Automata for Real-time Systems

B. Srivathsan

Chennai Mathematical Institute
Overview
Automata (*Finite State Machines*) are **good abstractions** of many real systems

hardware circuits, communication protocols, biological processes, ...
Automata can model many *properties* of systems

every request is followed by a response
System  ↓  Automaton $A$

Property  ↓  Automaton $B$
Does system satisfy property?

System

Automaton $A$

Property

Automaton $B$
$\mathcal{L}(A) \subseteq \mathcal{L}(B)$?

Does system satisfy property?
Model-checking

\[ \mathcal{L}(A) \subseteq \mathcal{L}(B) ? \]

Does system satisfy property?
In practice...

Huge system

Property

Some model-checkers: SMV, NuSMV, SPIN, ...

Turing Awards: Clarke, Emerson, Sifakis and Pnueli
In practice...

Huge system

\[ \downarrow \]

Higher-level description

Property

\[ \downarrow \]

Higher-level description

Some model-checkers: SMV, NuSMV, SPIN, ...

Turing Awards: Clarke, Emerson, Sifakis and Pnueli
In practice...

Huge system

Higher-level description

Automaton $A$

translation

Model-Checker

$L(A) \subseteq L(B)$?

Property

Higher-level description

Automaton $B$

Turing Awards: Clarke, Emerson, Sifakis and Pnueli

Some model-checkers: SMV, NuSMV, SPIN, ...
In practice...

Huge system \downarrow Higher-level description

Property \downarrow Higher-level description

\[ L(A) \subseteq L(B) ? \]

Some model-checkers: SMV, NuSMV, SPIN, …
In practice...

Huge system

Higher-level description

Automaton $A$

Property

Higher-level description

Automaton $B$

Translation

\[ L(A) \subseteq L(B)? \]

Some model-checkers: SMV, NuSMV, SPIN, ...

Turing Awards: Clarke, Emerson, Sifakis and Pnueli
Automata are **good abstractions** of many real systems
Automata are **good abstractions** of many real systems

**Our course:** Automata for **real-time** systems

Picture credits: F. Herbreteau

pacemaker, vehicle control systems, air traffic controllers, ...
Timed Automata

R. Alur and D. Dill in early 90s
Timed Automata

R. Alur and D. Dill in early 90s

Some model-checkers: UPPAAL, KRONOS, RED, ...
Goals of our course

Study language theoretic and algorithmic properties of timed automata
Lecture 1:
Timed languages and timed automata
Σ  : alphabet     \{a, b\}

Σ*  : words      \{ε, a, b, aa, ab, ba, bb, aab, \ldots \}

L  ⊆ Σ*  : language  →  property over words

$L_1 := \{\text{set of words starting with an "a"}\}$
\{a, aa, ab, aaa, aab, \ldots \}

$L_2 := \{\text{set of words with a non-zero even length}\}$
\{aa, bb, ab, ba, abab, aaaa, \ldots \}
\[ \Sigma : \text{alphabet} \quad \{a, b\} \]

\[ \Sigma^* : \text{words} \quad \{\varepsilon, a, b, aa, ab, ba, bb, aab, \ldots\} \]

\[ L \subseteq \Sigma^* : \text{language} \quad \rightarrow \quad \text{property over words} \]

\[ L_1 := \{\text{set of words starting with an "a"}\} \]
\[ \{a, aa, ab, aaa, aab, \ldots\} \]

\[ L_2 := \{\text{set of words with a non-zero even length}\} \]
\[ \{aa, bb, ab, ba, abab, aaaa, \ldots\} \]

Finite automata, pushdown automata, Turing machines, \ldots
\[ \sum : \text{alphabet} \quad \{a, b\} \]

\[ T\Sigma^* : \text{timed words} \]

\[ (a; 0.8, 2.5) \quad (abb; \pi, 203, 312.3) \]
\[ \Sigma : \text{alphabet} \quad \{a, b\} \]

\[ T\Sigma^* : \text{timed words} \]

\[ w = a_1 \ldots a_n \quad \text{where} \quad a_i \in \Sigma \]

\[ \tau = \tau_1 \ldots \tau_n \quad \text{where} \quad \tau_i \in \mathbb{R}_{\geq 0} \quad \tau_1 \leq \cdots \leq \tau_n \]

\[(aa; 0.8, 2.5) \quad (abb; \pi, 203, 312.3)\]
$L \subseteq T\Sigma^*$: Timed language $\rightarrow$ property over timed words

$L_1 := \{(ab(a+b)^*, \tau) \mid \tau_2 - \tau_1 = 1\}$

$L_2 := \{(w, \tau) \mid \tau_{i+1} - \tau_i \geq 2 \text{ for all } i < |w|\}$
$L \subseteq T\Sigma^*$: Timed language → \textit{property over timed words}

$L_1 := \{(ab(a + b)^*, \tau) | \tau_2 - \tau_1 = 1\}$

$L_2 := \{(w, \tau) | \tau_{i+1} - \tau_i \geq 2 \text{ for all } i < |w|\}$

Timed automata
Timed automaton: Finite automaton + Finite no. of *Clocks*

\[
\text{Clock} \quad 0 \quad \text{time}
\]
Timed automaton: Finite automaton + Finite no. of *Clocks*

\[
\{ ( ab(a + b)^*, \tau ) \mid \tau_2 \leq 2 \}
\]

\[
\begin{array}{c}
q_0 \rightarrow a \rightarrow q_1 \rightarrow b \rightarrow q_2 \\
\quad \quad a, b
\end{array}
\]
Timed automaton: Finite automaton + Finite no. of Clocks

\[
\{ (ab(a + b)^*, \tau) \mid \tau_2 \leq 2 \} 
\]
Timed automaton: Finite automaton + Finite no. of **Clocks**

\[
\phi := x \leq c \mid x \geq c \mid \neg \phi \mid \phi \land \phi
\]

where \( x \in \text{Clocks}, c \in \mathbb{Q}_{\geq 0} \)

\[
\{(ab(a + b)^*, \tau) \mid \tau_2 \leq 2\}
\]
Timed automaton: Finite automaton + Finite no. of Clocks

 Guards

\[ \phi := x \leq c \mid x \geq c \mid \neg \phi \mid \phi \land \phi \]

\[ x \in \text{Clocks}, \ c \in \mathbb{Q}_{\geq 0} \]

\[ \{ (ab(a + b)^*, \tau) \mid \tau_2 \leq 2 \} \]

\[ q_0 \xrightarrow{a} q_1 \xrightarrow{x \leq 2, b} q_2 \quad a, b \]

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

\[ q_0 \mid q_1 \mid q_2 \]

accept

\[ q_0 \xrightarrow{a} q_1 \xrightarrow{b} \times \]

reject
Timed automaton: Finite automaton + Finite no. of Clocks

Guards
\[ \phi := x \leq c \mid x \geq c \mid \neg \phi \mid \phi \land \phi \]
\[ x \in \text{Clocks}, \ c \in \mathbb{Q}_{\geq 0} \]

\[ \{ (ab(a+b)^*, \tau) \mid \tau_2 - \tau_1 \leq 2 \} \]

Diagram:
- States: \[ q_0, q_1, q_2 \]
- Transitions:
  - \( q_0 \rightarrow q_1 \) with label \( a \) and condition \( x \leq 2, b \)
  - \( q_1 \rightarrow q_2 \) with label \( a, b \)
  - Loop at \( q_2 \) with label \( a, b \)
Timed automaton: Finite automaton + Finite no. of Clocks

 Guards
\[ \phi := x \leq c \mid x \geq c \mid \neg \phi \mid \phi \land \phi \]
\[ x \in \text{Clocks}, \ c \in \mathbb{Q}_{\geq 0} \]

 Resets

\[ \{(ab(a + b)^*, \tau) \mid \tau_2 - \tau_1 \leq 2\} \]

\[
\begin{array}{c}
q_0 \xrightarrow{a} q_1 \xrightarrow{x \leq 2, b} q_2 \\
\{x\} \xrightarrow{a, b} \end{array}
\]
Timed automaton: Finite automaton + Finite no. of Clocks

Clock

 Guards

\[ \phi := x \leq c \mid x \geq c \mid \neg \phi \mid \phi \land \phi \]

\[ x \in \text{Clocks}, \ c \in \mathbb{Q}_{\geq 0} \]

 Resets

\[ \{ (ab(a+b)^*, \tau) \mid \tau_2 - \tau_1 \leq 2 \} \]

\[
\begin{align*}
q_0 \xrightarrow{a} q_1 & \quad \text{\{x\}} \\
q_1 \xrightarrow{x \leq 2, b} q_2 & \quad \text{\{a, b\}}
\end{align*}
\]

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>q_0</td>
<td>q_1</td>
<td>q_2</td>
</tr>
<tr>
<td>x : 0</td>
<td>x \leq 2</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>0</th>
<th>.5</th>
<th>1</th>
<th>2</th>
<th>2.5</th>
</tr>
</thead>
<tbody>
<tr>
<td>q_0</td>
<td>q_1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x : 0</td>
<td>x \leq 2</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

accept

reject
$L_3 := \{ (a^k, \tau) \mid k > 0, \tau_i = i \text{ for all } i \leq k \}$

An “$a$” occurs in every integer from $1, \ldots, k$
\[ L_3 := \{ (a^k, \tau) \mid k > 0, \tau_i = i \text{ for all } i \leq k \} \]

An “a” occurs in every integer from 1, \ldots, k

\[ \begin{align*}
0 & \quad 1 & \quad 2 & \quad 3 & \quad 4 & \quad 5 \\
q_0 & \quad a \quad a & \quad a & \quad a & \quad a & \quad a \\
q_1 & \quad \{x\} & \quad \{x\} & \quad \} & \quad \{x\} & \quad \} \\
\end{align*} \]
\[ L_4 := \{ (a^k, \tau) \mid \text{exist } i, j \text{ s.t. } \tau_j - \tau_i = 1 \} \]

There are 2 "a"s which are at distance 1 apart

```
<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>a</td>
<td>a</td>
<td>a</td>
<td>a</td>
<td>a</td>
</tr>
</tbody>
</table>
```

0 \quad t \quad t + 1
\[ L_4 := \{ (a^k, \tau) \mid \text{exist } i, j \text{ s.t. } \tau_j - \tau_i = 1 \} \]

There are 2 “a”s which are at distance 1 apart
Three mechanisms to exploit:

- **Reset**: to start measuring time
- **Guard**: to impose time constraint on action
- **Non-determinism**: for existential time constraints
\[ A = (Q, \Sigma, X, T, Q_0, F) \]
\[ T \subseteq Q \times \Sigma \times \text{guard} \times \text{reset} \times Q \]
\( A = (Q, \Sigma, X, T, Q_0, F) \)

\( T \subseteq Q \times \Sigma \times \text{guard} \times \text{reset} \times Q \)

\((ac; 0.4, 0.9)\)

\[ \begin{array}{c|c|c|c|c}
  \text{x} & 0 & 0.4 & 0.4 & 0.9 \\
  \text{y} & 0 & 0.4 & 0 & 0.5 \\
\end{array} \]
$A = (Q, \Sigma, X, T, Q_0, F)$

$T \subseteq Q \times \Sigma \times \text{guard} \times \text{reset} \times Q$

$A$ has an accepting run over $(w, \tau)$ if $(w, \tau) \in \mathcal{L}(A)$
$L_5 := \{ (abcd.\Sigma^*, \tau) \mid \tau_3 - \tau_1 \leq 2 \text{ and } \tau_4 - \tau_2 \geq 5 \}$

Interleaving distances

![Diagram showing the relationship between points a, b, c, and d with distances marked along the line.

Note: The diagram illustrates the concept of interleaving distances with specific points labeled a, b, c, and d, and distances marked along a line from 0 to 7.]
\[ L_5 := \{ (abcd.\Sigma^*, \tau) \mid \tau_3 - \tau_1 \leq 2 \text{ and } \tau_4 - \tau_2 \geq 5 \} \]

Interleaving distances
$n$ interleavings $\Rightarrow$ need $n$ clocks

$n + 1$ clocks more expressive than $n$ clocks
Timed automata

Runs

1 clock < 2 clocks < ...
$L_6 := \{ (a^k, \tau) \mid \tau_i \text{ is some integer for each } i \}$
$L_6 := \{ (a^k, \tau) \mid \tau_i \text{ is some integer for each } i \}$

Claim: No timed automaton can accept $L_6$
Step 1: \textit{Suppose} $L_6 = \mathcal{L}(A)$

Let $c_{\text{max}}$ be the maximum constant appearing in a guard of $A$
Step 1: Suppose $L_6 = \mathcal{L}(A)$

Let $c_{max}$ be the maximum constant appearing in a guard of $A$

Step 2: For a clock $x$,

$x = \lceil c_{max} \rceil + 1$ and $x = \lceil c_{max} \rceil + 1.1$ satisfy the same guards
Step 1: *Suppose* $L_6 = \mathcal{L}(A)$

Let $c_{\text{max}}$ be the maximum constant appearing in a guard of $A$

Step 2: For a clock $x$,

$$x = \lceil c_{\text{max}} \rceil + 1 \quad \text{and} \quad x = \lceil c_{\text{max}} \rceil + 1.1$$

satisfy the same guards

Step 3: $(a; \lceil c_{\text{max}} \rceil + 1) \in L_6$ and so $A$ has an accepting run

$$(q_0, v_0) \xrightarrow{\delta = \lceil c_{\text{max}} \rceil + 1} (q_0, v_0 + \delta) \xrightarrow{a} (q_F, v_F)$$
Step 1: Suppose $L_6 = \mathcal{L}(A)$
Let $c_{\text{max}}$ be the maximum constant appearing in a guard of $A$

Step 2: For a clock $x$,

$$x = \lceil c_{\text{max}} \rceil + 1$$
and

$$x = \lceil c_{\text{max}} \rceil + 1.1$$

satisfy the same guards

Step 3: $(a; \lceil c_{\text{max}} \rceil + 1) \in L_6$ and so $A$ has an accepting run

$$\left( q_0, v_0 \right) \xrightarrow{\delta = \lceil c_{\text{max}} \rceil + 1} \left( q_0, v_0 + \delta \right) \xrightarrow{a} \left( q_F, v_F \right)$$

Step 4: By Step 2, the following is an accepting run

$$\left( q_0, v_0 \right) \xrightarrow{\delta' = \lceil c_{\text{max}} \rceil + 1.1} \left( q_0, v_0 + \delta' \right) \xrightarrow{a} \left( q_F, v'_F \right)$$
Step 1: Suppose $L_6 = \mathcal{L}(A)$
Let $c_{max}$ be the maximum constant appearing in a guard of $A$

Step 2: For a clock $x$,

$$x = \lceil c_{max} \rceil + 1 \text{ and } x = \lceil c_{max} \rceil + 1.1$$

satisfy the same guards

Step 3: $(a; \lceil c_{max} \rceil + 1) \in L_6$ and so $A$ has an accepting run

$$\delta = \lceil c_{max} \rceil + 1 \quad \begin{array}{c}
(q_0, v_0) \\
\xrightarrow{\delta} \\
(q_0, v_0 + \delta) \\
\xrightarrow{a} \\
(q_F, v_F)
\end{array}$$

Step 4: By Step 2, the following is an accepting run

$$\delta' = \lceil c_{max} \rceil + 1.1 \quad \begin{array}{c}
(q_0, v_0) \\
\xrightarrow{\delta'} \\
(q_0, v_0 + \delta') \\
\xrightarrow{a} \\
(q_F, v'_F)
\end{array}$$

Hence $(a; \lceil c_{max} \rceil + 1.1) \in \mathcal{L}(A) \neq L_6$

Therefore no timed automaton can accept $L_6$
Timed automata

Runs
1 clock < 2 clocks < . . .

Role of max constant