Timed automata
Limitation with LTSs

- They allow to express sequences of actions, choices, loops, and concurrency
- But they cannot model time and time-dependent constraints
- Time is essential in many real-world scenarios
  - Railway systems, e.g.: a crossing barrier takes $x$ seconds to get lowered, and must be lowered $y$ seconds before the train arrives
  - Embedded controllers, e.g.: a safety system in a power plant must react within $x$ seconds
- Idea: extend LTSs by adding time
Timed LTS (TLTS)

A TLTS is a 6-ple \( \langle S, A, \Delta, T, \Theta, s_0 \rangle \)

- \( S \): States, \( A \): Actions, \( T \subseteq S \times A \times S \): Labelled transition relation, \( s_0 \): Initial state (like LTS)

- \( \Delta \): Time domain
  - Usually, \( \Delta = \mathbb{R}^{\geq 0} \) (real numbers \( \geq 0 \))

- \( \Theta \subseteq S \times \Delta \times S \): Timed transition relation
  - \( s \longrightarrow t \rightarrow s' \): From state \( s \), the system can reach state \( s' \) by waiting for a time \( t \)
TLTS: Constraints on \( \Box \)

We need to introduce these constraints so that the TLTS “makes sense” (i.e., it respects our intuitions about time)

- **Time determinism**
  - If \( s \rightarrow t \rightarrow s' \) and \( s \rightarrow t \rightarrow s'' \) then \( s' = s'' \)
  - Waiting cannot lead to different states

- **Time additivity**
  - If \( s \rightarrow t_1 \rightarrow s' \) and \( s' \rightarrow t_2 \rightarrow s'' \) then \( s \rightarrow (t_1 + t_2) \rightarrow s'' \)
  - Waiting \( t_1 \) and then \( t_2 \) is the same as waiting \( (t_1 + t_2) \)
Representation of TLTSs (1/2)

• We were able to represent LTSs as graphs with labelled edges. We cannot give a similar, graphical representation of TLTS

• Let’s try anyway…

• Example: double click in a GUI
  - At time $t = 0$, user clicks the mouse button.
  - If user clicks the button again while $t \leq 0.2s$, the computer registers a double click
  - Otherwise, the computer registers a single click
The user can do the 2nd click at any moment in that 0.2 seconds timespan

\( \emptyset \) and \( S \) will have an infinite (non-countable) number of elements!
Timed automata

• A “compact” formalism to describe TLTSs

• Communicating automata + clocks
  - Clocks = variables whose values increase continuously
  - The values of all clocks increase at the same speed
  - Can be tested: is the value of $c \ (\leq, \geq, =, \neq)$ some value?
  - Can be reset to 0

• Software support: Uppaal [www.uppaal.org](http://www.uppaal.org)
TA example: double click

- ? and ! denote input and output actions
- \( x \) is a clock
  - 1st click resets \( x \) (\( x := 0 \))
  - If a 2nd click happens while \( x \leq 0.2 \), a double click is registered
  - Otherwise, a single click is registered
- (\( \bigcirc \) : initial state)
Clock conditions (1/2)

- Guards (Attached to transitions)
  - The transition is enabled iff. the guard is satisfied

- Invariants (Attached to states)
  - The invariant is true as long as the system stays in that state
  - Example: this TA outputs “hello” before $x > 5$

![Diagram](image)
Clock conditions (2/2)

- A condition can be:
  - A comparison of the value of a clock $x$ with a constant $c$
  - A comparison of $(x - x')$ with $c$
  - A negation (NOT) of a condition, or a conjunction (AND) or disjunction (OR) of conditions

$\Psi ::= x \text{ op } c \mid x - x' \text{ op } c \mid \neg \Psi \mid \Psi \land \Psi \mid \Psi \lor \Psi$

$\text{op ::= } < \mid > \mid \leq \mid \geq \mid = \mid \neq$
TA: Definition

A TA is a 6-ple \( \langle S, A, X, T, Inv, s_0 \rangle \)
- \( S \): States, \( A \): Actions, \( s_0 \): Initial state (like LTS)
- \( X \): Set of clocks
- \( T \): Transition relation: set of 6-ples \( (s, a, g, r, s') \)
  - \( s, s' \): source and target states
  - \( a \in A \): action
  - \( g \in \Psi \): a guard over clocks
  - \( r \subseteq X \): a subset of clocks that will be reset
- \( Inv : S \rightarrow \Psi \) maps each state to an invariant
- All sets are finite
Exercise: Communication medium with timeout

Complete the following CA to make a TA such that:

- Action RCV can occur between 1 and 4 TU after action SND
- If action RCV has not occurred after 4 TU, then action TIMEOUT occurs within 1 TU
Solution

Complete the following CA to make a TA such that:

- Action RCV can occur between 1 and 4 TU after action SND
- If action RCV has not occurred after 4 TU, then action TIMEOUT occurs within 1 TU
Exercise

Complete the following CA to make a TA such that:

- Action B occurs between 2 and 4 TU after action A
- Action C occurs at least 4 TU after action A and at least 1 TU after action B

Hint: use two clocks
Complete the following CA to make a TA such that:

- Action B occurs between 2 and 4 TU after action A
- Action C occurs at least 4 TU after action A and at least 1 TU after action B

Hint: use two clocks
Semantics of TA (1/2)

• General idea: associate a TLTS to every TA
  \[ \text{TA} = \langle S, A, X, T, \text{Inv}, s_0 \rangle \]
  \[ \text{TLTS} = \langle S \times V, A, \mathbb{R}^{\geq 0}, T', \Theta, (s_0, v_0) \rangle \]

• States of TLTS = (States of TA) × (clock valuation)
  - A valuation \( v : X \rightarrow \mathbb{R}^{\geq 0} \) is a function that assigns a value to every clock. \( V \) is the set of all valuations
  - \( v_0 \) is the valuation such that all clocks are set to 0.
  - \( v + t \ (t \in \mathbb{R}^{\geq 0}) \) is the valuation \( v' \) where all values in \( v \) are increased by \( t \) time units: \( \forall x \in X. \ v'(x) = v(x) + t \)

• Initial state of TLTS: \( (s_0, v_0) \)
Semantics of TA (2/2)

• \( T' \) (discrete transitions): \((s, v) \rightarrow a \rightarrow (s', v')\) iff.
  - TA contains a transition \((s, a, g, r, s')\)
  - Valuation \(v\) satisfies the guard \(g\)
  - All clocks in \(r\) are reset to 0 in \(v'\), while all other clocks have the same value in \(v\) and \(v'\)
  - \(v'\) satisfies the invariant \(Inv(s')\)

• \( \Theta \) (timed transitions): \((s, v) \rightarrow t \rightarrow (s, v+t)\) iff.
  all valuations between \(v\) and \(v+t\) satisfy \(Inv(s)\)
  - \(\forall dt \in [0, t] . \ v+dt \models Inv(s)\)
Timelock

- May arise from using invariants incorrectly
- Example:

  - What happens when \( x = 3 \)?
    - Clock \( x \) is never reset: time stops
  - Unacceptable! Either reset \( x \), or add other edges/states describing what happens when \( x = 3 \)
  - Can be detected automatically via verification
Critical paths and Zeno effect

• Example: 

![Diagram showing a critical path example]

• Critical path: infinite actions in zero time
  - \((s, \emptyset) \rightarrow a \rightarrow (s, \emptyset) \rightarrow a \rightarrow (s, \emptyset) \rightarrow a \rightarrow ...\)

• Zeno effect: infinite actions in finite time
  - \((s, \emptyset) \rightarrow a \rightarrow (s, \emptyset) \rightarrow 1/2 \rightarrow (s, \emptyset) \rightarrow a \rightarrow (s, \emptyset) \rightarrow 1/4 \rightarrow ...\)
  - Will perform an infinity of \(a\) actions in 1 time unit

• These kinds of paths are generally allowed, but it’s good to prove that time passes (there are paths that are not critical/Zeno)
Time progress

- For some time interval $t$ and some $n$, every state of the TLTS admits at least one path of length $\leq n$ such that at least $t$ time units pass.
- The system may still contain critical/Zeno paths.
- Example:
  - “aaa...” path is critical
  - “bc” path takes at least 1 time unit
Parallel composition of TA (1/2)

- Same idea as with CA: we want to decompose complex (timed) systems into small components
- Again, rendez-vous on pairs of actions according to a synchronization set $L$
  - Symmetrical (same actions)
  - Asymmetrical (input/output pairs) (e.g., Uppaal)
- But we also have to take into account:
  - Guards
  - Resets
  - Invariants
Parallel composition of TA (2/2)

• $\text{TA}_1 = \langle S_1, A_1, X_1, T_1, \text{Inv}_1, s_{01} \rangle$,
• $\text{TA}_2 = \langle S_2, A_2, X_2, T_2, \text{Inv}_2, s_{02} \rangle$ with $X_1 \cap X_2 = \emptyset$
• $L \subseteq A_1 \cap A_2$ (synchronization actions)

Then,

$\text{TA}_1 \otimes_L \text{TA}_2 = \langle S_1 \times S_2, A_1 \cup A_2, X_1 \cup X_2, T, \text{Inv}, (s_{01}, s_{02}) \rangle$

• $\text{Inv}(s_1, s_2) = \text{Inv}_1 (s_1) \land \text{Inv}_2 (s_2)$

• $T$:
  
  $s_1 \xrightarrow{g,a,r} s'_1 \quad a \notin L$
  
  $s_2 \xrightarrow{g,a,r} s'_2 \quad a \notin L$

  
  $(s_1, s_2) \xrightarrow{g,a,r} (s'_1, s'_2)$

  
  $s_1 \xrightarrow{g_1,a,r_1} s'_1$
  
  $s_2 \xrightarrow{g_2,a,r_2} s'_2 \quad a \in L$

  
  $(s_1, s_2) \xrightarrow{(g_1 \land g_2), a, (r_1 \cup r_2)} (s'_1, s'_2)$
Exercise

\[ \text{TA}_1 \otimes \text{TA}_2 = \langle S_1 \times S_2, A_1 \cup A_2, X_1 \cup X_2, T, \text{Inv}, (s_{01}, s_{02}) \rangle \]

- \( \text{Inv}(s_1, s_2) = \text{Inv}_1(s_1) \land \text{Inv}_2(s_2) \)

- \( T: \)

\[
\begin{align*}
&\begin{array}{c}
\text{s}_1 & \xrightarrow{g, a, r} & \text{s}'_1 \\
(\text{s}_1, \text{s}_2) & \xrightarrow{g, a, r} & (\text{s}'_1, \text{s}_2)
\end{array} \quad \begin{array}{c}
\text{s}_2 & \xrightarrow{g, a, r} & \text{s}'_2 \\
(\text{s}_1, \text{s}_2) & \xrightarrow{g, a, r} & (\text{s}_1, \text{s}'_2)
\end{array} \\
&\begin{array}{c}
\text{s}_1 & \xrightarrow{g_1, a, r_1} & \text{s}'_1 \\
\text{s}_2 & \xrightarrow{g_2, a, r_2} & \text{s}'_2 \\
(\text{s}_1, \text{s}_2) & \xrightarrow{(g_1 \land g_2), a, (r_1 \cup r_2)} & (\text{s}'_1, \text{s}'_2)
\end{array}
\end{align*}
\]
Solution

send, idle
\(x \leq 6\)

[send, idle]
\[x \geq 4\] SND [h := 0; x := 0]

[h > 4] TIMEOUT

[h ≥ 1 \& h ≤ 4] RCV

send, sent
\(h \leq 5 \& x \leq 6\)
Conclusions

• TA allow to describe systems where time matters
• This introduces additional complexities
  - Underlying model (TLTS) has uncountably $\infty$ states and transitions
  - Timelocks, critical paths, Zeno effect...
• We can compose TAs via a product $\otimes$
• Automated tools can verify several aspects related to TA correctness