# SPSE: A Flexible QoS-based Service Scheduling Algorithm for Service-Oriented Grid

Laiping Zhao, Yizhi Ren, Mingchu Li, Kouichi Sakurai
Information Technology and Security Laboratory
Kyushu University

This paper is partly supported by the Grant of Graduate School of ISEE, Kyushu University for Supporting Students' Overseas Traveling.
The first author of this research is supported by the governmental scholarship from China Scholarship Council.

# Outline

- 1. Introduction
- 2. QoS-based scheduling algorithm
  - 2.1 SPSE description
  - 2.2 Filter operation
  - 2.3 Rank operation
  - 2.4 Update the preferences
- 3. Analysis & Experiments
- 4. Conclusion and future works

# 1. Introduction

Motivation:

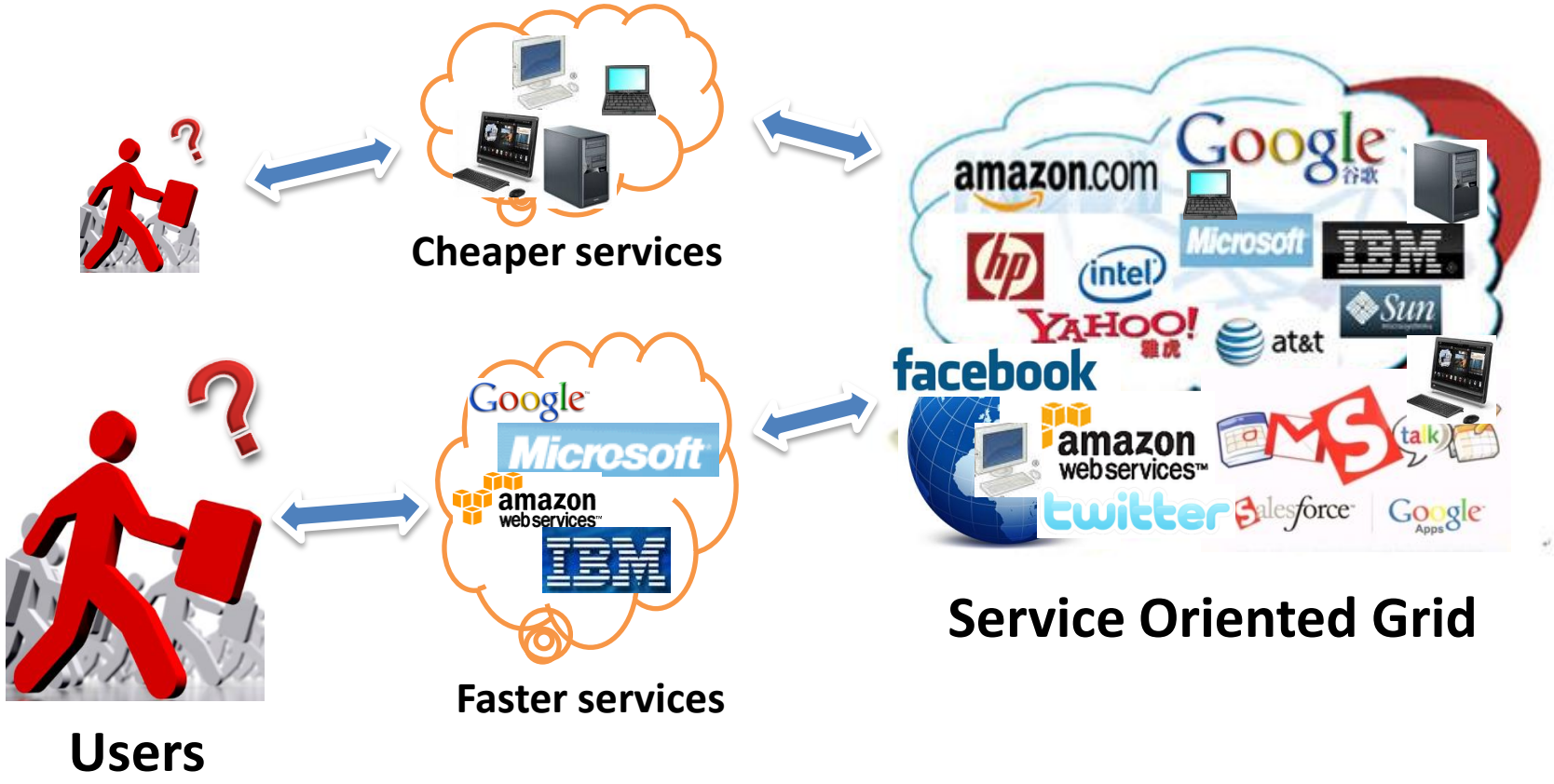

Which service is the most appropriate?

# 1. Introduction

- Problem:
  - Multi-objective supported?
    - Price? Time? Reliability? Trust?...
  - User personalization?
    - Different people have different preferences.
- We seek a *QoS-based service schedule algorithm*, which supports multi-objective and user personalization.

# 2. SPSE algorithm

- Service provider search engine (SPSE)



# 2. SPSE algorithm

- Job model:

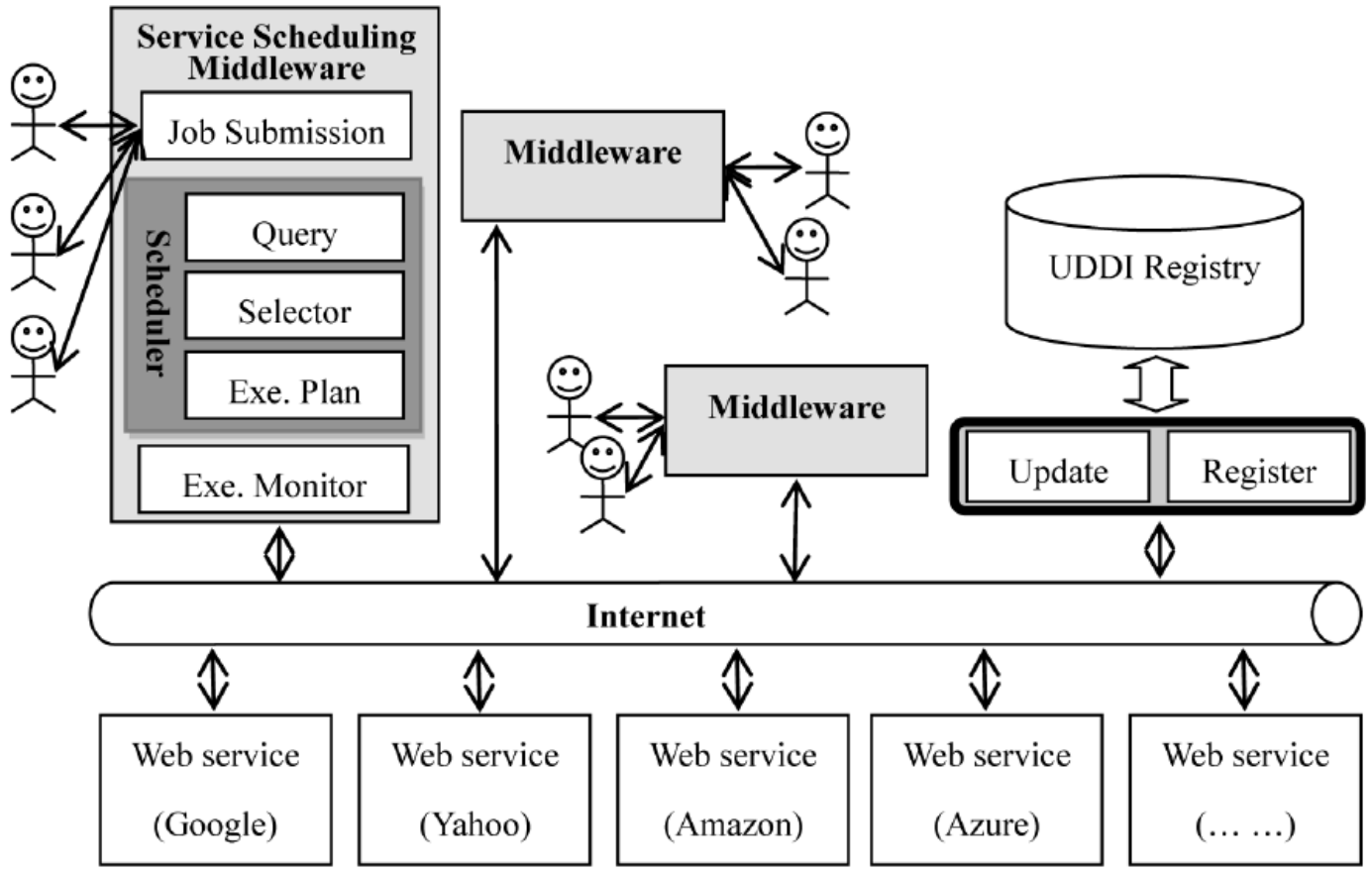
$$job = \{user\_id, job\_id, instructions, service\_type\}$$

- Service model:

$$resource = \{resource\_id, service\_type, cpu, price, trust, \ldots, (other\ criteria)\}$$

  - *cpu*, *price*, *trust*, *reliability* are the criteria.
- Service-oriented Grid environment:
  - There are many different kinds of services on the Internet. (The *service_type* indicates this.)
  - For each service, there are many service providers.
  - Web services from different organizations providing the same type of service come with the unified same interfaces.

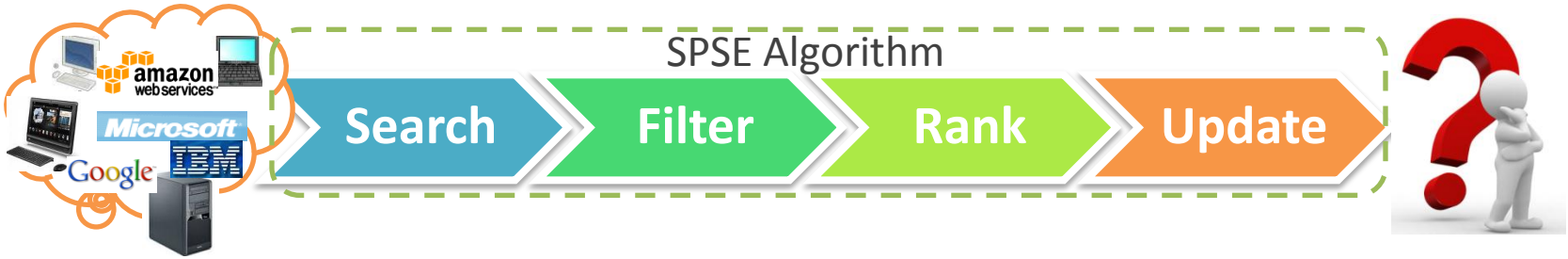
# 2. SPSE algorithm




The service scheduling middleware

# 2. SPSE algorithm

- The Service Provider Search Engine (SPSE).



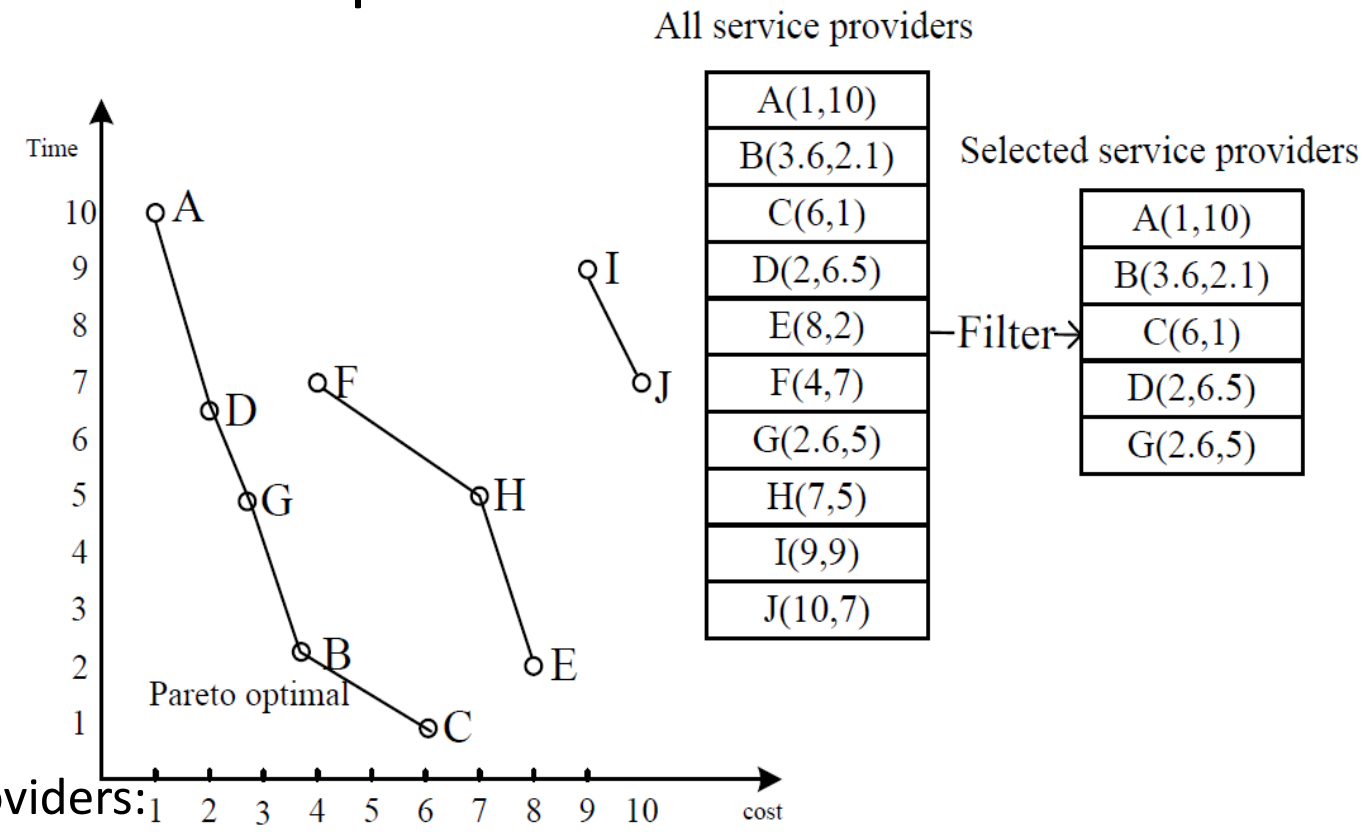
# 2. SPSE algorithm

- Operations in the algorithm:
  - *Search*:
    - Searches service providers from Grid information services (GIS) according to the *service_type*.
  - *Filter*
    - Deletes the poor service providers from all candidates.
  - *Rank*
    - Let better candidates rank higher than poorer candidates.
  - *Update*
    - User selects one candidate as the final choice, the user's choice will be used to update user's preferences.

# 2.1 Filter operation

**Goal**:

Delete the poor service providers from all candidates.




**Example 1**:

There exists 10 providers:

A,B,C,D,E,F,G,H,I,J have different performance on time and economic cost.

After the Pareto optimization based selection: A,D,G,B,C are left.

# 2.1 Filter operation

- The *time minimization service provider* (*cost minimization service provider, trust maximization service provider*) is not deleted by the filter operation.
  - (*Proof*)
- Therefore, SPSE can be used for *single objective scheduling*.

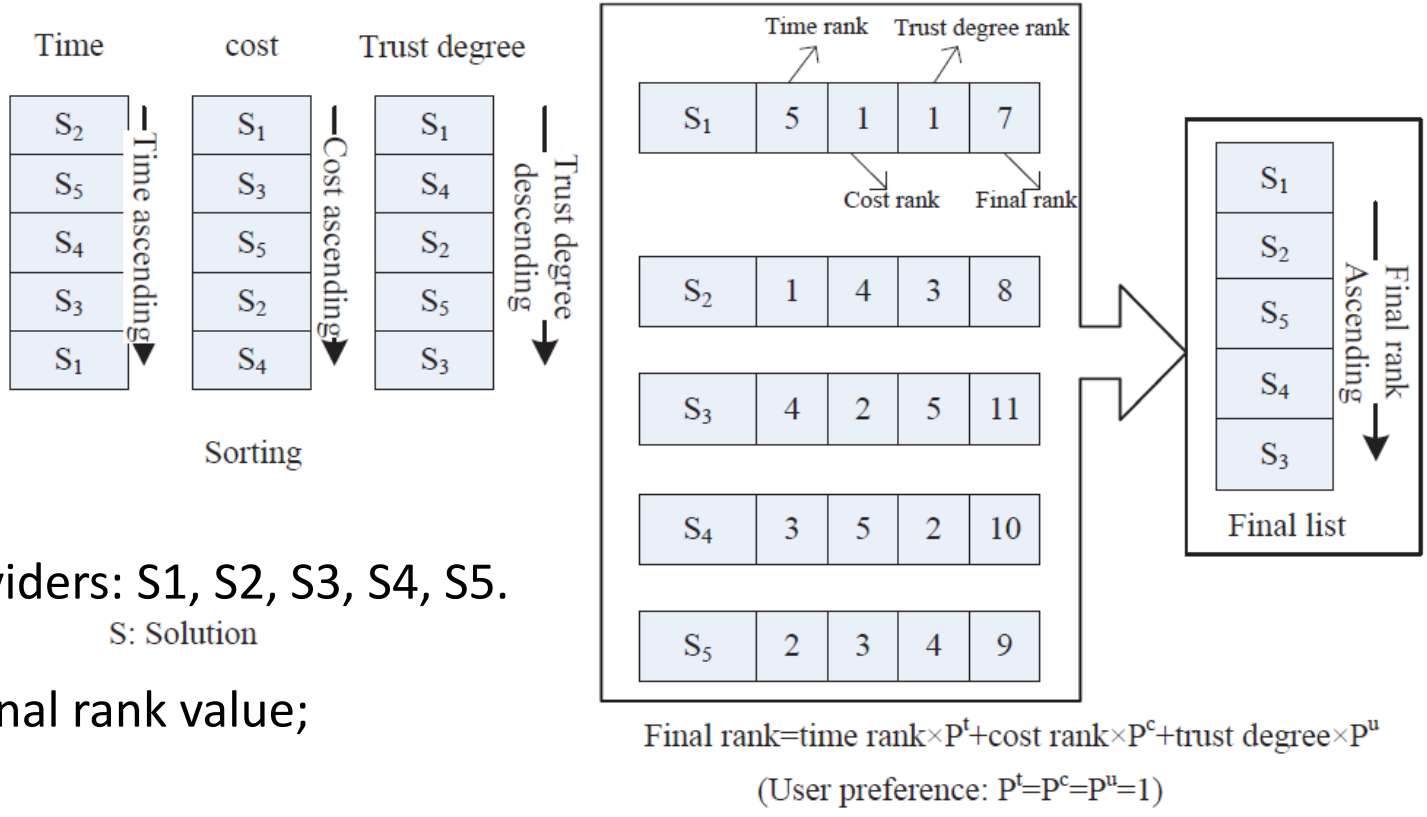
# 2.2 Rank operation

- User preference:
  - A set of parameters: $\{p_1, p_2, p_3, \ldots, p_m\}$
    - m is the number of criteria.
  - Each parameter reflects how highly user values the corresponding criteria.
- Attributes of user preferences:
  - Every user has a parameter set.
  - Initialization: $p_1^x = p_2^x = p_3^x = \ldots = p_m^x = 1$
  - The values are updated every time after one scheduling.

# 2.2 Rank operation

**Goal**:

Sort the service providers set into order.




**Example 2**:

There exists 5 providers: S1, S2, S3, S4, S5.

S: Solution

(1) Sort;

(2) Calculate the final rank value;

(3) Sort again.

Final rank=time rank×$P^t$+cost rank×$P^c$+trust degree×$P^u$

(User preference: $P^t=P^c=P^u=1$)

$$final\_rank = time\_rank \times p_1^x + cost\_rank \times p_2^x + trust\_rank \times p_3^x + (reserved \times p_i^x)$$

# 2.2 Rank operation

- Using the struct:

| Solution ID | Time rank | Cost rank | Trust rank | (Reserved) | Final rank |
|---|---|---|---|---|---|

- We get:

$$final\ rank_{s_1} = 5 \times 1 + 1 \times 1 + 1 \times 1 = 7$$

$$final\ rank_{s_2} = 1 \times 1 + 4 \times 1 + 3 \times 1 = 8$$

$$final\ rank_{s_3} = 4 \times 1 + 2 \times 1 + 5 \times 1 = 11$$

$$final\ rank_{s_4} = 3 \times 1 + 5 \times 1 + 2 \times 1 = 10$$

$$final\ rank_{s_5} = 2 \times 1 + 3 \times 1 + 4 \times 1 = 9$$

- Therefore, we get: S1 > S2 > S5 > S4 > S3

# 2.3 Update the preferences

- User select one provider from final set.
- Update the preference value on **time**:

$$p_1^{'} = p_1 \times (1 + \frac{\overline{t_{top}} - t_{user}}{\overline{t_{top}}})$$

Where $t_{user}$ is the execution time of user selected solution; $t_{top}$ is the is execution time of NO.1 solution; $p_1$ is the original preference value on time.

**Economic cost:** $$p_2^{'} = p_2 \times (1 + \frac{\overline{c_{top}} - c_{user}}{\overline{c_{top}}})$$

**Trust:** $$p_3^{'} = p_3 \times (1 + \frac{r_{user} - \overline{r_{top}}}{\overline{r_{top}}})$$

**Other criteria:** $$p_m^{'} = p_m \times (1 + \frac{\mp e_{user} \pm \overline{e_{top}}}{\overline{e_{top}}})$$

# 2.3 Update the preferences

- Update operation follows two rules:
  - If $p_i^{'} < 0$, then set $p_i^{'} = 0$
  - Reward principle:

$$\text{If}(\ p_1 < \delta\ \&\&\ \frac{\overline{t_{top}} - t_{user}}{\overline{t_{top}}} > \delta)\ \text{then set}\ p_1^{'} = 1.$$

$$\text{If}(\ p_2 < \delta\ \&\&\ \frac{\overline{c_{top}} - c_{user}}{\overline{c_{top}}} > \delta)\ \text{then set}\ p_2^{'} = 1.$$

$$\text{If}(\ p_3 < \delta\ \&\&\ \frac{r_{user} - \overline{r_{top}}}{\overline{r_{top}}} > \delta)\ \text{then set}\ p_3^{'} = 1.$$

$$\text{If}(\ p_m < \delta\ \&\&\ \frac{\mp e_{user} \pm \overline{e_{top}}}{\overline{e_{top}}} > \delta)\ \text{then set}\ p_m^{'} = 1$$

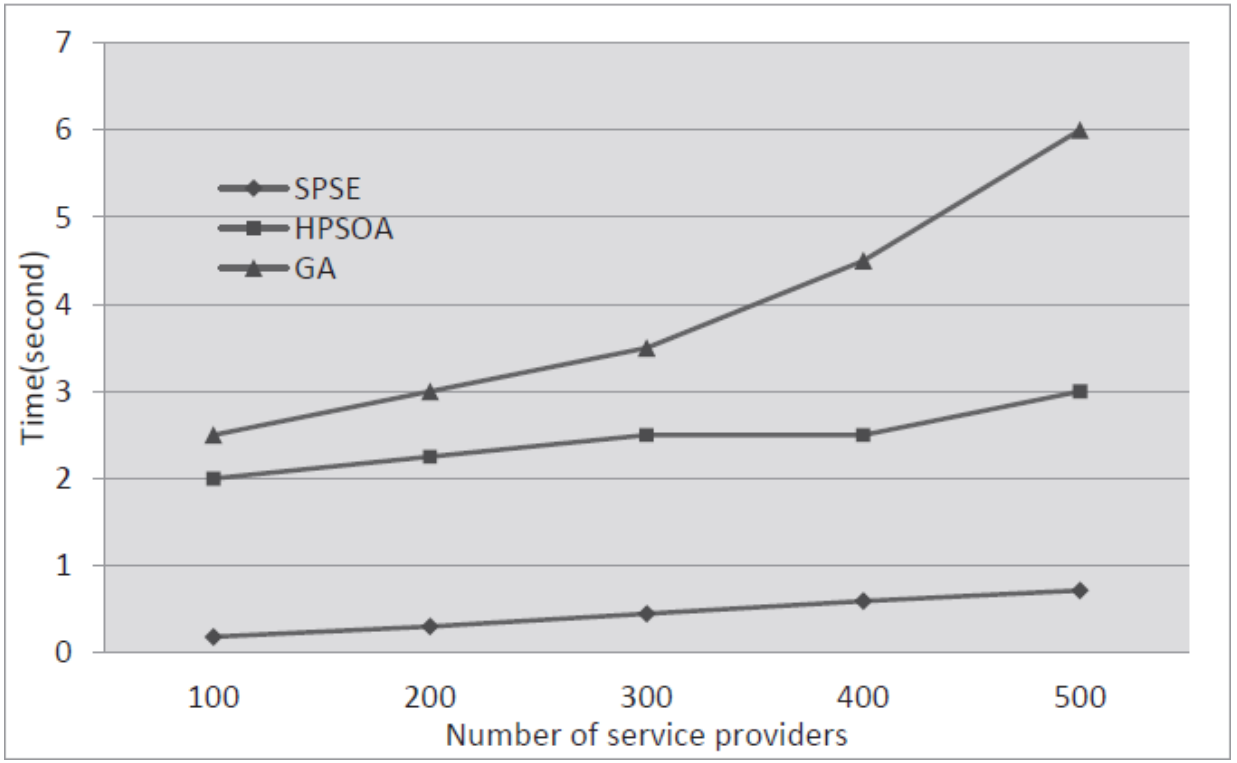
  - Where $\delta$ is a threshold value, indicating how big changes to the user's preferences.

# 3. Analysis & Experiments

**Time complexity**:

$$O(MN^2)$$

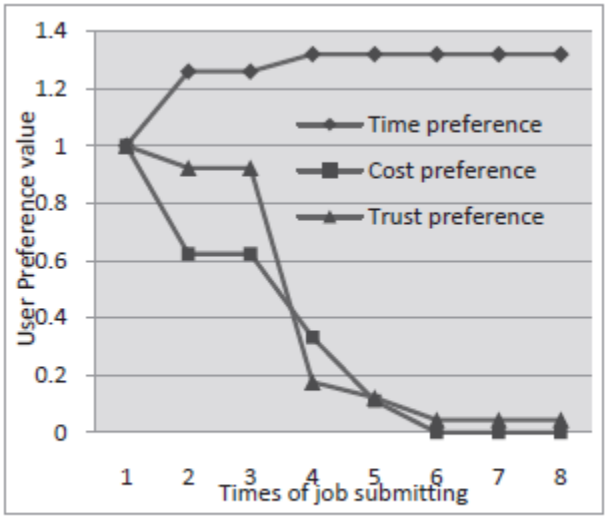
Where M is the number of criteria, N is the number of service provider candidates.



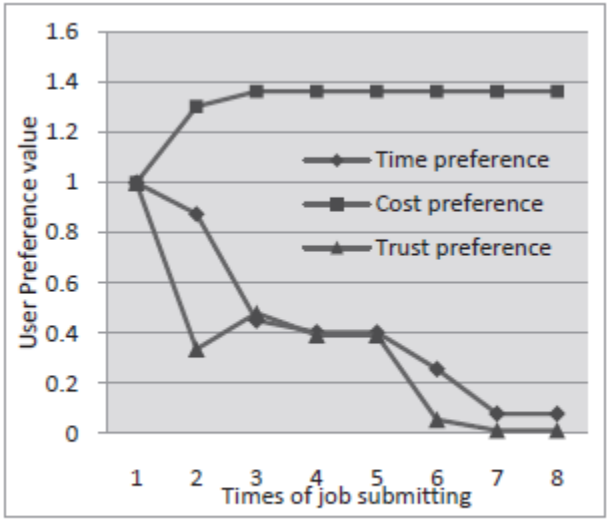

The scheduling time of SPSE with respect to different number of service providers

# 3. Analysis & Experiments

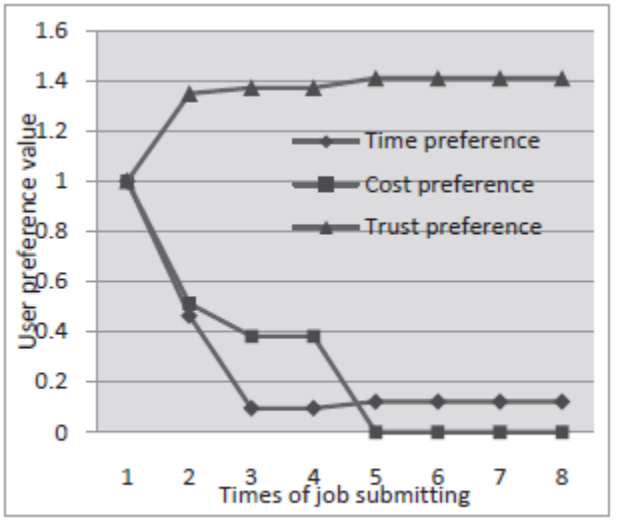
- Precision of preference values




(a)




(b)

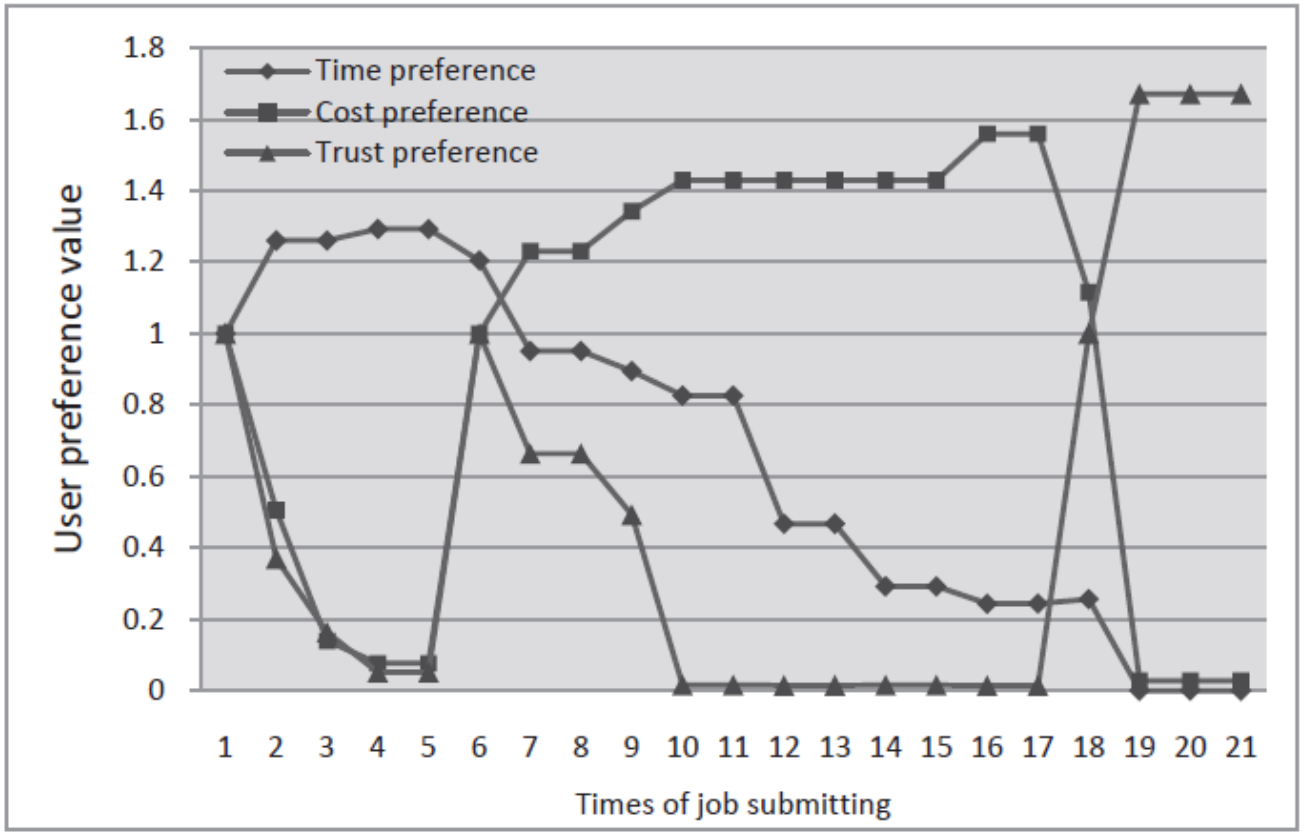



(c)

a: One person who prefers shorter time.
b: One person who prefers less economic cost.
c: One person who prefers higher provider's trust.

# 3. Analysis & Experiments

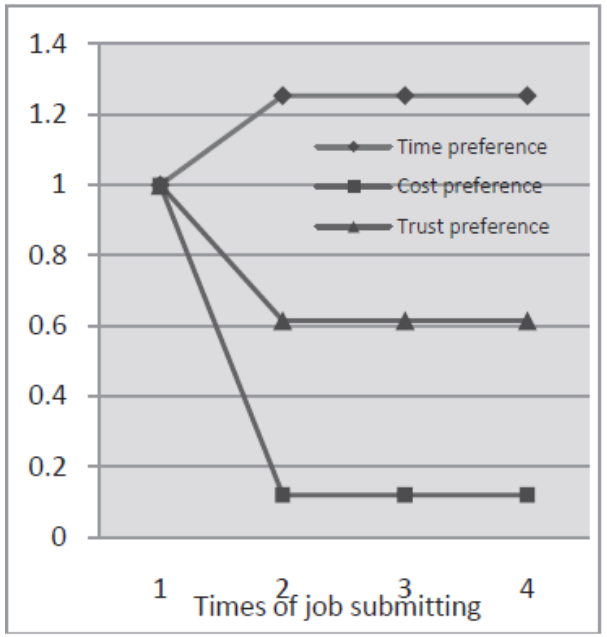
- Precision of preference values



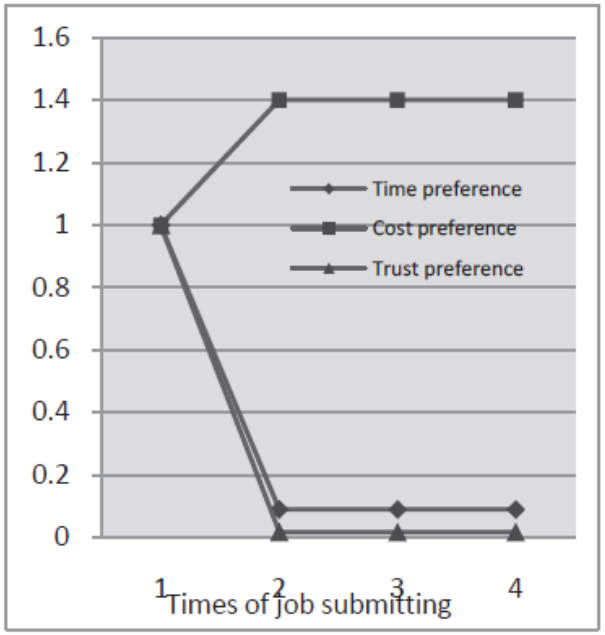

If one person changed his preference: from time to cost, from cost to trust.

# 3. Analysis & Experiments

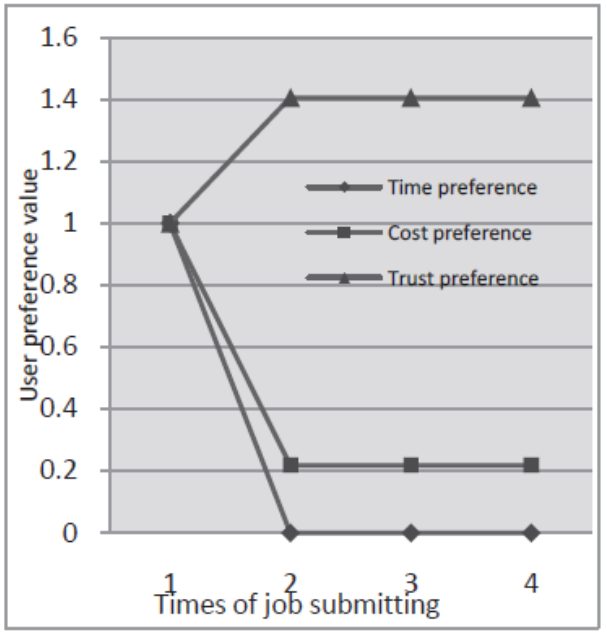
- Precision of solutions:




(a)




(b)




(c)

After the first job submission, users' preference value will be stable, if the job and Grid environment are not changed.

# 4. Conclusion and future works

- SPSE is the first algorithm on service search and scheduling:
  - Support the Multi-objective;
  - Support the User personalization.
- Experiments show that the most preferred service provider by an end-user is captured precisely.
- Future works:
  - The proposed SPSE is still simple, we ignore some issues like: fault tolerance, preemptible, which will be considered in our future work.

- Thank you for your attention.
- Questions?