Dipartimento di
Informatica e Sistemistica
"Antonio Ruberti"
SAPIENZA
UNIVERSITÀ DI ROMA

Composition via Simulation

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
SAPIENZA Università di Roma

Bisimulation

- A binary relation R is a **bisimulation** iff:

  $(s,t) \in R$ implies that
  - s is *final* iff t is *final*
  - for all actions a
    - if $s \rightarrow_a s'$ then $\exists\, t' .\ t \rightarrow_a t'$ and $(s',t') \in R$
    - if $t \rightarrow_a t'$ then $\exists\, s' .\ s \rightarrow_a s'$ and $(s',t') \in R$

- A state s_0 of transition system S is **bisimilar**, or simply **equivalent,** to a state t_0 of transition system T iff there **exists** a **bisimulation** between the initial states s_0 and t_0.

- Notably
  - **bisimilarity** is a bisimulation
  - **bisimilarity** is the largest bisimulation

Note it is a co-inductive definition!

Computing Bisimilarity on Finite Transition Systems

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
SAPIENZA Università di Roma

Algorithm ComputingBisimulation
Input: transition system $TS_S = < A, S, S^0, \delta_S, F_S>$ and
transition system $TS_T = < A, T, T^0, \delta_T, F_T>$
Output: the **bisimilarity** relation (the largest bisimulation)

Body

$R = \emptyset$

$R' = S \times T - \{(s,t) \mid \neg(s \in F_S \equiv t \in F_T)\}$

while $(R \neq R')$ {

$R := R'$

$R' := R' - (\{(s,t) \mid \exists s',a.\ s \rightarrow_a s' \wedge \neg\exists t'.\ t \rightarrow_a t' \wedge (s',t') \in R' \}$
$\{(s,t) \mid \exists t',a.\ t \rightarrow_a t' \wedge \neg\exists s'.\ s \rightarrow_a s' \wedge (s',t') \in R' \})$

}

return R'

Ydob

Simulation

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
SAPIENZA Università di Roma

- A binary relation R is a **simulation** iff:

  (s,t) $\in R$ implies that
  - s is *final* implies that t is *final*
  - for all actions a
    - if $s \rightarrow_a s'$ then $\exists\, t' .\ t \rightarrow_a t'$ and $(s',t') \in R$

- A state s_0 of transition system S is **simulated by** a state t_0 of transition system T iff there **exists** a **simulation** between the initial states s_0 and t_0.

- Notably
  - **simulated-by** is a simulation
  - **simulated-by** is the largest simulation

Note it is a co-inductive definition!

- NB: A simulation is just one of the two directions of a bisimulation

Computing Simulation on Finite Transition Systems

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
SAPIENZA Università di Roma

Algorithm ComputingSimulation
Input: transition system TS_S = < A, S, S^0, δ_S, F_S> and
transition system TS_T = < A, T, T^0, δ_T, F_T>
Output: the **simulated-by** relation (the largest simulation)

Body

$R = S \times T$

$R' = S \times T - \{(s,t) \mid s \in F_S \wedge \neg(t \in F_T)\}$

while ($R \neq R'$) {

$R := R'$

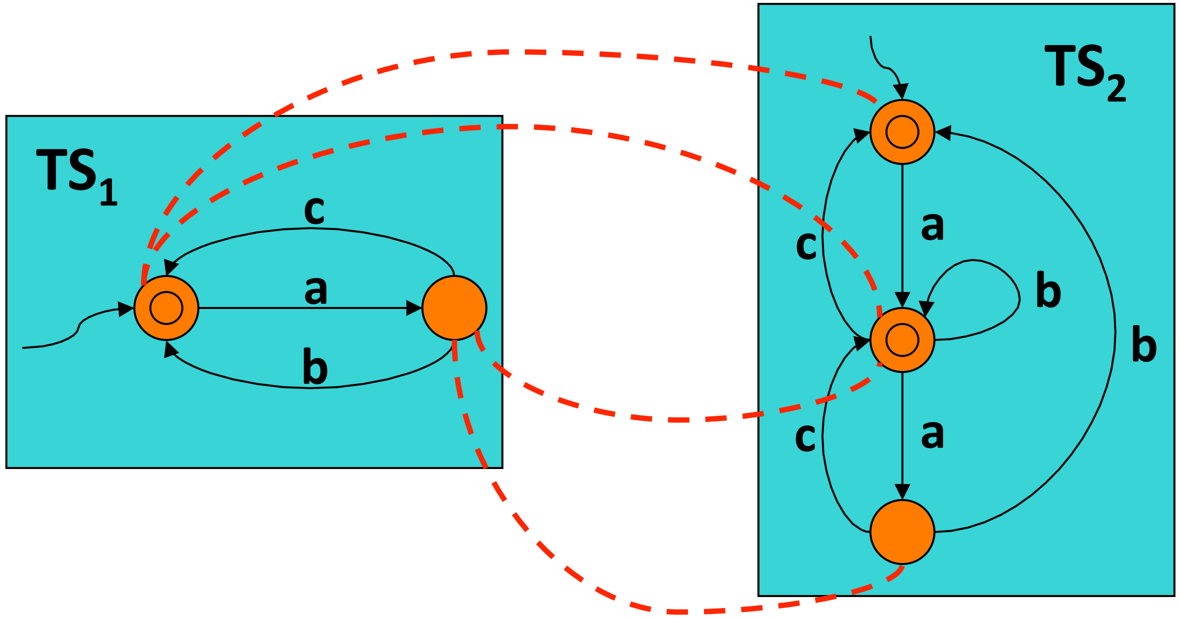
$R' := R' - \{(s,t) \mid \exists s',a.\ s \rightarrow_a s' \wedge \neg\exists t'.\ t \rightarrow_a t' \wedge (s',t') \in R' \}$

}

return R'

Ydob

Example of simulation



TS2's behavior "includes" TS1's

Potential Behavior of the Whole Community

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
SAPIENZA Università di Roma

- The potential behavior of the whole community is obtained by executing concurrently all TSs allowing for all possible interleaving (no synchronization).

- Formally we need to do the **asynchronous product** of the TSs.

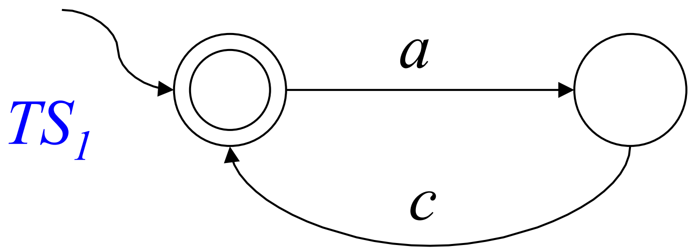
- Let $TS_1, \cdots, TS_n$ be the TSs of the component services. The **asynchronous product** of $TS_1, \cdots, TS_n$, (also called the **Community TS**) is defined as: $TS_c = < A, S_c, S_c^0, \delta_c, F_c >$ where
  - A is the set of actions
  - $S_c = S_1 \times \cdots \times S_n$
  - $S_c^0 = \{(s^0{}_1, \cdots, s^0{}_n)\}$
  - $F \subseteq F_1 \times \cdots \times F_n$
  - $\delta_c \subseteq S_c \times A \times S_c$ is defined as follows:

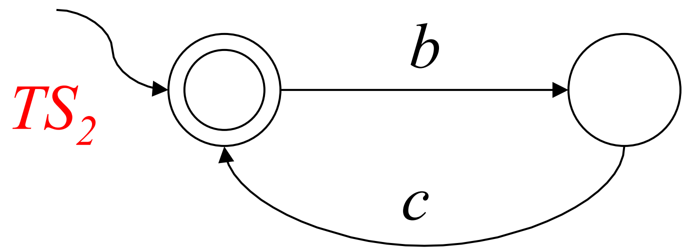
    $(s_1, \cdots, s_n) \rightarrow_a (s'_1, \cdots, s'_n)$ iff

    1. $\exists\, i.\ s_i \rightarrow_a s'_i \in \delta_i$
    2. $\forall\, j \neq i.\ s'_j = s_j$

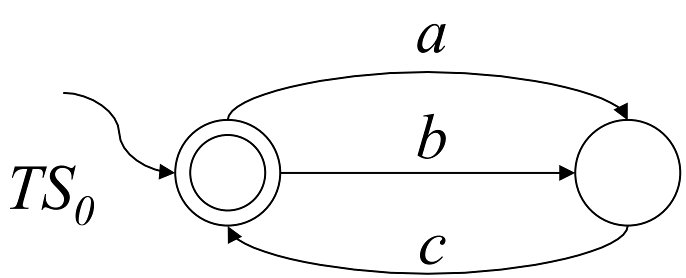
Example of Composition

- Available Services





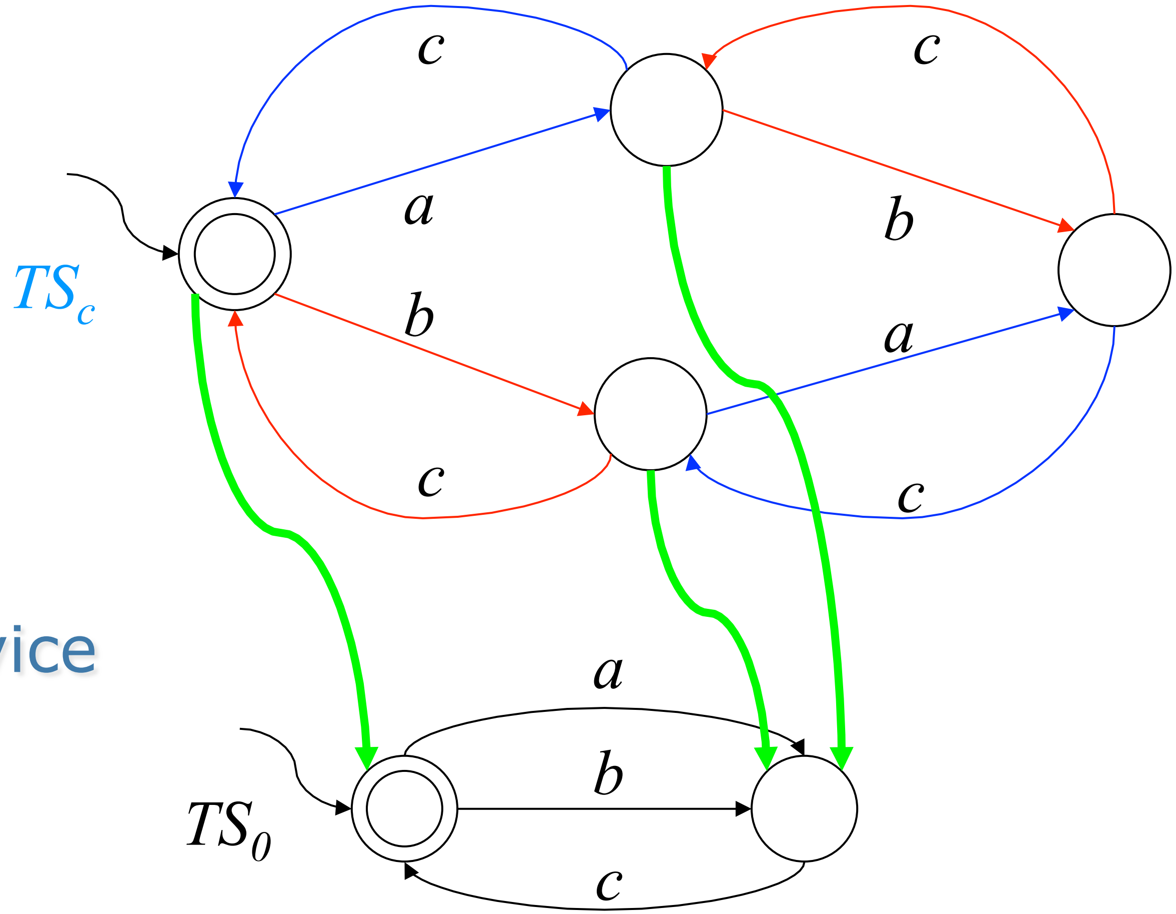
- Target Service



Dipartimento di
Informatica e Sistemistica
"Antonio Ruberti"
SAPIENZA
Università di Roma

Example of Composition

Community TS



Target Service

Composition exists!

Composition via Simulation

- **Thm[IJFCS08]**
  A composition realizing a target service TS TS_t exists if there **exists** a simulation relation between the initial state s_t^0 of TS_t and the initial state $(s_1^0, .., s_n^0)$ of the community TS TS_c.

- Notice if we take the union of all simulation relations then we get the largest simulation relation ***S***, still satisfying the above condition.

- **Corollary[IJFCS08]**
  A composition realizing a target service TS TS_t exists
  iff $(s_t^0 , (s_1^0, .., s_n^0)) \in S$.

- **Thm[IJFCS08]**
  Computing the largest simulation ***S*** is polynomial in the size of the target service TS and the size of the community TS…
- ... hence it is EXPTIME in the size of the available services.

Composition via Simulation

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
SAPIENZA Università di Roma

- Given the largest simulation ***S*** form TS_t to TS_c (which include the initial states), we can build the ***orchestrator generator***.

- This is an orchestrator program that can change its behavior reacting to the information acquired at run-time.

- Def: $OG = < A, [1,\ldots,n], S_r, s_r^0, \omega_r, \delta_r, F_r >$ with
  - A : the **actions** shared by the community
  - [1,…,n]: the **identifiers** of the available services in the community
  - $S_r = S_t \times S_1 \times \cdots \times S_n$: the **states** of the orchestrator program
  - $s_r^0 = (s^0_t, s^0_1, \ldots, s^0_m)$: the **initial state** of the orchestrator program
  - $F_r \subseteq \{ (s_t , s_1 , \ldots, s_n) \mid s_t \in F_t$: the **final states** of the orchestrator program
  - $\omega_r : S_r \times A_r \to [1,\ldots,n]$: the **service selection function**, defined as follows:

    $\omega_r(t, s_1,..,s_n, a) = \{ i \mid$ TS_t and **TS_i** can do **a** and remain in ***S*}**

    i.e., …= $\{i \mid s_t \to_{a,} s'_t \wedge \exists s_i'. s_i \to_{a,} s_i' \wedge (s_t', (s_1 , \ldots, s'_i , \ldots, s_n)) \in$ ***S***$\}$

  - $\delta_r \subseteq S_r \times A_r \times [1,\ldots,n] \to S_r$: the **state transition function**, defined as follows:
    Let $k \in \omega_r(s_t, s_1 , \ldots, s_k , \ldots, s_n, a)$ then
    $(s_t, s_1 , \ldots, s_k , \ldots, s_n) \to_{a,k} (s_t', s_1 , \ldots, s'_k , \ldots, s_n)$ where $s_k \to_{a,} s'_k$

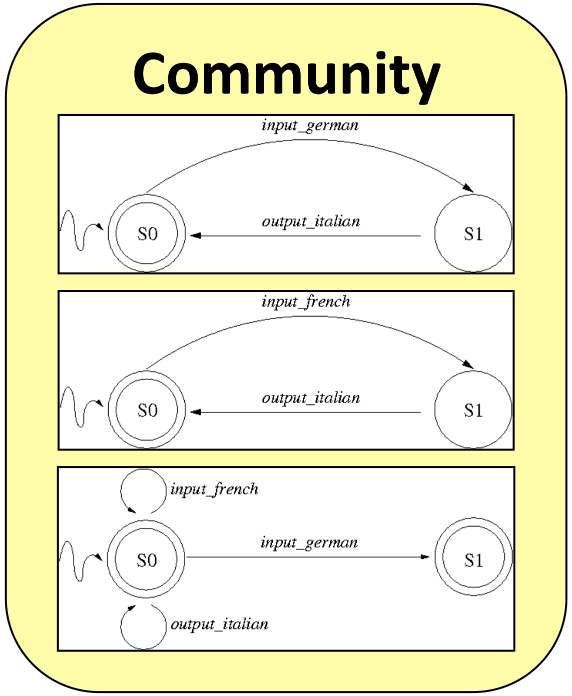
Composition via Simulation

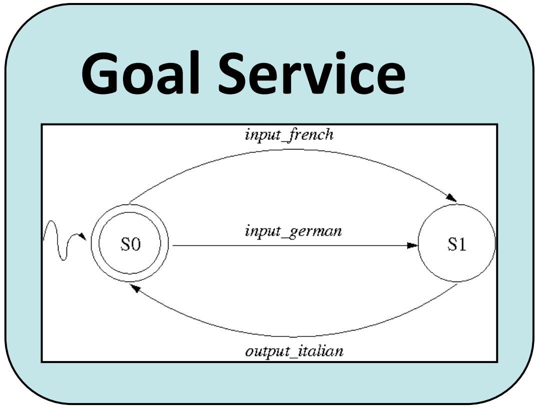
Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
SAPIENZA Università di Roma

- For generating OG we need only to compute ***S*** and then apply the template above
- For running an orchestrator from the OG we need to store and access ***S*** *(polynomial time, exponential space)* ...
- ... and compute ω_r and δ_r at each step *(polynomial time and space)*

Example of composition via simulation (1)

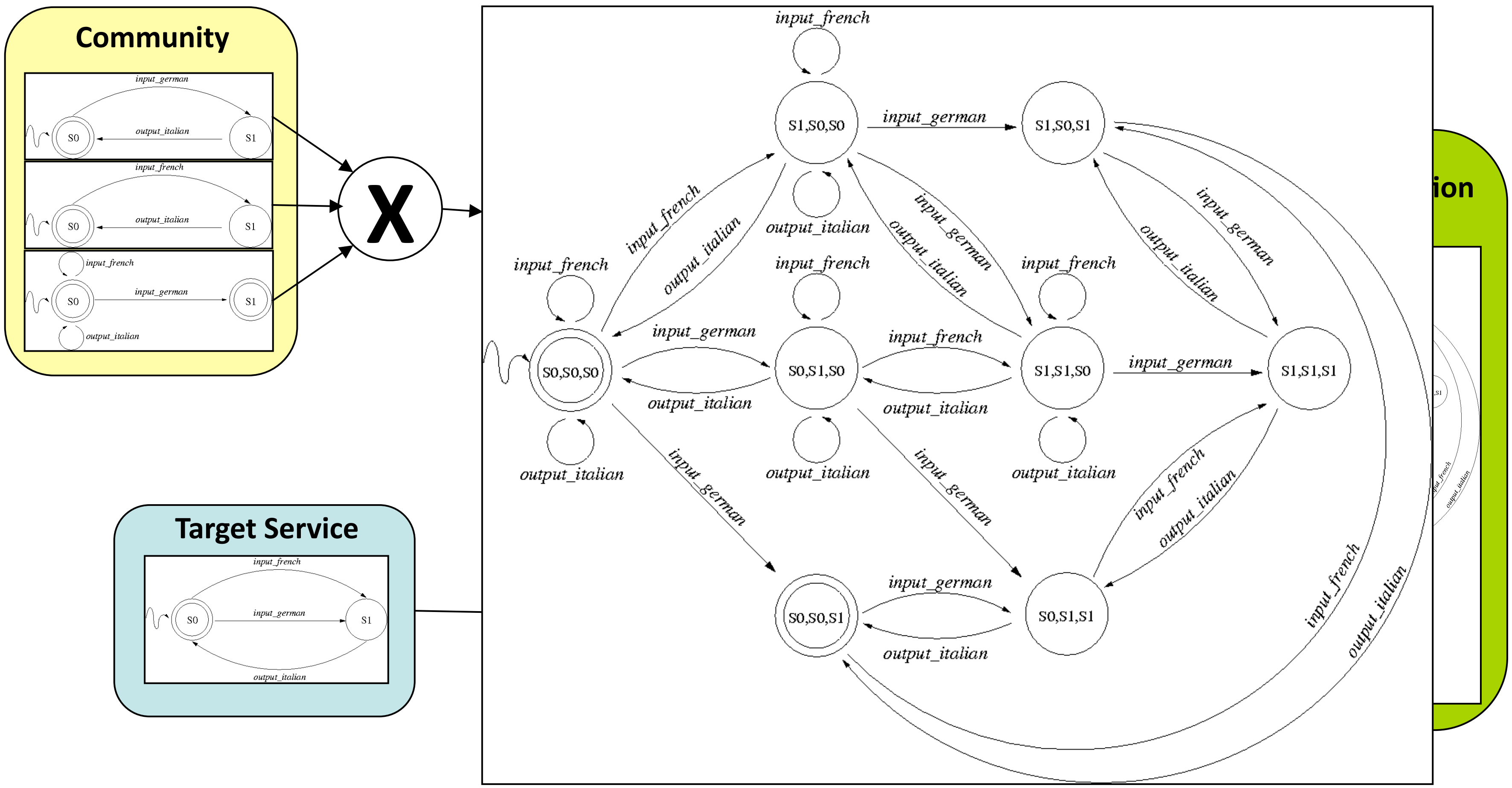
- A *Community* of *services* over a shared alphabet $\mathcal{A}$
- A (Virtual) *Goal* service over $\mathcal{A}$





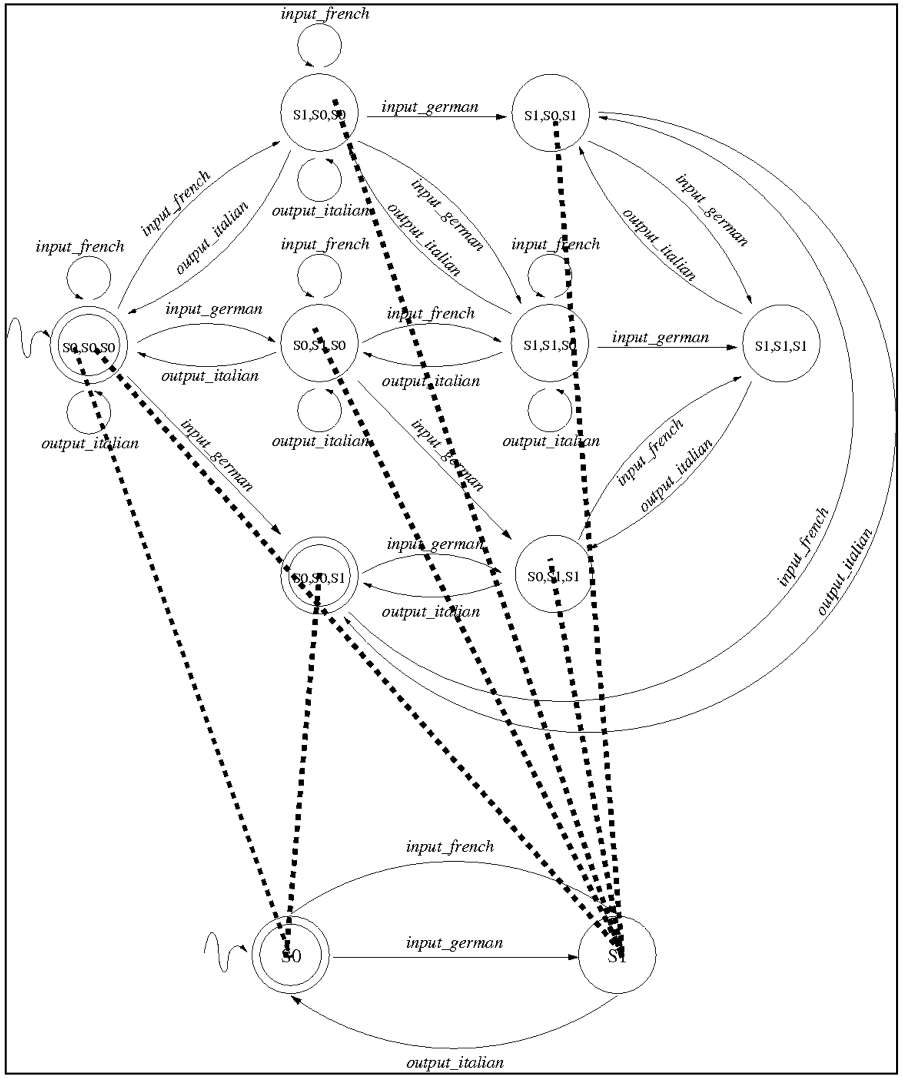
Example of composition via simulation (2)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
SAPIENZA Università di Roma



Example of composition via simulation (3)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
SAPIENZA Università di Roma



Example of composition via simulation (4)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
SAPIENZA Università di Roma