QoS-Aware Service Selection in Geographically Distributed Clouds

Xin Li,

Jie Wu,

Sanglu Lu

Nanjing University

Temple University

Outline

- Background and Motivation
- Problem Statement
- Our Approach
- Evaluation

Background

Cloud Service

- More and more services can be accessible as the growing of cloud computing.
  - All over the world
- There are many services with equivalent function but various quality, e.g. execution time.
- Service composition is an effective way to utilize the plenitude of services.

Motivation

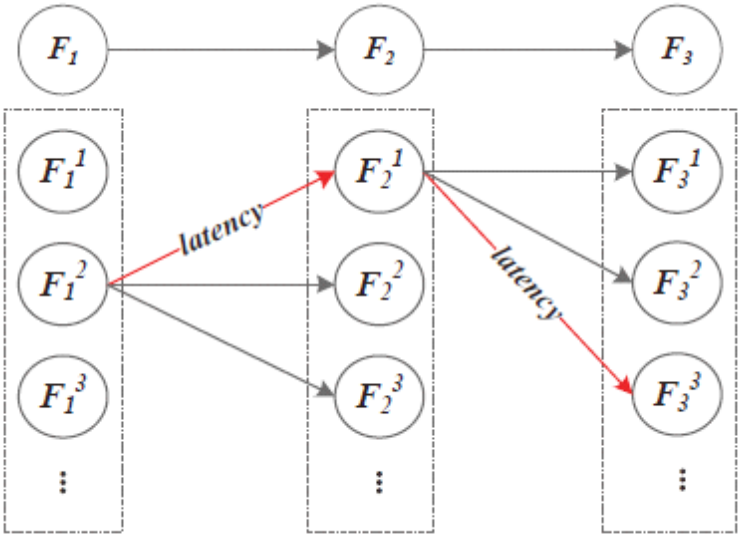
An opportunity

- This vision provides an new opportunity.
- Satisfy the diverse demands of users via service composition based on the cloud services.
- Provide the best QoS for the users.
  - Minimal latency

The problem

- How to select the optimal service set when many functional equivalent servers exist?
- The total number of service instance is limited due to the constraint of cost.

Motivation Example

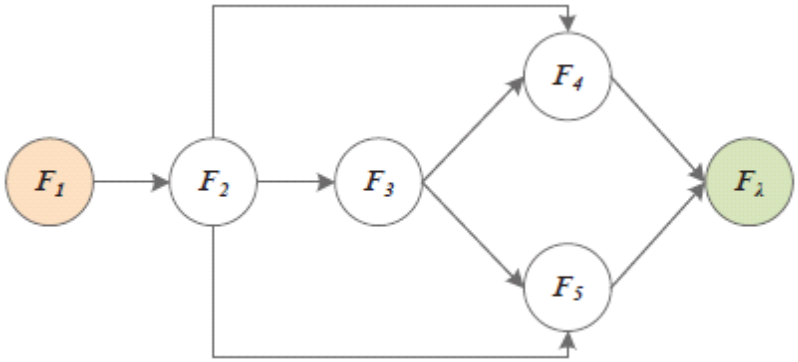


An example of service composition with 3 basic simple services.

Which services should be selected as to composition components?

Preliminaries

❖ Function Graph



❖ Initialization (abstract level & concrete level)

- Select service instances

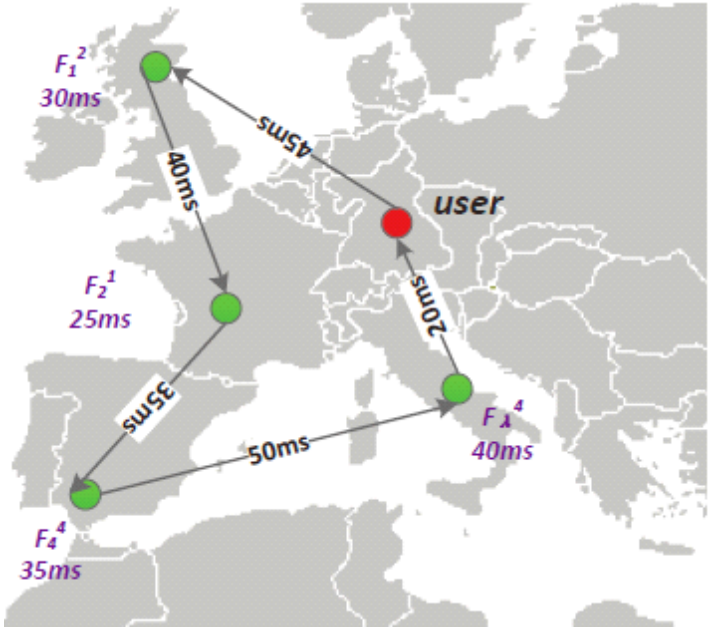
  - *for the abstract functional component .*

Preliminaries (Cont.)

Network Model

- NCS (network coordinate system)

Data Flow



Data Flow

- Total response time for user and data flow .

- Delay of a data flow .

Total Quality

- Latency for some user .

- Quality for the selected service set
- Factor to measure the total QoS.

Problem Statement

Problem Formalization

$$\textit{min.} \quad \frac{1}{\mu}\sum_{u=1}^{\mu} R(u)$$

$$\textit{s.t.} \quad \sum_{i=1}^{\lambda} N(F_i) \leq \gamma$$

$$1 \leq N(F_i) \leq |I_i|, \forall 1 \leq i \leq \lambda$$

Hardness

- NP-hard

Algorithm – Simple Case

- **Only one service instance for each component.**
- **Basic idea**
  - Select the instance for initial and terminal component.
  - Shortest path

Algorithm 1 Selection Algorithm

Input: the user set U, service instance set I, functional graph $FG =< F, E, \lambda, K >$

Output: service instance set $\mathcal{S}$, where $|S| = \lambda, \forall r, t \in S, \pi^{'}(r) \neq \pi^{'}(t)$.

1: $\mathcal{S} \leftarrow \emptyset$
2: $\pi(F_1) = facilityLoc(U, I_1)$
3: $\pi(F_\lambda) = facilityLoc(U, I_\lambda)$
4: **for** $k = 1; k \leq K; k = k + 1$ **do**
5: $\quad \mathcal{S} = \mathcal{S} \cup shortestPath(k, \pi(F_1), \pi(F_\lambda))$
6: $\mathcal{S} = combine(\mathcal{S})$
7: **return** $\mathcal{S}$

Algorithm – General Case

- **There may be multiple instances for each component, but the total number of instances in limited.**
- **Basic idea**
  - Voting: each user declares her preference for the service selection.

Algorithm 3 *voting(u,k)*

Input: service instance set I, user u, and functional path P_k.
Output: service instance set $\mathcal{S}$

1: $\mathcal{S} \leftarrow shortestPath(I, u, k)$
2: **for** $\forall s \in \mathcal{S}$ **do**
3: $\quad s.score \leftarrow s.score + \wp(P_k)$

General Case (Cont.)

Basic idea

- Voting
- Selection: sort the instances according to their scores, which is the results of voting.

Algorithm 2 Voting Algorithm

Input: the user set U, service instance set I, functional graph $FG =< F, E, \lambda, K >$, instance number limitation γ

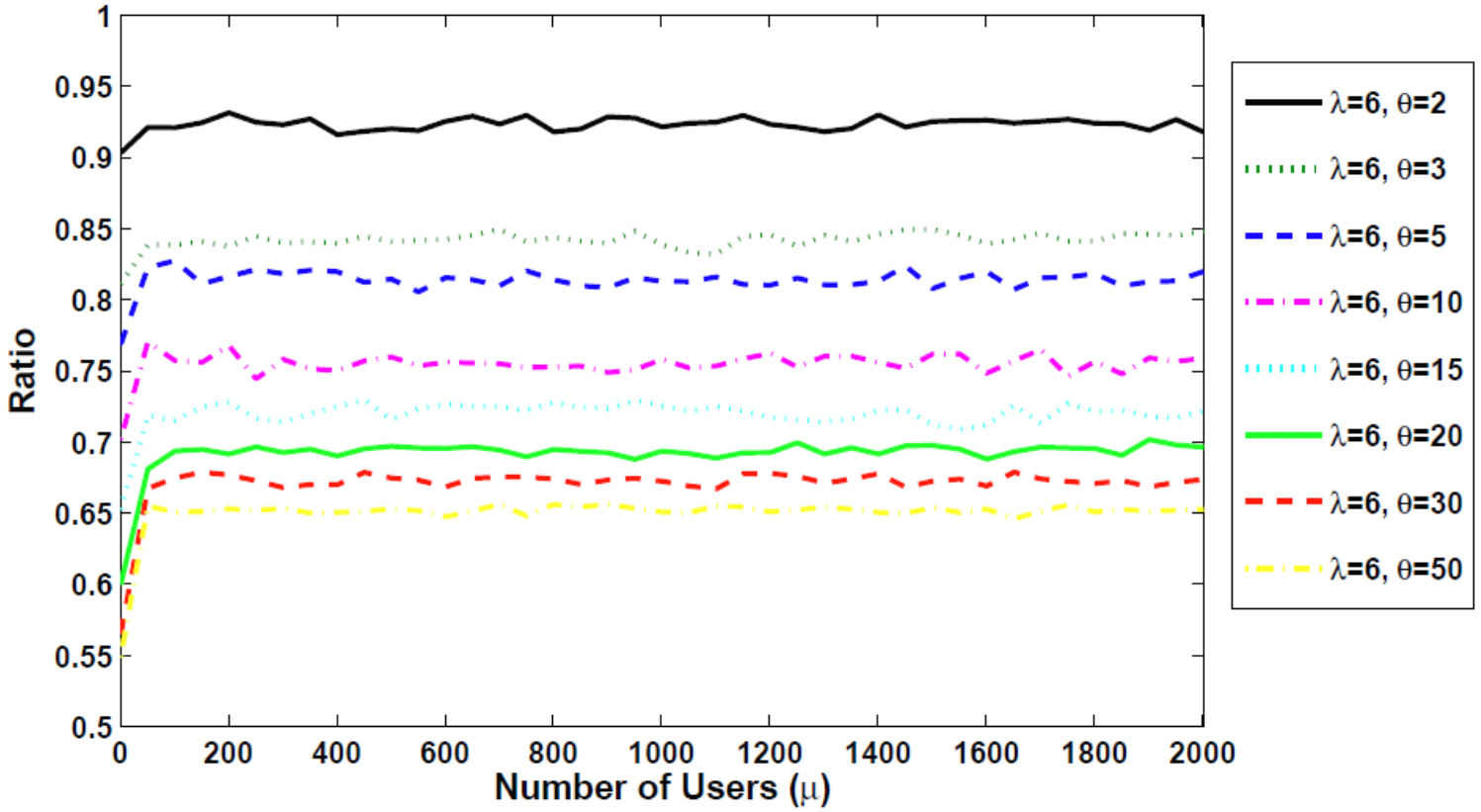
Output: service instance set $\mathcal{S}$, where $\lambda \leq |S| \leq \gamma, N(F_i) \geq 1$.

1: **for** $u = 1; u \leq \mu; u = u + 1$ **do**
2: **for** $k = 1; k \leq K; k = k + 1$ **do**
3: $voting(u, k)$
4: **for** $i = 1; i \leq \lambda; i = i + 1$ **do**
5: $I_i \leftarrow rank(I_i)$
6: $\mathcal{S} = \mathcal{S} \cup I_i.element(0)$
7: $I \leftarrow rank(I - \mathcal{S})$
8: $\mathcal{S} = \mathcal{S} \cup I.top(\gamma - \lambda)$
9: **return** $\mathcal{S}$

Evaluation – Simple Case

- Evaluation results for simple case

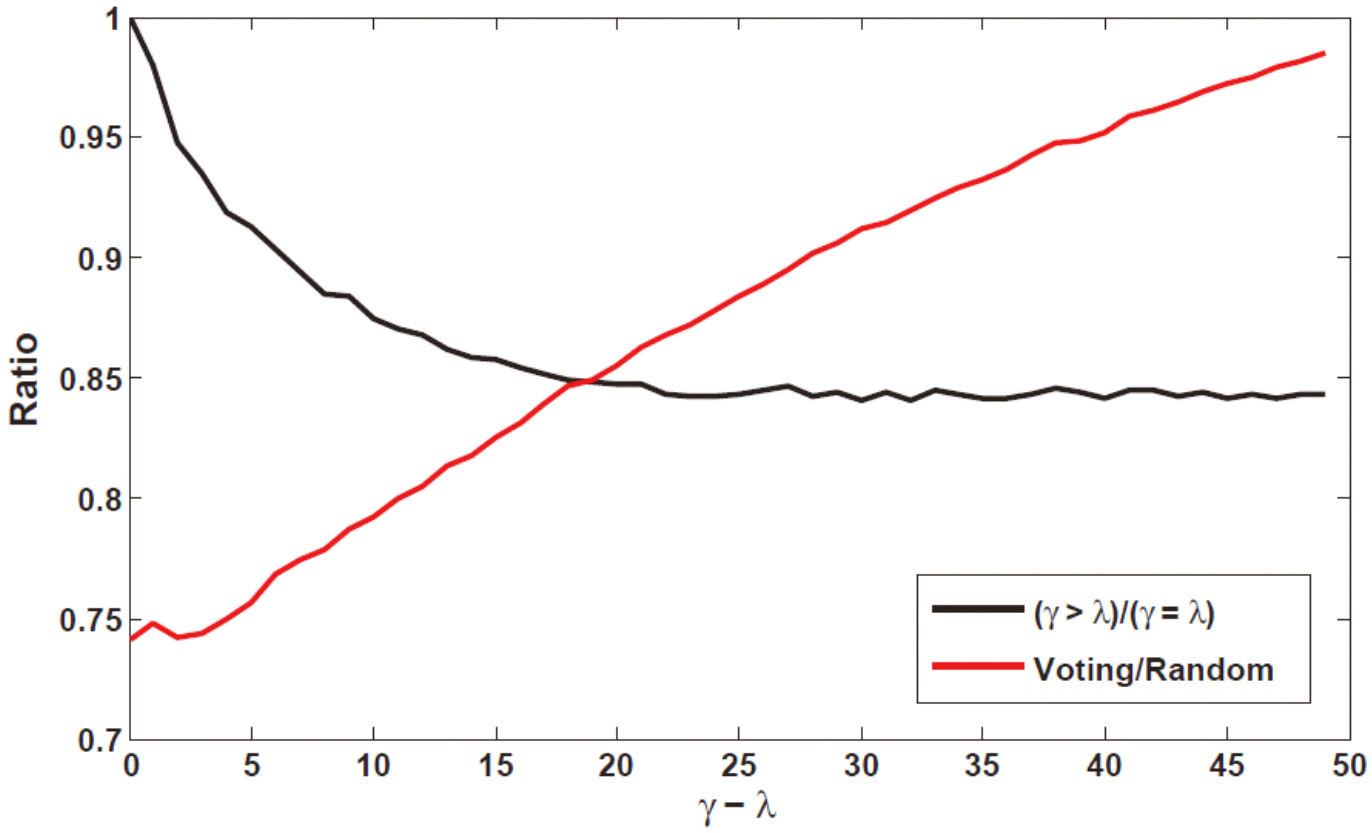
- Impact of



Evaluation – General Case

- Evaluation results for simple case

- Impact of



Thanks!

Q&A