Introduction
Semantics
Extending CHMs with time

Hierarchy and Concurrency in Finite State Machines

G. Caravagna
caravagn@cli.di.unipi.it

Automata Theory

*FSMs are useful to represent the flow of control*

*and are amenable to formal analysis such as model checking*
1 Introduction
   - Finite State Machine
   - Adding Hierarchy and Concurrency

2 Semantics
   - Semantics of CHSMs

3 Extending CHMs with time
   - CHTTAs
Index

1. Introduction
   - Finite State Machine
   - Adding Hierarchy and Concurrency

2. Semantics
   - Semantics of CHSMs

3. Extending CHMs with time
   - CHTTAs
Definition (FHM)

The canonical definition is a 5-tuple: \( \langle Q, \Sigma, q^i, q^f, \rightarrow \rangle \)

- a finite set of states \( Q \)
- a finite alphabet \( \Sigma \)
- an initial state \( q^i \in Q \)
- a final state \( q^f \in Q \)
- a set of transitions \( \rightarrow \subseteq Q \times \Sigma \times Q \)

i.e.

\[
M = \langle \{0, 1\}, \{a, b\}, 0, 1, \{(0, a, 1), (1, b, 1)\} \rangle
\]
Accepting Run in a FSM

Given a word on $\Sigma$: $\rho = \sigma_0\sigma_1 \ldots \sigma_n$, it's accepting run is the sequence

$$q_0 \xrightarrow{\sigma_0} q_1 \xrightarrow{\sigma_1} \ldots q_n \xrightarrow{\sigma_n} q_{n+1}$$

such that

- $q_0 = q^i$, $q_{n+1} = q^f$
- $\forall 0 \leq i \leq n. \ (q_i, \sigma_i, q_{i+1}) \in \rightarrow$

Language accepted by a FSM is

$$\mathcal{L}(M) = \{ w \in \Sigma^* \mid w \text{ has an accepting run in } M \}$$
Accepted Language, an example

\[ M = \langle \{0, 1\}, \{a, b\}, 0, 1, \{(0, a, 1), (1, b, 1)\} \rangle \]

\[ 0 \xrightarrow{a} 1 \xrightarrow{b} 1 \xrightarrow{b} 1 \in \mathcal{L}(M) = \{a b^*\} \]

Automata are FSM!!
Accepted Language, an example

\[ M = \langle \{0, 1\}, \{a, b\}, 0, 1, \{(0, a, 1), (1, b, 1)\} \rangle \]

\[ 0 \xrightarrow{a} 1 \xrightarrow{b} 1 \xrightarrow{b} 1 \in \mathcal{L}(M) = \{ab^*\} \]

Automata are FSM !!
Index

1 Introduction
- Finite State Machine
- Adding Hierarchy and Concurrency

2 Semantics
- Semantics of CHSMs

3 Extending CHMs with time
- CHTTAs
What we mean with

**concurrency** a composition of machines which synchronize on transitions labeled with common alphabet symbols

**hierarchy** a state of a machine can be another machine (*superstate*)
Concurrent Hierarchical State Machine (CHM)

Definition (CHM)

**base case** Any FSM is a CHM

**concurrency** If $M_1, M_2, \ldots, M_n$ are CHMs, then

$$M_1 \parallel M_2 \parallel \ldots \parallel M_n$$

is a CHM

**hierarchy** If $M = \{M_1, M_2, \ldots, M_n\}$ are CHMs and $N$ is a FSM with states $Q$, then

$$(N, M, \mu : Q \mapsto M)$$

is a CHM
Sample: the power of hierarchy

Suppose $HSM = " CHM without concurrency"$. Say $\Sigma = \{\sigma\}$ and build a sequence of $M_i, \ldots, M_n$ where $M_i$ is a HSM.

- $M_1$ is a FSM with $L(M_1) = \{\sigma\}$.
- $M_{j>1}$ is a HSM with

$$L(M_j) = L(M_{j-1}) \cdot \sigma \cdot L(M_{j-1})$$

such that:
- $M_j$ has two superstates 0, 1 mapped to $M_{j-1}$

$$L(M_j) \overset{?}{=} \{s\} \text{ where } |s| = 2^j - 1$$
Sample: the power of hierarchy

Suppose $HSM = "CHM without concurrency"$. Say $\Sigma = \{\sigma\}$ and build a sequence of $M_1, \ldots, M_n$ where $M_i$ is a HSM.

- $M_1$ is a FSM with $L(M_1) = \{\sigma\}$.
- $M_{j>1}$ is a HSM with
  
  $L(M_j) = L(M_{j-1}) \cdot \sigma \cdot L(M_{j-1})$

such that:

- $M_j$ has two superstates 0, 1 mapped to $M_{j-1}$

  $L(M_j) \overset{?}{=} \{s\}$ where $|s| = 2^j - 1$
Special cases of FSM

- Acceptors and transducers
  - Automata for regular languages
  - Moore and Mealy Machines

- Place/Transition Petri Nets

- Statecharts and UML

- Labeled Transition Systems
1 Introduction
   - Finite State Machine
   - Adding Hierarchy and Concurrency

2 Semantics
   - Semantics of CHSMs

3 Extending CHMs with time
   - CHTTAs
We map each CHM $M$ in a FSM $\llbracket M \rrbracket$

$\llbracket - \rrbracket : \text{CHM} \rightarrow \text{FSM} \quad \mathcal{L}(M \in \text{CHM}) \overset{\text{def}}{=} \mathcal{L}(\llbracket M \rrbracket)$

Semantics will be given in terms of a FSM!
Definition ($\llbracket M \rrbracket$)

**base case** Any FSM $M = \langle Q, \Sigma, q^i, q^f, \rightarrow \rangle$ has semantic

$$\llbracket M \rrbracket \overset{\text{def}}{=} M$$
Definition ($[[M]]$)

Any CHM $M_{\parallel} = M_1 \parallel M_2 \parallel \ldots \parallel M_n$ such that

$[[M_i]] \overset{\text{def}}{=} \langle Q_i, \Sigma_i, q^i_1, q^f_i, \rightarrow_i \rangle$ has semantic

$[[M_{\parallel}]] \overset{\text{def}}{=} [\langle Q_1, \Sigma_1, q^1_1, q^f_1, \rightarrow_1 \rangle] \times [\langle Q_2, \Sigma_2, q^1_2, q^f_2, \rightarrow_2 \rangle] \times \ldots \times [\langle Q_n, \Sigma_n, q^1_n, q^f_n, \rightarrow_n \rangle]$

- $Q_{\parallel} = Q_1 \times \ldots \times Q_n$
- $\Sigma_{\parallel} = \Sigma_1 \cup \ldots \cup \Sigma_n$
- $q^i_{\parallel} = \langle q^i_1, \ldots, q^i_n \rangle$ and $q^f_{\parallel} = \langle q^f_1, \ldots, q^f_n \rangle$

what about transitions?
Definition ($\llbracket M \rrbracket$)

**conc.** Any CHM $M \parallel = M_1 \parallel M_2 \parallel \ldots \parallel M_n$ such that

$\llbracket M_i \rrbracket \stackrel{def}{=} \langle Q_i, \Sigma_i, q_i^i, q_i^f, \rightarrow_i \rangle$ has semantic

$$\llbracket M \parallel \rrbracket \stackrel{def}{=} \llbracket M_1 \rrbracket \times \llbracket M_2 \rrbracket \times \ldots \times \llbracket M_n \rrbracket$$

- $Q \parallel = Q_1 \times \cdots \times Q_n$
- $\Sigma \parallel = \Sigma_1 \cup \cdots \cup \Sigma_n$
- $q_i^i = \langle q_1^i, \ldots, q_n^i \rangle$ and $q_i^f = \langle q_1^f, \ldots, q_n^f \rangle$

- what about transitions?
Definition ($[M]$)

conce.

$[M]$ has a $\sigma$-transition from $\langle q_1, \cdot \cdot \cdot, q_n \rangle$ to $\langle w_1, \cdot \cdot \cdot, w_n \rangle$ iff

- $\sigma \in \Sigma_i \implies (q_i, \sigma, w_i) \in \rightarrow_i$
- $q_i = w_i$ otherwise

Which language are we defining?
Introduction
Semantics
Extending CHMs with time

Sample 1/3

### Semantics of CHSMs

{$Q_1 = \{0, 1\}$}
{$\Sigma_1 = \{a, b\}$}
$
\rightarrow_1 = \{(0, a, 1), (1, b, 1)\}$

Parallel composition is
{$Q_\parallel = Q_1 \times Q_2 = \{00, 01, 10, 11\}$}
{$\Sigma_\parallel = \Sigma_1 \cup \Sigma_2 = \{a, b\}$}
{$q^i_\parallel = 00 \quad q^f_\parallel = 11$}
{$\rightarrow_\parallel = \{(00, a, 01), (10, b, 11)\}$}

$L_1 = ab^* \quad L_2 = a^*b$
$L_\parallel = ab$

G.Caravagna
Hierarchy and Concurrency in Finite State Machines
Parallel composition is

\[ L_1 = (ac)^* \quad L_2 = (ab)^* \]

\[ L_\parallel = (a(bc|cb))^* \]
Building [ ]

**Definition ([M])**

Any CHM $M = (N, M, \mu)$ with $N$ top-level FSM, has

- $Q_H = \{(q, w)\}$ were $q \in Q_N$ and $w \in \llbracket \mu(q) \rrbracket$
- $\Sigma_H = \Sigma_N \cup \Sigma_{\llbracket \mu(q) \rrbracket}$ for each $q \in Q_N$
- $q^i_H = (q^i_N, \llbracket \mu(q^i_N) \rrbracket)$ and $q^f_H = (q^f_N, \llbracket \mu(q^f_N) \rrbracket)$

- what about transitions?

We have two kind of transitions!

- for top-level machine
- for hierarchical machines
Definition ($\llbracket M \rrbracket$)

Any CHM $M = (N, M, \mu)$ with $N$ top-level FSM, has

- $Q_H = \{(q, w)\}$ were $q \in Q_N$ and $w \in \llbracket \mu(q) \rrbracket$
- $\Sigma_H = \Sigma_N \cup \Sigma_{\llbracket \mu(q) \rrbracket}$ for each $q \in Q_N$
- $q^i_H = (q^i_N, \llbracket \mu(q^i_N) \rrbracket)$ and $q^f_H = (q^f_N, \llbracket \mu(q^f_N) \rrbracket)$

- what about transitions?

We have two kind of transitions!
- for top-level machine
- for hierarchical machines
Definition ([M])

\textit{conc.} (Top-level transition) for each \((q, \sigma, w) \in \rightarrow_N\)

\(( (q, q_f^{\mu(q)}), \sigma, (w, q_i^{\mu(w)}) ) \in \rightarrow_H\)
Definition ([M])

conc. (Hierarchial transition) for each \((w, \sigma, w') \in \rightarrow [\mu(q)]\) when \(q \in Q_N\)

\[
( (q, w), \sigma, (q, w') ) \in \rightarrow_H
\]
Introducing CHMs with time

Semantics of CHSMs

Semantics

Sample 3/3

\[ L_1 = ab^* \quad L_2 = a^*b \]

\[ L_H = L_1 \cdot \sigma \cdot (L_2 \cdot \tau)^+ = ab^* \sigma(a^*(b\tau)^*)^*b \]
Index

1 Introduction
   - Finite State Machine
   - Adding Hierarchy and Concurrency

2 Semantics
   - Semantics of CHSMs

3 Extending CHMs with time
   - CHTTAs
What we model

- **ACID transactions**
  - atomic
  - locks on resources
  - roll-back

- **long-running transactions**
  - sequential, parallel tasks
  - sub-tasks
  - deadline
  - compensations
Long-running transactions

- composed by atomic activities $A_i$
  - not allowed partial execution
  - all activities $A_i$ have a compensation (atomic) activity $B_i$

"$B_i$ repairs from the effects of successful execution of $A_i$"

i.e. Sequential Transactions
$A_1, \ldots, A_n$ such that $A_{i+1}$ starts when $A_i$ is committed.

Scenario:
- or $A_1, \ldots, A_n$ is executed.
- or $A_1, \ldots, A_1, B_i, \ldots, B_1$ is executed for $i < n$. 
Communicating Hierarchical Transaction-based Timed Automata

Definition (CHTTA)
informally are extensions of CHMs with:

- different terminal states
  - commit state
  - abort state
- different communication mechanisms
  - private channels (restrictions)
  - read/write actions
- explicit notion of time
Some interesting rules of the SOS semantics are

\[(q, \alpha, \phi, B, q') \in \delta \quad \nu \models \phi \quad q' \notin S\]

\[\langle A, \mu \rangle, (q, \nu) \xrightarrow{\alpha} \langle A, \mu \rangle, (q', \nu[B])\]

\[(q, \alpha, \phi, B, q') \in \delta \quad \nu \models \phi \quad q' \notin S \quad \text{Init}(\mu(q)) = (c, \nu')\]

\[\langle A, \mu \rangle, (q, \nu) \xrightarrow{\alpha} \langle A, \mu \rangle, (q'.c, \nu[B].\nu')\]

\[(A_1, (c_1, \nu)) \xrightarrow{a!} (A_1, (c'_1, \nu')) \quad (A_2, (c_2, \nu')) \xrightarrow{a?} (A_2, (c''_2, \nu''))\]

\[(A_1 \parallel A_2, (c_1; c_2, \nu)) \xrightarrow{\tau} (A_1 \parallel A_2, (c'_1; c''_2, \nu''))\]
Fig. 7. A Double Request

We model a typical all–or–nothing scenario in which a client performs two concurrent requests to two different servers, waits for replies, and sends back acknowledgements either to both servers (if it receives both replies) or to none of them (if it receives at most one reply).
client sends the request to the server by synchronizing on channel \textit{req}_i and waits for the reply as a synchronization on channel \textit{rep}_i. The time deadline for the reply is \(T_i\). This is expressed as a constraint on the value of clock \(x_i\) which is set to zero when the request is sent. If the reply is received in time, the transaction commits, otherwise a stop message is sent to the server as a synchronization on channel \textit{stop}_i, and the transaction is aborted. The compensation of this transaction consists in a synchronization on channel \textit{cancel}_i, which corresponds to sending an undo message to the server.
A server is modeled by the automaton given in Figure 7 (b). We denote such an automaton with $S_i$. The server receives a request and sends the reply by synchronizing on the proper channels, and it spends a time between these two synchronizations which is greater than $R_i$. This amount of time models the time spent by the server to satisfy the request of the client. Then, the server reaches a state in which it waits for either an acknowledge or an undo message from the client. These two communications are modeled as synchronizations on channels $ack_i$ and $cancel_i$, respectively, and lead to commit and abort of the server activity, respectively.
The activity of sending acknowledgments to two servers $S_1$ and $S_2$ is modeled by the transaction given in Figure 7 (c). We denote such a transaction with $A_{ack} \triangleright B_{ack}$. Finally, the whole client transaction in which two requests are sent to two different servers and the corresponding acknowledgments are sent if both requests are satisfied, is modeled by the long-running transaction $T = (A_1 \triangleright B_1 || A_2 \triangleright B_2) \cdot A_{ack} \triangleright B_{ack}$ and the whole system in which both the client and the two servers are modeled is $SYSTEM = T || S_1 || S_2$. 
Sample Webserver: Model Checking

<table>
<thead>
<tr>
<th></th>
<th>$T_1 &gt; R_1$</th>
<th>$T_1 &lt; R_1$</th>
<th>$T_2 &gt; R_2$</th>
<th>$T_2 &lt; R_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>$A \diamond (T. \odot \lor T. \otimes)$</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>2.</td>
<td>$(A_1. \odot \lor A_2. \otimes) \leadsto T. \otimes$</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>3.</td>
<td>$(A_1. \odot \land A_2. \otimes) \leadsto T. \odot$</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>4.</td>
<td>$T. \odot \leadsto (S_1. \odot \land S_1. \otimes)$</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>5.</td>
<td>$x_1 &gt; T_1 \leadsto T. \otimes$</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>6.</td>
<td>$x_2 &gt; T_2 \leadsto T. \odot$</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>7.</td>
<td>$E \diamond T. \odot$</td>
<td>true</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>8.</td>
<td>$E \diamond T. \otimes$</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
</tbody>
</table>

Properties 1–3 express the correctness of the encoding of long-running transactions into automata. These properties must be satisfied for any setting of the parameters. In particular, property 1 says that either the commit or the abort states of the transaction (denoted $T. \odot$ and $T. \otimes$, respectively) must be eventually reached. Property 2 requires that if at least one of the abort states of the parallel activities $A_1$ and $A_2$ is reached, then the whole transaction must reach its abort state, and property 3 requires that if both parallel activities $A_1$ and $A_2$ reach their commit states, then the whole transaction must reach its commit state.
R. Alur, S. Kannan and M. Yannakakis
"Communicating Hierarchical State Machines".
LNCS, 1644: p.169–??, 1999

R. Lanotte, A. Maggiolo-Schettini, P. Milazzo, A. Troina
"Modeling Long-Running Transactions with Communicating Hierarchical Timed Automata".
LNCS, 4037: p.108–122, 2006