

# *Enhancing the Simulation-based Technique in Automatic Service Synthesis*

---

Ario Santoso

[[ario.santoso@stud-inf.unibz.it](mailto:ario.santoso@stud-inf.unibz.it)]

[[santoso.ario@gmail.com](mailto:santoso.ario@gmail.com)]



# Outline



- *Introduction*
  - Service
  - Service Oriented Computing
  - Service Composition
- *Related works*
- *Simulation-based technique [Pat09]*
- *Challenges*



# Service Oriented Computing

---

- *Service*
  - **Software module**, self-described, self-contained.
  - Available via **network**.
  - **Consists of** several **operations/actions**.
  - **Communicates** with the client (can be another service) through a certain **interface**.
- *Service oriented computing (SOC)*
  - Uses **services** as its building block.
  - When the **requested service is not available**, (part of) **available services** can be **composed** into a **new service** to fulfill the request.



# Service Composition

---

- *Community of available service* (**Community**)  
The set of **available services** for composition (to obtain a new service).
- **Target service**  
Client's **requested** service.
- *Service composition*
  - The process of **composing available services** into a new service (**composite service**).
  - Consists of two tasks: **Service synthesis** and **Service orchestration**.



# Service Synthesis and Orchestration

- Service **synthesis**:
  - **Input**: a **target service** and a **community of available services**.
  - **Objective**: **Construct** a **specification** which **specifies how to compose** (part of) the **available services** in order to obtain the target service.
  - Can be done automatically or manually.
  - The one who perform this task is called Service Synthesizer (**Synthesizer**).
- Service **orchestration**:
  - **Coordinate** the execution of the (component of) **available services**, and monitors the process and the data flow in order to **guarantee** the **correct realization** of the **target service based on** the **specification** produced in service synthesis.
  - The one who perform this task is called Service Orchestrator (**Orchestrator**).



# Example of Service Composition

- Machine translation
  - Available services:
    - **Italian\_German\_Translator**, It has operations:
      - *translate\_German\_To\_Italian* [Gr\_It]
      - *translate\_Italian\_To\_German* [It\_Gr]
    - **Indonesian\_German\_Translator**, It has operations:
      - *translate\_German\_To\_Indonesian* [Gr\_In]
      - *translate\_Indonesian\_To\_German* [In\_Gr]
  - Target service: ***translate\_Italian\_To\_Indonesian***
  - Realization:  
Italian\_text → [It\_Gr] → [Gr\_In] → Indonesian\_text



# Related work

---

- **AI Planning** approach [Dwu03]
- Linear logic with **theorem proving** [Rao06]
- **Semantic web-based** approach [Med03]
- **PDL-based** approach [Ber05]
- **Simulation-based** approach [Pat09]

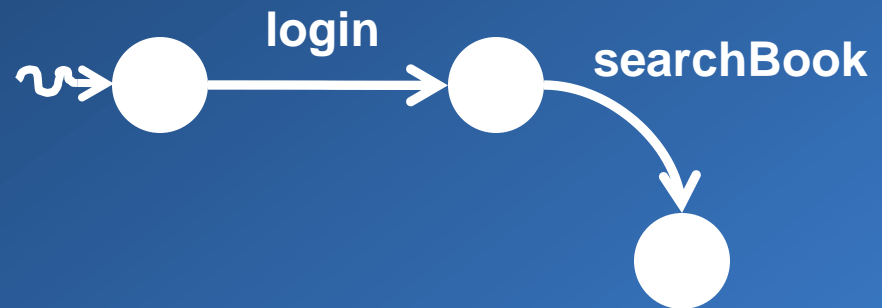
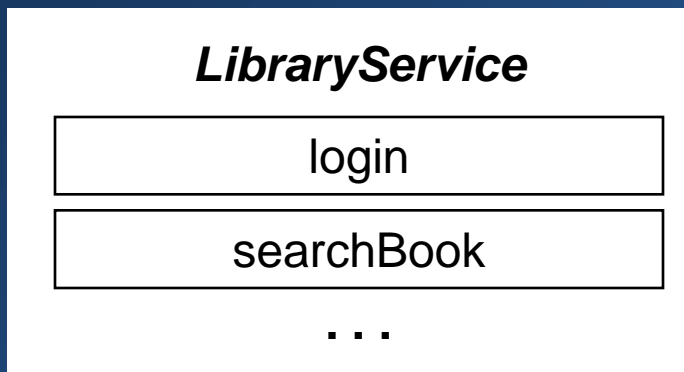
# Service Behavior

---

- Client and service **atomic interaction**
  - An activity which consists of the following steps:
    - The **service proposes a choice** of **operations** to the client.
    - The **client chooses one** of the offered **operation**.
    - The **service executes the client's choice**.
- Client and service conversation (**Conversation**)
  - **Sequence** of **atomic interactions** between clients and service
- **Behavior** of a service:
  - All of the **possible conversations**.

# Behavioral Vs Non-behavioral Approach

- **Behavioral** approach:
  - It models the **behavior of a service** (i.e., the **possible conversations**)
- **Non-behavioral** approach:
  - It doesn't express the behavior of the services
    - e.g.: **only** specifies the **input/output of each operations** inside the service



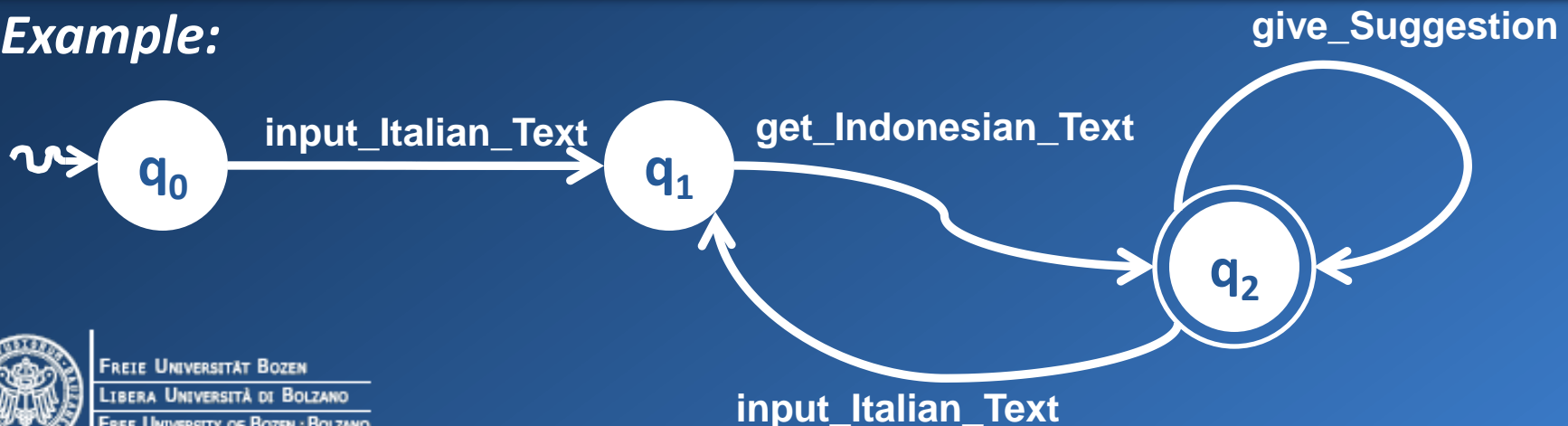
# Service as Finite Transition System (TS)

## Definition 1 (Service's Transition System):

A **service (behavior)** is finitely represented by a **finite transition system (TS)**  $S = (Q, O, q_0, \delta, F)$ , where:

- $Q$  is a finite set of **states**. (state of the service)
- $O$  is a finite **alphabet** of **operations**. (service's operations)
- $q_0$  is the **initial state**. (service's initial state)
- $\delta : Q \times O \rightarrow Q$  is the **transition** function.
- $F \subseteq Q$  is the set of **final states**. (service final state)

**Example:**



# Community TS

**Asynchronous product** of all **TS** of the services in the **service community**.

## Definition 2 (Community's Transition System):

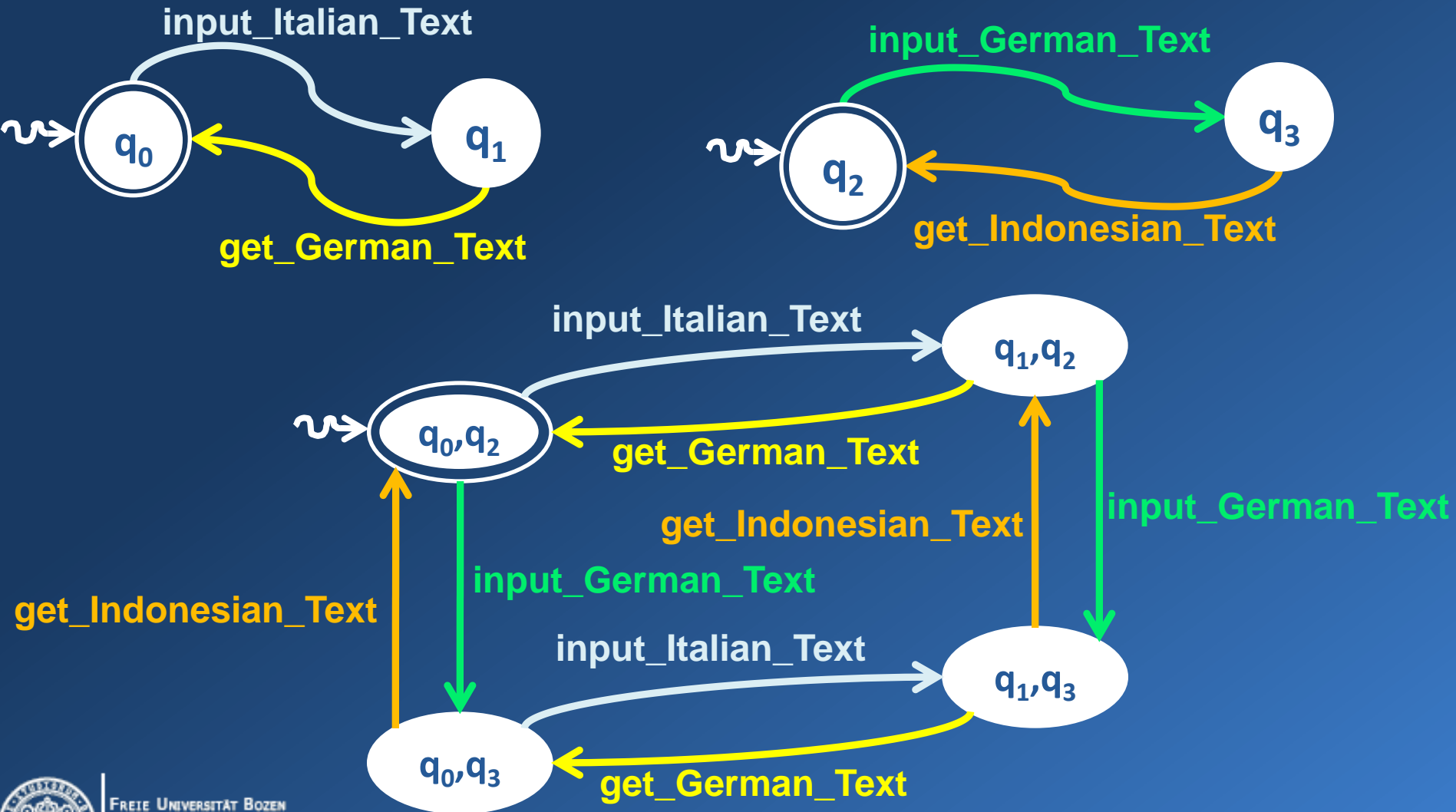
Given a community of services  $C = \{S_1, \dots, S_n\}$ , where  $S_i = (Q_i, O_i, q_{i0}, \delta_i, F_i)$  for  $i = 1, \dots, n$ , the **Community TS** of  $C$  is the transition system

$S_C = (Q_C, O, q_{C0}, \delta_C, F_C)$ , where:

- $Q_C = Q_1 \times \dots \times Q_n$  is a finite set of  $S_C$ 's **states**.
- $O$  is a **shared** finite **alphabet** of **operations**.
- $q_{C0} = (q_{10}, \dots, q_{n0})$  is the **initial state**.
- $F_C = F_1 \times \dots \times F_n$  is the set of  $S_C$ 's **final states**.
- $\delta_C \subseteq Q_C \times O \times Q_C$  is the  $S_C$ 's **transition** relation, where  $\delta_C((q_1, \dots, q_n), o, (q_1', \dots, q_n'))$  iff there exists  $k \in \{1, \dots, n\}$  such that  $\delta_C(q_k, o, q_k')$  in  $S_k$  and for each  $i \neq k$ ,  $q_i' = q_i$

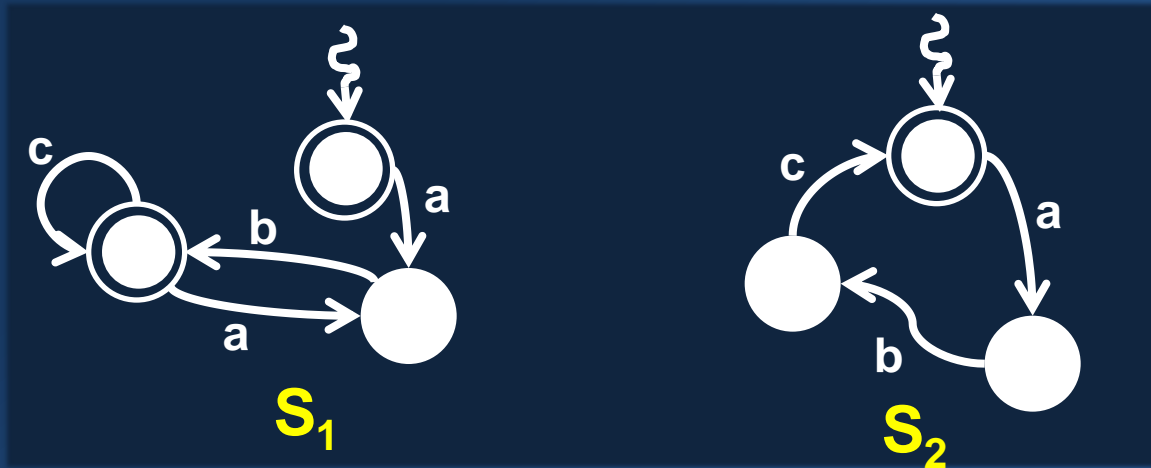


# Example of Community TS



# Intuitive Idea of Simulation

- Informally, transition system  $S_1$  **simulates**  $S_2$  if  $S_1$  shows at least the **same behavior as  $S_2$** .



- $S_1$  **simulates**  $S_2$  since **each conversation** of  $S_2$  is supported by  $S_1$ .
- But,  $S_2$  **doesn't simulate**  $S_1$  since **not each conversation** of  $S_1$  is supported by  $S_2$ .

# Simulation Relation

## Definition 3 (Simulation Relation):

Given two transition system  $S_t$  and  $S_C$ , a **simulation relation** of  $S_t$  by  $S_C$  is a relation  $R \subseteq Q_t \times Q_C$ , such that  $(q_t, q_C) \in R$  implies:

1. If  $q_t \in F_t$  then  $q_C \in F_C$
2. For each transition  $\delta_t(q_t, o, q_t')$  in  $S_t$  there exists a transition  $\delta_C(q_C, o, q_C')$  in  $S_C$  and  $R(q_t', q_C')$ .



# Service Composition as Simulation

## Definition 4:

Let  $S_t$  be the **target service TS** and  $S_C$  be the **community TS**, as above.

- **A state**  $q_t \in Q_t$  **is simulated by a state**  $q_C \in Q_C$  (or  $q_C$  simulates  $q_t$ ), **denoted**  $q_t \preceq q_C$ , iff there exists a simulation relation  $R$  of  $S_t$  by  $S_C$  such that  $R(q_t, q_C)$ .
- $S_t$  **is simulated by**  $S_C$  (or  $S_C$  **simulates**  $S_t$ ) iff  $q_{t0} \preceq q_{C0}$ , where  $q_{t0}$  and  $q_{C0}$  is the initial states of the target and the community TS respectively.

## Theorem 1: [Pat09]

A **composition** of the target service  $S_t$  for community  $C = \{S_1, \dots, S_n\}$  **exists** iff  $S_t$  is **simulated** by  $S_C$ .



# Open Challenge: Cost Modeling

- **Consider** the **cost factor** during the **service synthesis**.
- Model the **cost** for each **operation execution**.
- Example scenario:
  - There are **several possible composition**.
  - **Choose** the **one** which has the **best cost**.
- Plan:
  - Using a **weighted device** for **embedding cost information** into the **service** formal representation.
    - E.g. **weighted transition system**.

# Semiring

- A semiring  $\hat{S}$  is a structure  $(\hat{C}, +, \cdot, 0, 1)$  where:
  - $\hat{C}$  is a **non-empty set**.
  - $+$  is a binary, **associative, commutative** operation on  $\hat{C}$ .
  - $\cdot$  is a binary, **associative** operation on  $\hat{C}$ .
  - $0$  is **neutral** w.r.t.  $+$ , i.e.  $c + 0 = c$ , for all  $c \in \hat{C}$ .
  - $1$  is **neutral** w.r.t.  $\cdot$ , i.e.  $c \cdot 1 = c = 1 \cdot c$ , for all  $c \in \hat{C}$ .
  - $\cdot$  **distributes** over  $+$  from both sides.
  - $0$  **annihilating** w.r.t.  $\cdot$ , i.e.  $c \cdot 0 = 0 = 0 \cdot c$ , for all  $c \in \hat{C}$ .



# Service as Weighted TS

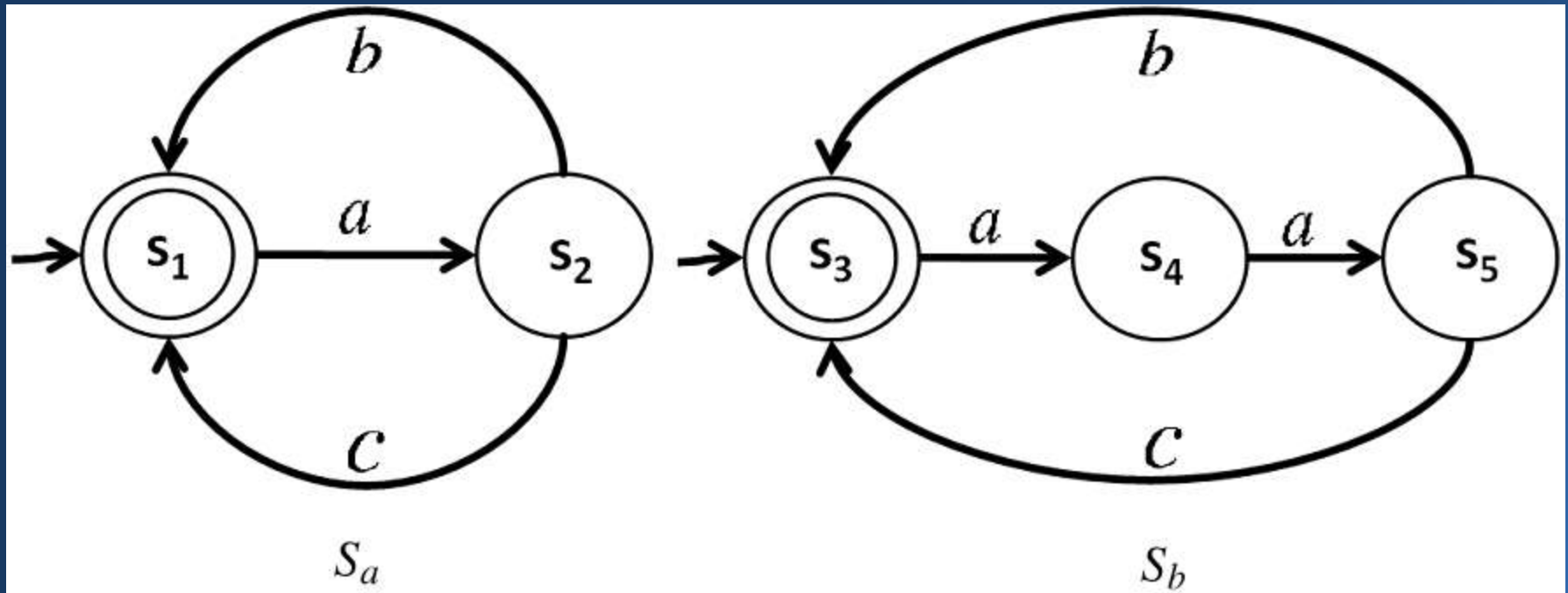
- Let  $\hat{S} = (\hat{C}, +, \cdot, 0, 1)$  is a semiring. A service (behavior) is represented by a **weighted finite deterministic transition system**  $S = (Q, O, q_0, F, \nu)$ , where:
  - $Q$  is the finite set of **states**;
  - $O$  is the finite alphabet of **operations**;
  - $q_0$  is the **initial state**;
  - $F \subseteq Q$  is the set of **final states**;
  - $\nu = Q \times O \times Q \rightarrow \hat{C}$  assigns a **weight** to each **transition**;

# Community Weighted TS

- Let  $\hat{S} = (\hat{C}, +, \cdot, 0, 1)$  is a semiring. Given a community  $C = \{S_1, \dots, S_n\}$ , where  $S_i = (Q_i, O_i, q_{i0}, F_i, v_i)$ , for  $i = 1, \dots, n$ . The Weighted Community TS of C is the transition system  $S_C = (Q_C, O, q_{C0}, F_C, v_C)$ , where:
  - $Q_C = Q_1 \times \dots \times Q_n$  is a finite set of  $S_C$ 's **states**.
  - $O$  is a **shared** finite **alphabet** of **operations**.
  - $q_{C0} = (q_{10}, \dots, q_{n0})$  is the **initial state**.
  - $F_C = F_1 \times \dots \times F_n$  is the set of  $S_C$ 's **final states**.
  - $v_C \subseteq Q_C \times O \times Q_C \rightarrow \hat{C}$  assigns a weight to each transition. Where  $v_C((q_1, \dots, q_n), o, (q_1', \dots, q_n')) = v_C(q_k, o, q_k')$  iff there exists  $k \in \{1, \dots, n\}$  such that  $v_k(q_k, o, q_k')$  in  $S_k$  and for each  $i \neq k, q_i' = q_i$



# Open Challenge: Reachability



Two behavioral services [pat09]

- $S_a$  is **not simulated** by  $S_b$ , but if the **orchestrator** are **allowed** to perform a **silent operation execution**, then  $S_a$  is **simulated** by  $S_b$

# Summary

---

- *Service composition*
  - Service Synthesis
  - Service Orchestration
- *Simulation-based framework*
  - Checking the existence of composition by checking the existence of simulation relation
- *Challenge:*
  - Cost modeling
    - Plan: weighted TS
  - Reachability

# References

---

- [Ber05] D. Berardi. *Automatic Service Composition. Models, Techniques and Tools*. PhD thesis, Università Degli Studi di Roma“LaSapienza”,2005.
- [Dwu03] D. Wu, B. Parsia, E. Sirin, J. Hendler, and D. Nau. *Automating daml-s web services composition using shop2*. In Proc. of ISWC, 2003.
- [Pat09] F. Patrizi. *Simulation-based Techniques for Automated Service Composition*. PhD thesis, Sapienza-Università di Roma, 2009.
- [Rao06] J. Rao, P. Kungas, and M. Matskin. *Composition of semantic web services using linear logic theorem proving*. Information Systems, 31(4-5):340–360, 2006. The Semantic Web and Web Services.
- [Med03] B. Medjahed, A. Bouguettaya, and A. K. Elmagarmid. *Composing web services on the semantic web*. VLDB journal, 2003.

*terima kasih    grazie    cảm ơn*  
*Obrigado    Hvala*  
*köszönöm !תודה    děkuji*  
*mahalo    고맙습니다*  
*thank you*  
*merci    谢谢    danke*  
*Ευχαριστώ    شكرا*  
*どうもありがとう    gracias*

