0, \nu_0), t_0 \xrightarrow{\lambda_1} (\vec{\ell}_1, \nu_1), t_1 \xrightarrow{\lambda_2} (\vec{\ell}_2, \nu_2), t_2 \xrightarrow{\lambda_3} \ldots
  \]
which is infinite or maximally finite.

- Given $\xi$ and $t \in \text{Time}$, we use $\xi(t)$ to denote the set
  \[
  \{(\vec{\ell}, \nu) \mid \exists i \in \mathbb{N}_0 : t_i \leq t \leq t_{i+1} \land \vec{\ell} = \vec{\ell}_i \land \nu = \nu_i + t - t_i\}
  \]
of configurations at time $t$.

- Why is it a set? 
- Can it be empty?
  \[
  \xi(0) = \{ (\vec{\ell}_0, \nu_0) \}
  \]
  \[
  \xi(0.14) = \{ (\vec{\ell}_0, \nu_0 + 0.14) \}
  \]
  \[
  \xi(3.0) = \{ (\vec{\ell}_0, \nu_0), (\vec{\ell}_2, \nu_2) \}
  \]

27/34
Satisfaction of Uppaal-Logic by Configurations

- We define a **satisfaction relation**

\[
\langle \vec{\ell}_0, \nu_0 \rangle, t_0 \models F
\]

between time stamped configurations

\[
\langle \vec{\ell}, \nu \rangle, t
\]

of a network \( C(A_1, \ldots, A_n) \) and **formulae** \( F \) of the Uppaal logic.

- It is defined inductively as follows:

  - \( \langle \vec{\ell}_0, \nu_0 \rangle, t_0 \models A_i \ell \) iff \( \ell_0, i = \ell \)
  
  - \( \langle \vec{\ell}_0, \nu_0 \rangle, t_0 \models \varphi \) iff \( \nu_0 \models \varphi \)
  
  - \( \langle \vec{\ell}_0, \nu_0 \rangle, t_0 \models \neg \text{term} \) iff \( \langle \vec{\ell}_0, \nu_0 \rangle, t_0 \nmodels \text{term} \)
  
  - \( \langle \vec{\ell}_0, \nu_0 \rangle, t_0 \models \text{term}_1 \land \text{term}_2 \) iff \( \langle \vec{\ell}_0, \nu_0 \rangle, t_0 \models \text{term}_i, i = 1, 2 \)

- **Exists finally**:

\[
\exists \xi \text{ path of } N \text{ starting in } \langle \vec{\ell}_0, \nu_0 \rangle, t_0 \text{ of } \xi \text{ such that } t_0 \leq t \wedge \langle \vec{\ell}, \nu \rangle \in \xi(t) \wedge \langle \vec{\ell}, \nu \rangle, t \models \text{term}
\]

**Example**: \( \exists \diamond \varphi \)
Satisfaction of Uppaal-Logic by Configurations

**Exists globally:**

- \( \langle \vec{l}_0, \nu_0 \rangle, t_0 = \exists \square \text{term} \) if and only if there exists a path \( \xi \) in the system starting in \( \langle \vec{l}_0, \nu_0 \rangle, t_0 \):

\[
\forall t \in \text{Time}, (\vec{l}, \nu) \in \text{Conf} : t_0 \leq t \wedge (\vec{l}, \nu) \in \xi(t) \implies (\vec{l}, \nu), t = \text{term}
\]

**Example:** \( \exists \square \varphi \)

```
\begin{tikzpicture}

\node[blue, circle] (l0) at (0,0) {$\langle \vec{l}_0, \nu_0 \rangle, t_0$}
    child {node[blue, circle] (l1) {$\varphi$}
        child {node[orange, circle] (l11) {$\neg \varphi$}
            child {node[orange, circle] (l111) {$\varphi$}}
        }
        child {node[orange, circle] (l12) {$\neg \varphi$}}
    }
    child {node[blue, circle] (l2) {$\neg \varphi$}
        child {node[orange, circle] (l21) {$\varphi$}
            child {node[orange, circle] (l211) {$\neg \varphi$}}
        }
        child {node[orange, circle] (l22) {$\neg \varphi$}}
    }
    child {node[blue, circle] (l3) {$\varphi$}
        child {node[orange, circle] (l31) {$\neg \varphi$}
            child {node[orange, circle] (l311) {$\varphi$}}
        }
        child {node[orange, circle] (l32) {$\neg \varphi$}}
    }

\end{tikzpicture}
```

**Always finally:**

- \( \langle \vec{l}_0, \nu_0 \rangle, t_0 = \forall \Diamond \text{term} \) if and only if there exists a path \( \xi \) starting in \( \langle \vec{l}_0, \nu_0 \rangle, t_0 \), but it is not the case that \( \exists \square \neg \text{term} \):

```
\begin{tikzpicture}

\node[blue, circle] (l0) at (0,0) {$\langle \vec{l}_0, \nu_0 \rangle, t_0$}
    child {node[blue, circle] (l1) {$\varphi$}
        child {node[orange, circle] (l11) {$\neg \varphi$}
            child {node[orange, circle] (l111) {$\varphi$}}
        }
        child {node[orange, circle] (l12) {$\neg \varphi$}}
    }
    child {node[blue, circle] (l2) {$\neg \varphi$}
        child {node[orange, circle] (l21) {$\varphi$}
            child {node[orange, circle] (l211) {$\neg \varphi$}}
        }
        child {node[orange, circle] (l22) {$\neg \varphi$}}
    }
    child {node[blue, circle] (l3) {$\varphi$}
        child {node[orange, circle] (l31) {$\neg \varphi$}
            child {node[orange, circle] (l311) {$\varphi$}}
        }
        child {node[orange, circle] (l32) {$\neg \varphi$}}
    }

\end{tikzpicture}
```

**Always globally:**

- \( \langle \vec{l}_0, \nu_0 \rangle, t_0 = \forall \square \text{term} \) if and only if \( \langle \vec{l}_0, \nu_0 \rangle, t_0 \neq \exists \Diamond \neg \text{term} \):

```
\begin{tikzpicture}

\node[blue, circle] (l0) at (0,0) {$\langle \vec{l}_0, \nu_0 \rangle, t_0$}
    child {node[blue, circle] (l1) {$\varphi$}
        child {node[orange, circle] (l11) {$\neg \varphi$}
            child {node[orange, circle] (l111) {$\varphi$}}
        }
        child {node[orange, circle] (l12) {$\neg \varphi$}}
    }
    child {node[blue, circle] (l2) {$\neg \varphi$}
        child {node[orange, circle] (l21) {$\varphi$}
            child {node[orange, circle] (l211) {$\neg \varphi$}}
        }
        child {node[orange, circle] (l22) {$\neg \varphi$}}
    }
    child {node[blue, circle] (l3) {$\varphi$}
        child {node[orange, circle] (l31) {$\neg \varphi$}
            child {node[orange, circle] (l311) {$\varphi$}}
        }
        child {node[orange, circle] (l32) {$\neg \varphi$}}
    }

\end{tikzpicture}
```
Satisfaction of Uppaal-Logic by Configurations

Leads to: $\text{"}\mathcal{N}(\ell_{\text{fin}}, \nu_{\text{fin}})\text{"}$

- $\langle \ell_0, \nu_0 \rangle, t_0 \models \text{term}_1 \rightarrow \text{term}_2$ iff $\forall$ path $\xi$ of $\mathcal{N}$ starting in $\langle \ell_0, \nu_0 \rangle, t_0$

  $\forall t \in \text{Time}, (\ell, \nu) \in \text{Conf} :$

  $t_0 \leq t \land (\ell, \nu) \in \xi(t)$

  and $(\ell, \nu), t \models \text{term}_1$

  implies $(\ell, \nu), t \models \forall \Box \text{term}_2$

Example: $\varphi_1 \rightarrow \varphi_2$


\[\begin{array}{c}
\text{Satisfaction of Uppaal-Logic by Networks}
\\
\text{We write } \\
\mathcal{N} \models e\text{-formula} \\
\text{if and only if} \\
\text{for some } \langle \ell_0, \nu_0 \rangle \in C_{\text{ini}}, \langle \ell_0, \nu_0 \rangle, 0 \models e\text{-formula}, \\
\text{and} \\
\mathcal{N} \models a\text{-formula} \\
\text{if and only if} \\
\text{for all } \langle \ell_0, \nu_0 \rangle \in C_{\text{ini}}, \langle \ell_0, \nu_0 \rangle, 0 \models a\text{-formula},
\end{array}\]

where $C_{\text{ini}}$ are the initial configurations of $T_e(\mathcal{N})$.

- If $C_{\text{ini}} = \emptyset$, (1) is a contradiction and (2) is a tautology.
- If $C_{\text{ini}} \neq \emptyset$, then

  $\mathcal{N} \models F$ if and only if $(\ell_{\text{fin}}, \nu_{\text{fin}}), 0 \models F$. 
Example

\[
\begin{align*}
\tau & \xrightarrow{\text{press?}} x := 0 \\
\tau & \xrightarrow{\text{press?}} x \leq 3 \\
\tau & \xrightarrow{\text{press?}} x > 3
\end{align*}
\]

\[
\begin{align*}
& \models \exists \diamond \mathcal{L}. \text{bright} \quad \checkmark \\
& \models \exists \square \mathcal{L}. \text{bright} \quad \checkmark \\
& \models \exists \square \mathcal{L}. \text{off} \quad \checkmark \\
& \models \forall \diamond \mathcal{L}. \text{light} \quad \times \quad x \leq 3 \\
& \models \forall \square (\mathcal{L}. \text{bright} \implies x \geq 3) \quad \times \\
& \models \mathcal{L}. \text{bright} \implies \mathcal{L}. \text{off} \quad \times
\end{align*}
\]

Example

\[
\begin{align*}
\tau & \xrightarrow{\text{press?}} x := 0 \\
\tau & \xrightarrow{\text{press?}} x \leq 3 \\
\tau & \xrightarrow{\text{press?}} x > 3
\end{align*}
\]

\[
\begin{align*}
& \models \exists \diamond \mathcal{L}. \text{bright} \quad \checkmark \\
& \models \exists \square \mathcal{L}. \text{bright} \quad \checkmark \\
& \models \exists \square \mathcal{L}. \text{off} \quad \checkmark \\
& \models \forall \diamond \mathcal{L}. \text{light} \quad \times \quad x \leq 3 \\
& \models \forall \square (\mathcal{L}. \text{bright} \implies x \geq 3) \quad \times \\
& \models \mathcal{L}. \text{bright} \implies \mathcal{L}. \text{off} \quad \times
\end{align*}
\]

Now we have a closed network.
References