Synthesis by projection

MSC M3

MSC M4

Protocol synthesis from requirements – p. 15/29
A HMSC $H$ is **local** if for every pair of sequences of MSCs
\[ \rho = M_1 \circ M_2 \circ \cdots \circ M_k \] and
\[ \rho' = M'_1 \circ M'_2 \circ \cdots \circ M'_q \]
starting from the same node, there exists **only one** process which is the unique deciding process for $\rho$ and $\rho'$. 
Local HMSCs

Theorem 1 Deciding if a HMSC $H$ is in $\text{co-NP}$. Furthermore, for local HMSCs, $\mathcal{L}(H) \subseteq \mathcal{L}(T(H))$

Inclusion only ?! What’s the problem?
Synchronization loss
Synchronization loss

Problem: ordering loss.

\[ \mathcal{L}(T(H)) \cap \mathcal{L}(H) \]

\[ \in \mathcal{L}(T(H)) \text{ but } \notin \mathcal{L}(H) \]
Asynchronous control

Let $H$ be a local HMSC over a set of bMSCs $\mathcal{B}$.

We can tag any event in an execution of $H$ with a vector $\mathbb{N}^\mathcal{B}$ counting the number of times each bMSC was used.

For each execution, the set of tags used forms a total order.
Asynchronous control

maintaining tags in synthesized CFSM

avoiding ordering loss problem
Asynchronous control

Architecture: add controllers that intercept and tag messages, and can delay receptions.
Asynchronous control

Each automaton $A_p$ communicates only via its controller $C_p$

Sending: $p!q(m)$ becomes $p!C_p(m, q)$

Reception: $p?q(m)$ becomes $p?C_p(m, q)$

Each controller $C_p$ maintains a local tag $\tau_p$, and has FIFO communication buffers with other controllers.
Asynchronous control

Controller $C_p$ : reception from automaton $A_p : C_p ? p(m, q)$

- check if this message starts a new bMSC, update vector clock $\tau_p$ accordingly
- send to controller $C_q : C_p ! C_q(p, m, \tau_p)$

Controller $C_p$ monitoring a queue: $C_p ? C_q(m, \tau_q)$ (applied on every queue)

- Check if this message should be consumed (i.e. clock $\tau_q$ is a successor of $\tau_p$)
- If yes, forward to $p : C_p ! p(m, q)$, and update $\tau_q$
- if not leave the message in queue
Correctness

Let $T'(.)$ be the algorithm that synthesizes automata and their controllers from $H$.

We can not require $\mathcal{L}(T'(H)) = \mathcal{L}(H)$

Now : hide controllers and compare

Let $w \in \mathcal{L}(T'(H))$.

- $\text{Unc}(w)$ is the projection of $w$ on automata’s actions (all $C_p!xxx(yyy)$ disappear)
- $\text{Ren}(w)$ renaming of actions as for algo. $T(.)$
  
  $p!C_p(m, q) \rightarrow p!q(m)$
Correctness

Proposition 1  Let $w \in \mathcal{L}(H)$, $w' \in \mathcal{L}(T'(H))$ be two words such that $w' = \text{Ren}(\text{Unc}(w))$. Then the tags used in $H$ and in $T(H)$ during the execution of $w$ and $w'$ are identical.

Theorem 2  Let $H$ be a local choice HMSC. Then

\[
\mathcal{L}(H) = \text{Ren}\left(\text{Unc}\left(\mathcal{L}(T'(H))\right)\right)
\]
Conclusion

- Local choice HMSCs can all be implemented by projection
- Asynchronous control: simple delay of communications
- What if we have time constraints?
- Can we ensure more than correctness (some bound on channels, a property $\varphi$, ...)?
Reconstructible HMSCs

Definition 1  A HMSC is reconstructible iff, for every sequence $\rho = M_1 \circ M_2 \circ \cdots \circ M_k$ labeling a cycle of $H$, and every sequence $\rho' = M'_1 \circ M'_2 \circ \cdots \circ M'_q$ starting from the same node, for every $p \in P$, the ordering between the maximal event in $\rho$ on $p$ and the minimal event of $\rho'$ in $\rho \circ \rho'$ can be "inferred" from the behavior of all other processes.

In short: no global coordination is lost during projection.
Theorem 3  Let $H$ be a reconstructible HMSC, then $\mathcal{L}(A_H) = \mathcal{L}(H)$