QoS Management of Web Services

Zibin Zheng (Ben)

Supervisor: Prof. Michael R. Lyu

Department of Computer Science & Engineering
The Chinese University of Hong Kong

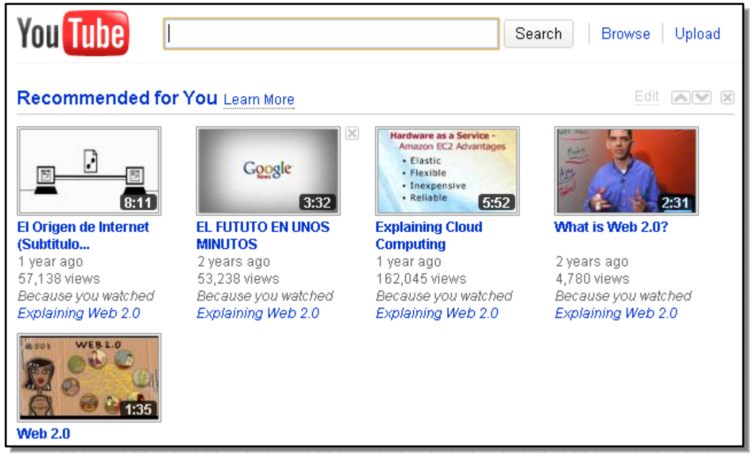
Dec. 10, 2010

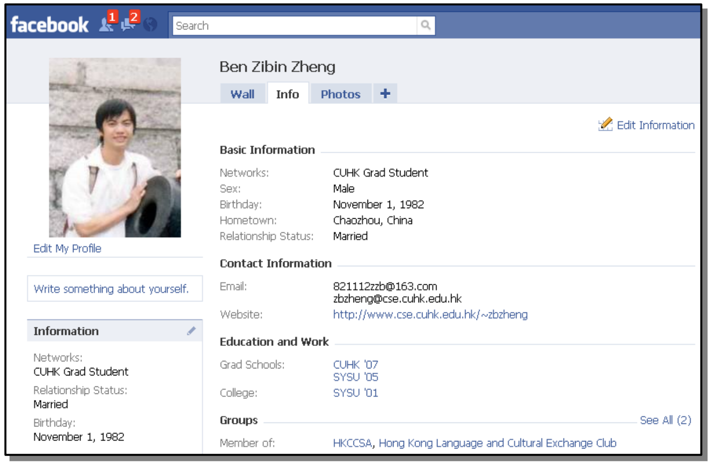
Outline

- Introduction
- Web Service QoS Evaluation
- Web Service QoS Prediction
  - Neighborhood-based method
  - Model-based method
  - Ranking-based method
- Fault-Tolerant Web Services
  - QoS-Aware Fault-tolerance for Web Services
  - QoS-Aware Selection Framework for Web Services
- Conclusion

Introduction

Web applications are becoming more and more important!



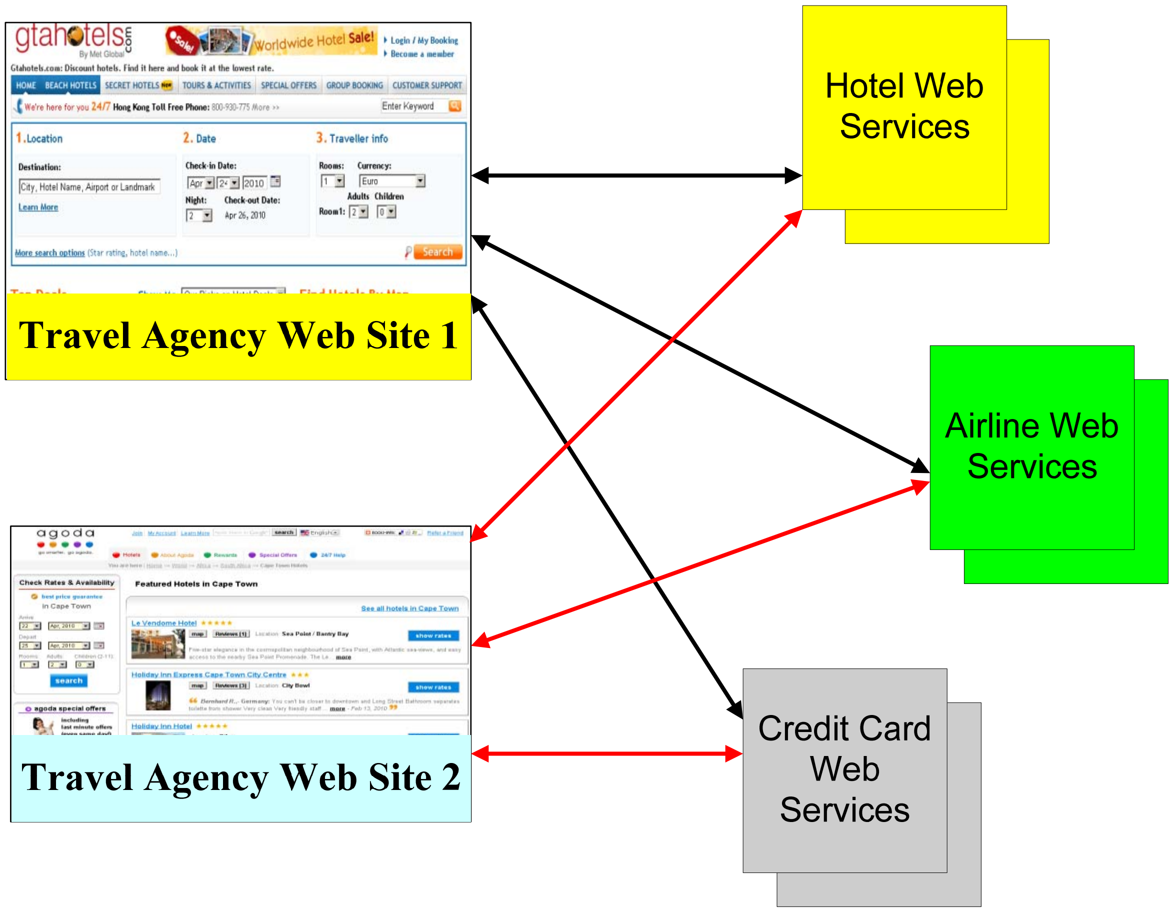


Introduction

- The age of Web 2.0
  - Web pages and Web services
- Web services (WS) are Web APIs that can be accessed over a network and executed on remote systems
  - Open standards
  - Interoperability

Introduction

- Service-oriented systems
  - Composed by distributed Web services

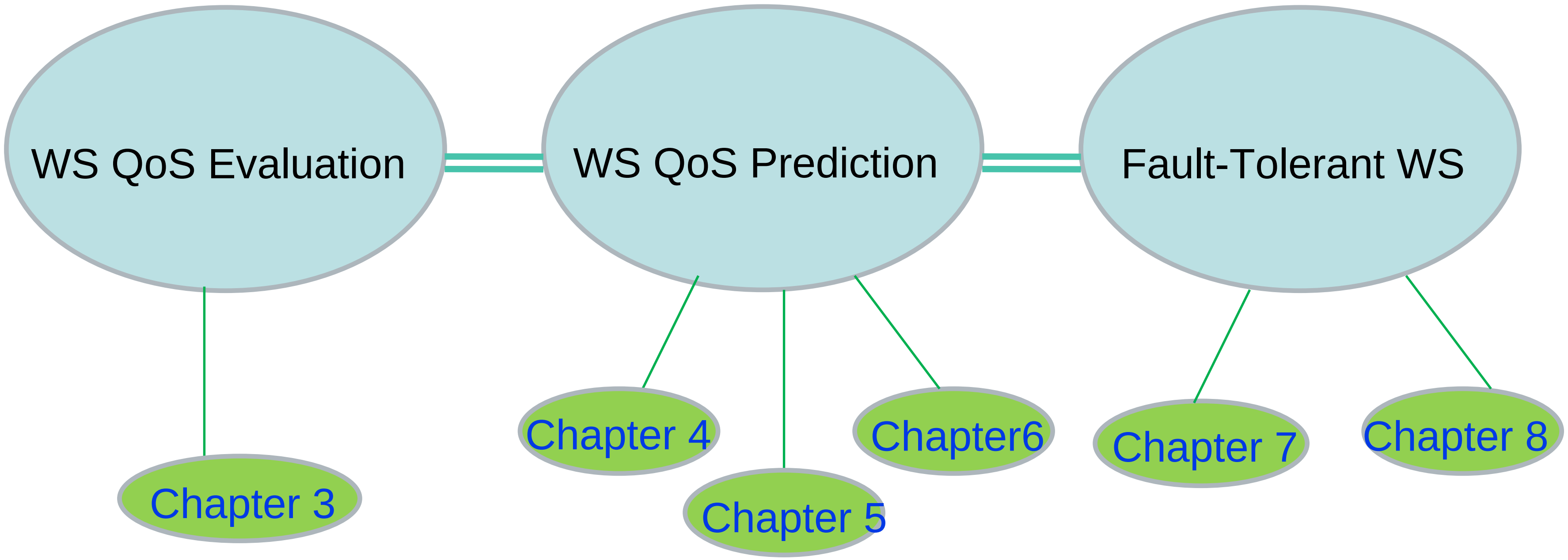


Quality-of-Service

- Quality-of-Service (QoS): Non-functional performance
  - User-independent QoS properties
    - Price, popularity
    - No need for evaluation
  - User-dependent QoS properties.
    - Failure probability, response time, throughput
    - Different users receive different performance

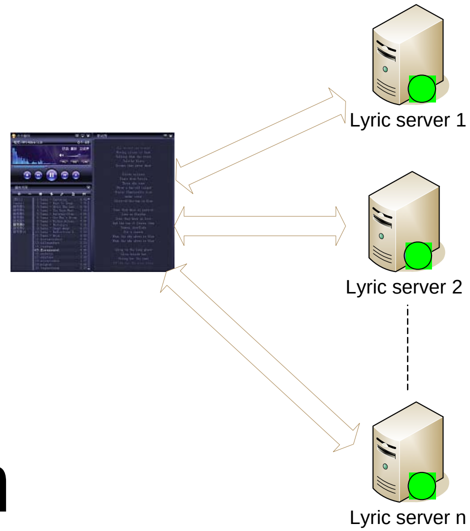
Structure of My Thesis

Title: QoS Management of Web Services



QoS-Driven Approaches

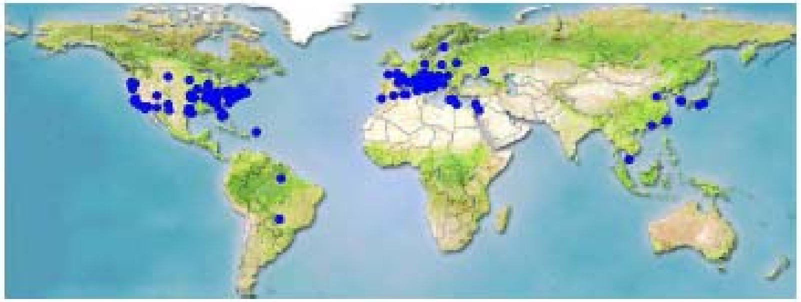
- Web service selection
- Web service composition
- Web service ranking
- Web service recommendation
- Fault-tolerant Web services
- ……………….



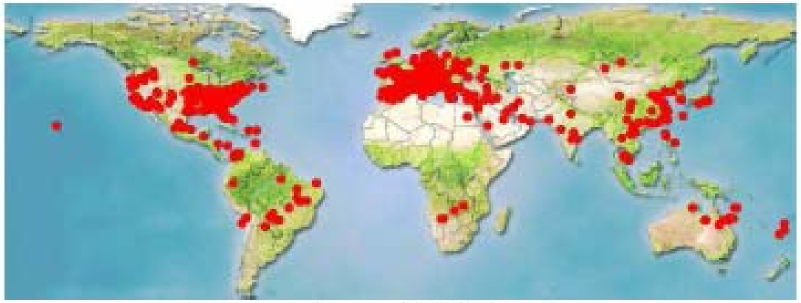
Limited real-world Web service QoS datasets for experimental studies！

Part 1: Web Service QoS Evaluation

- Two large-scale evaluations on real-world Web services
- 21,358 publicly available Web services
- 339 distributed computers
- 235,262,555 lines of Java codes



(a) Locations of Service Users



(b) Locations of Web Services

Drawbacks of Web Service Evaluation

- Difficult to conduct real-world Web service evaluations
  - Web service invocations may be charged
  - Too many Web service candidates
  - Time-consuming
  - Resource-consuming
  - Require professional knowledge

Part 2: Web Service QoS Prediction

- **Target**: Predict Web service QoS values for a user without invoking these Web services
- **Idea**: Take advantages of the social wisdom of service users
  - The past Web service usage experiences of other service users.
- Propose three QoS prediction approaches
  - Neighborhood-based
  - Model-based
  - Ranking-based

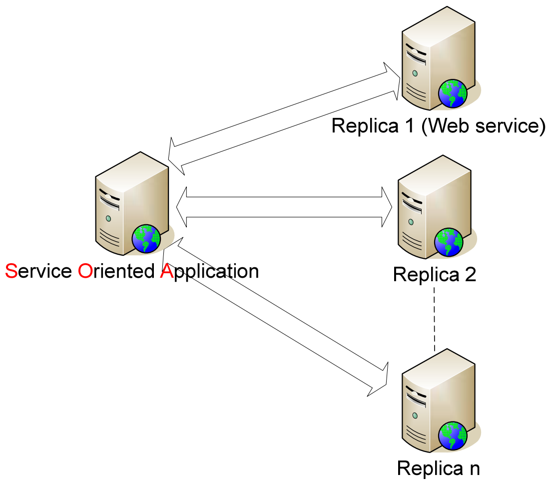
Web Service QoS Values

- QoS values of Web services can be obtained by:
  - Web service QoS Evaluation
  - Web service QoS Prediction

How to use these QoS values?

Part 3: Fault-Tolerant Web Services

- Building highly reliable service-oriented systems is a challenging task
  - Remote Web services may become unavailable
  - Remote Web services may contain faults
  - Internet environment is unpredictable



- Two fault-tolerance approaches for Web services using Web service QoS values.

Contents

1. Evaluation

- Chapter 3: QoS Evaluation of WS

2. Prediction

- Chapter 4: Neighborhood-based QoS Prediction of WS
- Chapter 5: Model-based QoS Prediction of WS
- Chapter 6: Ranking-based QoS Prediction of WS

3. Fault-Tolerant WS

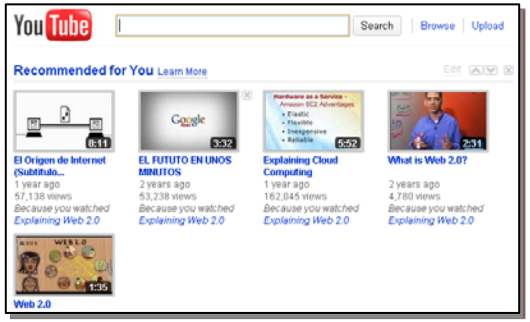
- Chapter 7: QoS-Aware Fault Tolerance for WS
- Chapter 8: QoS-Aware Selection Framework for WS

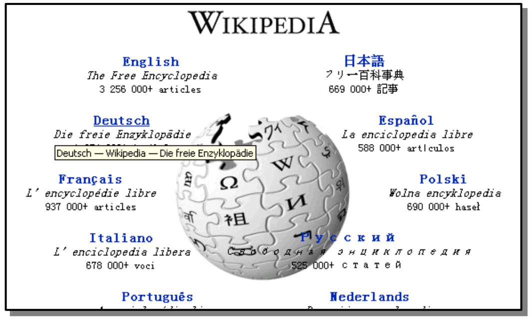
Part 1 (Ch. 3): QoS Evaluation of Web Services

Background

- (Al-Masri et al., 2008)
  - Released a Web service QoS dataset
  - 1 user and 2570 Web services
  - Best Student Paper Award Nomination at WWW2008.
- Different users observe quite different QoS of the same Web service.
- Web service evaluation from distributed locations.
  - Control distributed computers
  - Time consuming and resource consuming
- **Our contributions:**
  - A user-collaborative framework for WS evaluation
  - Large-scale distributed evaluations on real-world Web services
  - Release research datasets

User-Collaboration

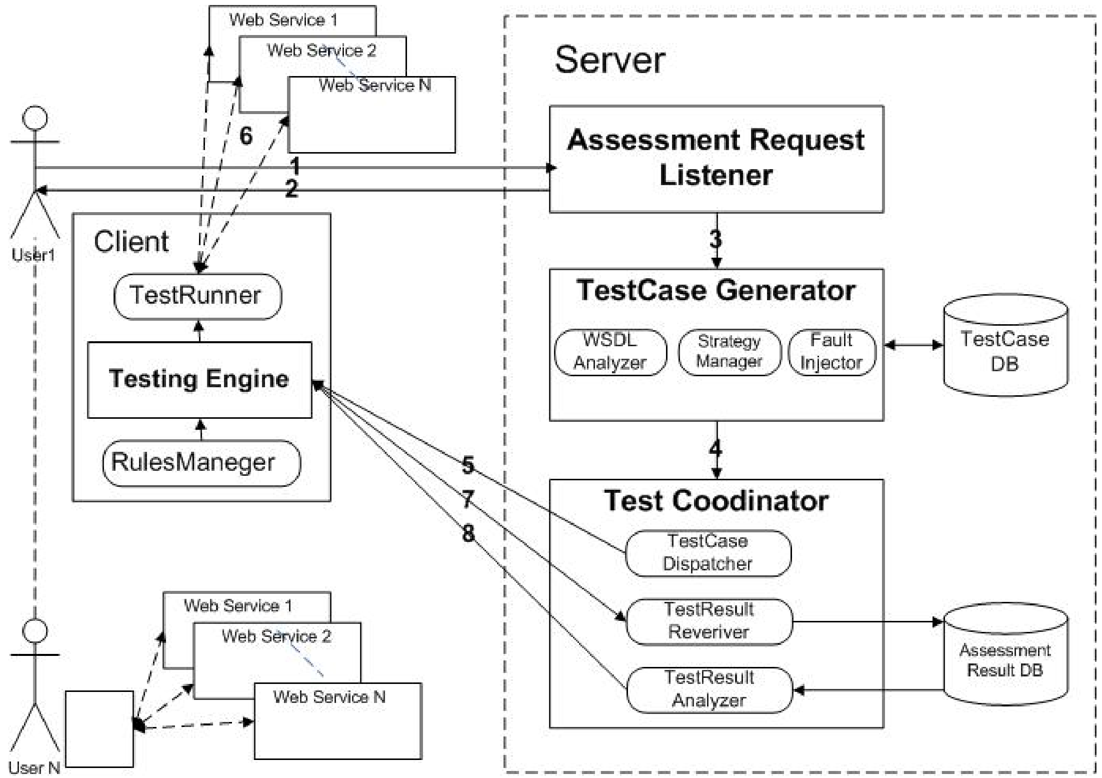
- 

Users contribute videos
- 

Users contribute knowledge

- WS Evaluation: users contribute evaluation results of Web services

Distributed Evaluation Framework

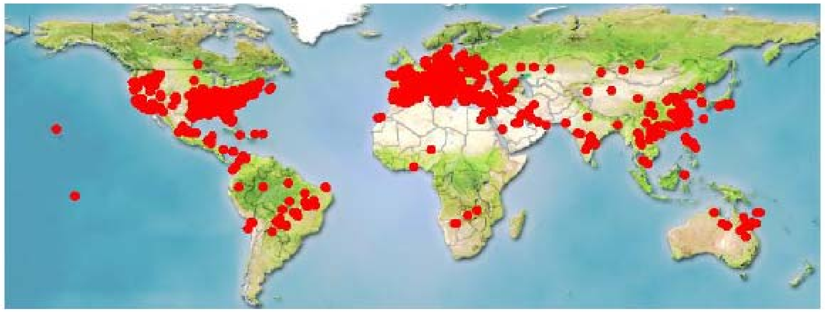


1. Evaluation request
2. Load Applet
3. Create test cases
4. Schedule test tasks
5. Assign test cases
6. Client run test cases
7. Send back results
8. Analyze and return final results to client.

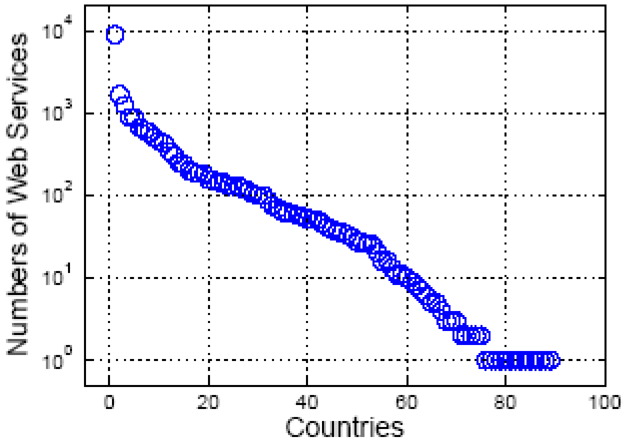
- Evaluation results from different locations
- No need good knowledge on Web service evaluation
- No need to implement evaluation mechanism

Location Information

- 21,358 Web services from 89 countries
- The top 3 countries provide 55.5% of the obtained Web services
  - United States: 8867 Web services
  - United Kingdom: 1657 Web services
  - Germany: 1246 Web services



Locations of Web Services

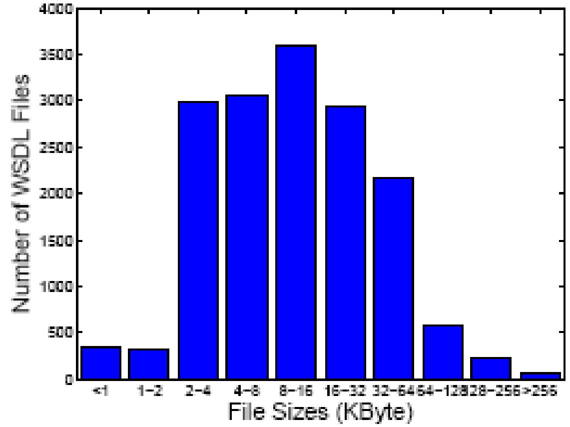


Distributions of Web Services

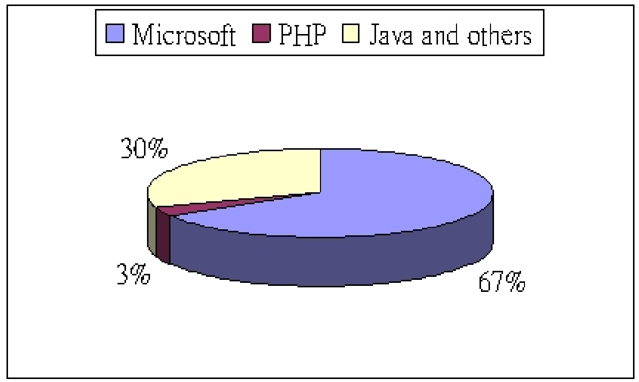
WSDL File Information

WSDL File Obtaining Failures

Code	Description	# WS	Percent
400	Bad Request	173	3.57%
401	Unauthorized	106	2.19%
403	Forbidden	153	3.16%
404	File Not Found	1468	30.31%
405	Method Not Allowed	1	0.02%
500	Internal Server Error	505	10.43%
502	Bad Gateway	51	1.05%
503	Service Unavailable	22	0.45%
504	Gateway Timeout	788	16.27%
505	HTTP Version Not Support	1	0.02%
N/A	Connection Timed Out	774	15.98%
N/A	Read Timed Out	787	16.25%
N/A	Unknown Host	12	0.25%
N/A	Redirected Too Many Times	3	0.06%
Total		4844	100.00%



Distributions of WSDL File Sizes



Development Technologies

Java Code Generation

- Axis 2 to generate Java codes for the Web services.
- Totally 235,262,555 lines of Java codes are produced.

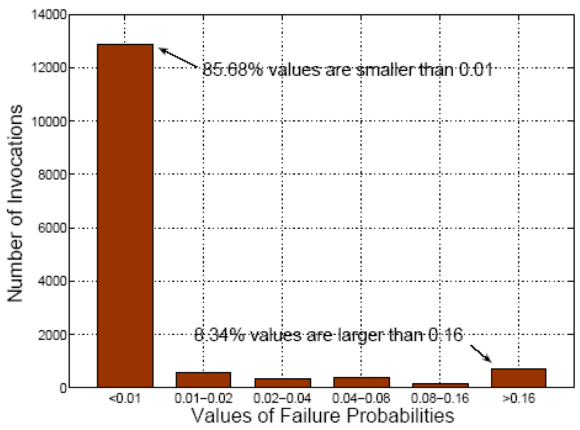
Java Code Generation Failures

Failure Type	# WS	Percent
Empty File	249	7.31%
Invalid File Format	1232	36.17%
Error Parsing WSDL	1135	33.32%
Invocation Target Exception	764	22.43%
Null QName	22	0.65%
Databinding Unmatched Type Exception	4	0.12%
Total	3406	100%

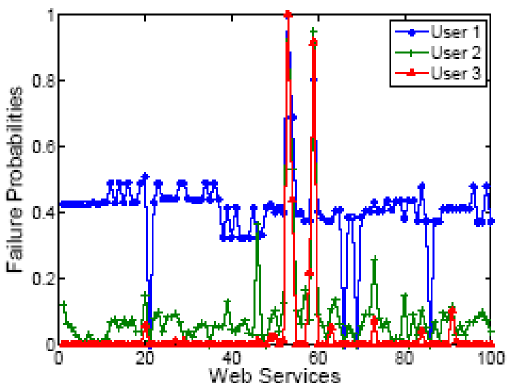
Dataset 1: 150*100*100

Statistics of the Dataset 1

Statistics	Values
Num. of Web Service Invocations	1,542,884
Num. of Service Users	150
Num. of Web Services	100
Num. of User Countries	24
Num. of Web Service Countries	22
Range of Failure Probability	0-100%
Mean of Failure Probability	4.05%
Standard Deviation of Failure Probability	17.32%



Distribution of Failure Probabilities



Three Users' Failure Probabilities

Failure Types

(1) Web service invocations can fail easily.

(2) WS invocation failures are unavoidable in the unpredictable Internet.

(3) Service fault tolerance approaches are becoming important.

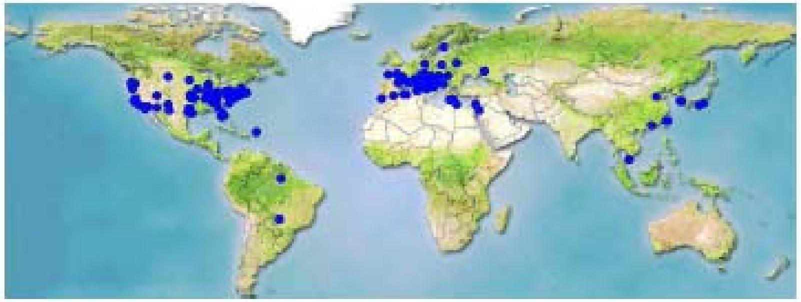
Failures of the Dataset 1

Descriptions	Number
(400)Bad Request	3
(500)Internal Server Error	26
(502)Bad Gateway	33
(503)Service Unavailable	609
java.net.SocketException: Network is unreachable	3
java.net.SocketException: Connection reset	1175
java.net.NoRouteToHostException: No route to host	415
java.net.ConnectException: Connection refused	619
java.net.SocketTimeoutException: Read timed out	4606
java.net.UnknownHostException	5847
java.net.SocketTimeoutException: Connect timed out	44809
Other errors	39
Total	58184

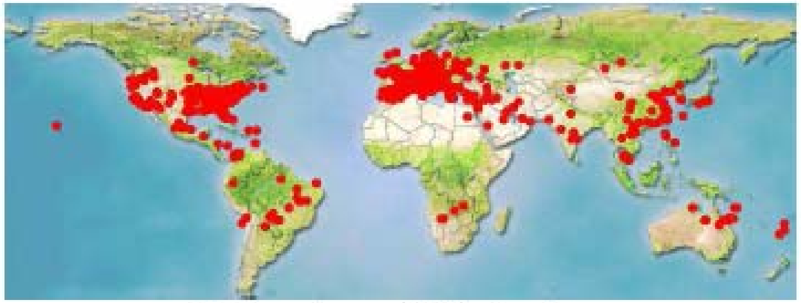
Dataset 2: 339*5825*1

Statistics of the Dataset 2

Statistics	Values
Num. of Web Service Invocations	1,974,675
Num. of Service Users	339
Num. of Web Services	5,825
Num. of User Countries	30
Num. of Web Service Countries	73
Mean of Response Time	1.43 s
Standard Deviation of Response Time	31.9 s
Mean of Throughput	102.86 kbps
Standard Deviation of Throughput	531.85 kbps



(a) Locations of Service Users



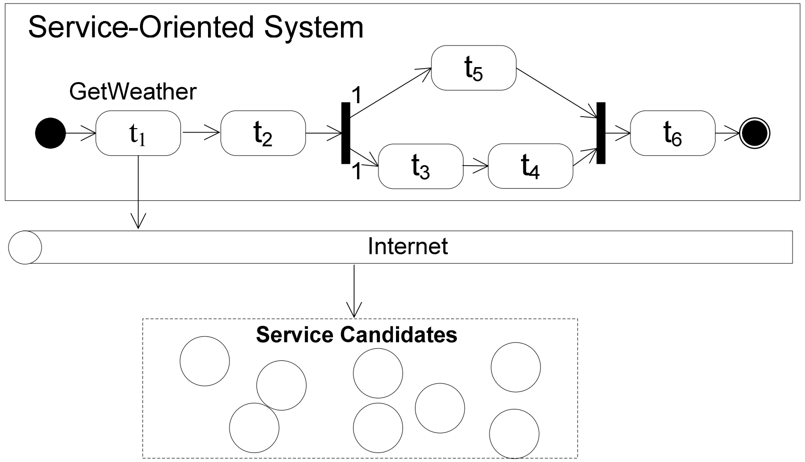
(b) Locations of Web Services

Dataset Publication

- The evaluation results are released at: http://www.wsdream.net
- Downloaded about 100 times by more than 50 universities (or research institutes) from more than 15 counties.
- The datasets can be used in research topics of:
  - Web service selection and composition
  - Web service recommendation
  - Web service QoS prediction
  - Fault-tolerant Web services
  - …………………
- The largest-scale real-world Web service QoS evaluation
- Recognized by the Best Student Paper of ICWS2010

Part 2 (Ch. 4-6): QoS Prediction of WS

Web Service Selection



- **Target**: determine the optimal Web service from a set of functionally equivalent candidates.
- **Method 1:** evaluate all the candidates
- **Weak points of Method 1:**
  - **Expensive:** Requiring a lot of Web service invocations
  - **Time-consuming:** A large number of candidates to evaluate
  - **Inaccurate:** Users are not experts on WS evaluation

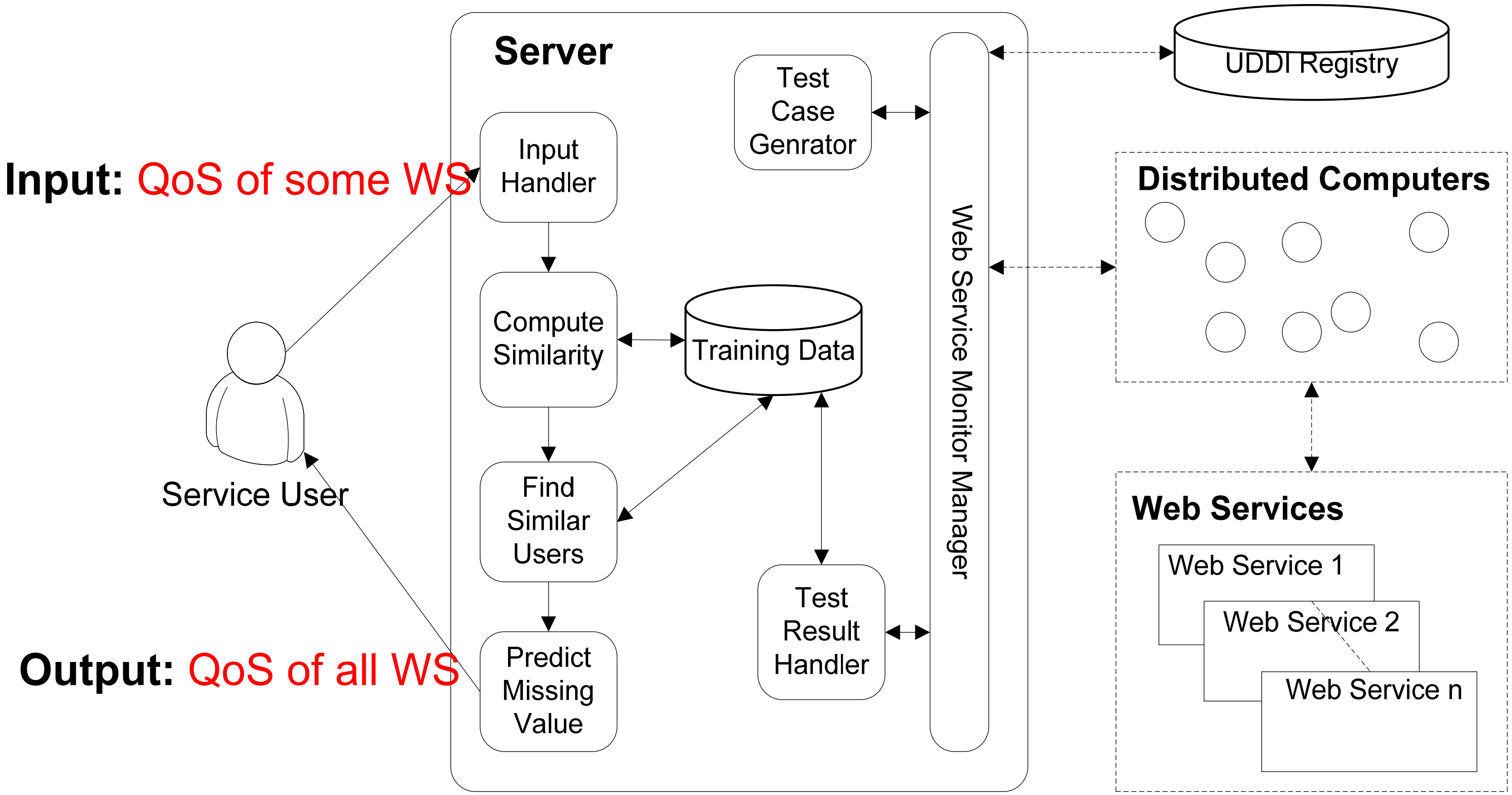
Web Service QoS Prediction

- **Method 2:** predict Web service QoS values
  - The prediction should be personalized for a specify user
  - A user may invoked some or none of the service candidates
- **Advantages**:
  - **Low cost**: no additional WS invocations for evaluation purpose
  - **Efficient**: no need to wait for the evaluation results
- **Research problem**:
  - How to make personalized WS QoS value prediction?

Previous Work

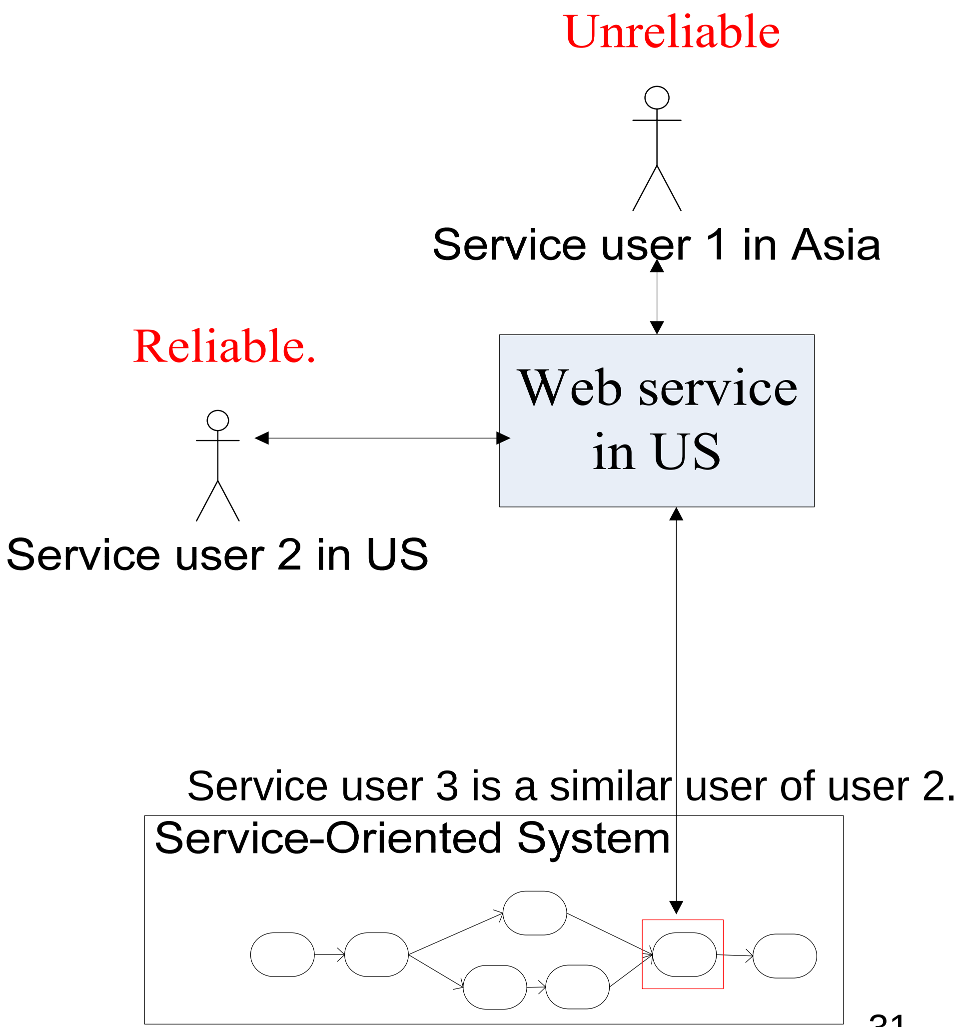
- (Sreenath & Singh, 2003; Karta, 2005;)
  - Mention the idea of applying neighborhood-based collaborative filtering methods to Web service QoS prediction
  - Employs the MovieLens dataset for experimental studies
- (Shao et al., 2007)
  - Propose a neighborhood-based collaborative filtering methods
  - Experiments using 20 real-world Web services
- Why so limited previous work?
  - No real-world WS QoS datasets from different users
  - The characteristics of Web service QoS cannot be fully mined
  - The performance of the proposed algorithms cannot be justified
- **Our contributions**:
  - Three prediction approaches
  - Convincing experiments using our released datasets

System Architecture



Approach 1: Neighborhood-based

- Key idea: Using QoS values of similar users.
- Issue: How to calculate user similarity?



Similarity Computation

- User-item matrix: M$\times$N, each entry is the failure probability of a Web service.

	ws_1	ws_2	ws_3	ws_4	ws_5	ws_6
u_1	0.1	0.1		0.2		0.3
u_2		0.1		0.2	0.5	0.3
u_3	0.4		0.3		0.1	
u_4		0.6		0.4		
u_5	0.5		0.3			0.3

- Pearson Correlation Coefficient (PCC)

$$Sim(a,u) = \frac{\sum_{i \in I_a \cap I_u} (p_{a,i} - \overline{p_a})(p_{u,i} - \overline{p_u})}{\sqrt{\sum_{i \in I_a \cap I_u} (p_{a,i} - \overline{p_a})^2} \sqrt{\sum_{i \in I_a \cap I_u} (p_{u,i} - \overline{p_u})^2}}$$

Similar User Selection

- For a user *u,* a set of similar users *S(u)* can be found by:

$$S(u) = \{a | Sim(u,a) \geq Sim_k, Sim(u,a) > 0, a \neq u\}$$

- Sim_k is the k^{th} largest PCC value with the current user *u*.
- *Sim*(*u, a*) > 0 is to exclude the dissimilar users.
- *Sim*(*u, a*) can be calculated by PCC.

Missing Value Prediction

- Given a missing value $p_{u,i}$, if the user u has similar users ($S(u) \neq null$), the missing value can be predicted by:

$$p_{u,i} = \overline{p_u} + \sum_{a \in S(u)} w_a \times (p_{a,i} - \overline{p_a}),$$

- $\overline{p_u}$ and $\overline{p_a}$ are average failure probabilities of different Web services observed by user *u* and user *a*.
- w_a can be calcualted by:

$$w_a = \frac{Sim(a,u)}{\sum_{b \in S(u)} Sim(b,u)}.$$

Missing Value Prediction

- Similar user + Similar Web services

$$p_{u,i} = \lambda \times \boxed{(\overline{p_u} + \sum_{a \in S(u)} w_a \times (p_{a,i} - \overline{p_a}))} + \xrightarrow{} \text{UPCC}$$

$$(1-\lambda) \times \boxed{(\overline{p_i} + \sum_{k \in S(i)} w_k \times (p_{u,k} - \overline{p_k}))}, \xrightarrow{} \text{IPCC}$$

$$p_{u,i} = \begin{cases} \lambda \times \overline{p_u} + (1-\lambda) \times \overline{p_i}, & \overline{p_u} \neq null \;\&\; \overline{p_i} \neq null \\ \overline{p_u}, & \overline{p_u} \neq null \;\&\; \overline{p_i} = null \\ \overline{p_i}, & \overline{p_u} = null \;\&\; \overline{p_i} \neq null \\ NoPrediction, & \overline{p_u} = null \;\&\; \overline{p_i} = null \end{cases},$$

Experiments

- 150 service users and 100 Web services
- 150*100 user-item matrix
- Randomly remove entries
- Predict the removed values
- The removed values are ground truth.

Locations of the Service Users and Web Services

User Locations	Num	WS Locations	Num
United States	72	United States	33
European Union	37	Canada	10
Japan	6	China	8
Canada	5	Germany	7
Germany	4	France	6
Brazil	3	Spain	6
France	3	United Kingdom	5
United Kindom	3	Netherlands	4
Republic of Korea	2	Poland	3
Belgium	1	Republic of Korea	3
Cyprus	1	Switzerland	3
Republic of Czech	1	Italy	2
Finland	1	Australia	1
Greece	1	Belgium	1
Hungary	1	Ireland	1
Ireland	1	Islamic Republic of Iran	1
Norway	1	Japan	1
Poland	1	New Zealand	1
Portugal	1	Norway	1
Puerto Rico	1	Serbia and Montenegro	1
Slovenia	1	South Africa	1
Spain	1	Thailand	1
Taiwan	1		
Uruguay	1		
Total	**150**	**Total**	**100**

Experiments

- Metrics of Prediction Accuracy

$$MAE = \frac{\sum_{u,i} |p_{u,i} - \widehat{p}_{u,i}|}{N}$$

$$RMSE = \sqrt{\frac{\sum_{u,i} (p_{u,i} - \widehat{p}_{u,i})^2}{N}}$$

$p_{u,i}$: the expected value

$\widehat{p}_{u,i}$: the predicted value

N : the number of predicted values

Performance Comparison

MAE and RMSE Comparison With Basic Approaches (A smaller MAE or RMSE value means a better performance)

Metric	Density	Methods	Training Users = 100						Training Users = 140					
			Response Time			Failure Rate			Response Time			Failure Rate		
			G10	G20	G30	G10	G20	G30	G10	G20	G30	G10	G20	G30
MAE	10%	UMEAN	1623	1539	1513	5.71%	5.58%	5.53%	1521	1439	1399	5.01%	5.00%	4.97%
		IMEAN	903	901	907	2.40%	2.36%	2.46%	861	872	855	1.62%	1.58%	1.68%
		UPCC	1148	877	810	4.85%	4.20%	3.86%	968	782	684	4.11%	3.47%	3.28%
		IPCC	768	736	736	2.24%	2.16%	2.21%	585	596	605	1.39%	1.33%	1.42%
		WSRec	**758**	**700**	**672**	**2.21%**	**2.08%**	**2.08%**	**560**	**533**	**500**	**1.36%**	**1.26%**	**1.24%**
	20%	UMEAN	1585	1548	1508	5.74%	5.53%	5.51%	1464	1410	1390	5.21%	4.98%	4.95%
		IMEAN	866	859	861	2.36%	2.34%	2.29%	833	837	840	1.56%	1.61%	1.62%
		UPCC	904	722	626	4.40%	3.43%	2.85%	794	626	540	3.93%	2.96%	2.43%
		IPCC	606	610	639	2.01%	1.98%	1.98%	479	509	538	1.17%	1.22%	1.28%
		WSRec	**586**	**551**	**546**	**1.93%**	**1.80%**	**1.70%**	**445**	**428**	**416**	**1.10%**	**1.08%**	**1.07%**
	30%	UMEAN	1603	1543	1508	5.64%	5.58%	5.56%	1494	1430	1387	5.12%	4.98%	4.93%
		IMEAN	856	854	853	2.26%	2.29%	2.30%	823	823	827	1.56%	1.58%	1.58%
		UPCC	915	671	572	4.25%	3.25%	2.58%	803	576	491	3.76%	2.86%	2.06%
		IPCC	563	566	602	1.84%	1.83%	1.86%	439	467	507	1.10%	1.12%	1.17%
		WSRec	**538**	**504**	**499**	**1.78%**	**1.69%**	**1.63%**	**405**	**385**	**378**	**1.05%**	**1.00%**	**0.98%**
RMSE	10%	UMEAN	3339	3250	3192	15.47%	15.04%	14.74%	3190	3109	3069	14.75%	14.42%	13.99%
		IMEAN	1441	1436	1442	5.61%	5.58%	5.85%	1112	1140	1107	3.27%	3.26%	3.38%
		UPCC	2036	1455	1335	10.84%	7.51%	6.55%	1585	1174	1005	8.86%	5.42%	4.96%
		IPCC	1335	1288	1278	5.36%	5.27%	5.53%	850	871	867	2.87%	2.82%	2.96%
		WSRec	**1329**	**1247**	**1197**	**5.31%**	**5.12%**	**5.11%**	**819**	**789**	**734**	**2.80%**	**2.61%**	**2.61%**
	20%	UMEAN	3332	3240	3211	15.49%	15.05%	14.80%	3190	3124	3062	14.72%	14.24%	14.07%
		IMEAN	1269	1252	1257	4.67%	4.62%	4.54%	997	1001	1002	2.53%	2.61%	2.63%
		UPCC	1356	1128	1019	8.07%	5.31%	4.58%	1028	837	730	7.35%	4.20%	3.24%
		IPCC	1020	1016	1056	4.15%	4.13%	4.12%	664	700	731	2.00%	2.09%	2.19%
		WSRec	**997**	**946**	**937**	**4.04%**	**3.83%**	**3.67%**	**620**	**598**	**581**	**1.88%**	**1.84%**	**1.83%**
	30%	UMEAN	3336	3246	3197	15.49%	15.00%	14.68%	3178	3103	3086	14.68%	14.25%	14.07%
		IMEAN	1207	1209	1203	4.21%	4.23%	4.22%	955	954	957	2.28%	2.29%	2.28%
		UPCC	1267	1035	924	7.72%	5.09%	4.15%	988	741	644	6.49%	3.90%	2.66%
		IPCC	950	957	995	3.72%	3.71%	3.75%	611	642	685	1.73%	1.74%	1.81%
		WSRec	**921**	**884**	**869**	**3.64%**	**3.46%**	**3.37%**	**564**	**540**	**528**	**1.64%**	**1.55%**	**1.52%**

Drawbacks of Neighborhood-based Method

- Computational complexity $O(mn + n^2)$
- Matrix sparsity problem
  - Not easy to find similar users

	i_1	i_2	i_3	i_4	i_5	i_6
u_1	0.5					0.4
u_2		0.1				
u_3					0.9	
u_4				0.7		
u_5	0.5					

Approach 2: Model-based Method

- A small number of factors influencing the QoS performance
- A user's Web service QoS experiences correspond to a linear combination of the factors
- Each row of U^T are a set of feature factors, and each column of V is a set of linear predictors

	c_1	c_2	c_3	c_4	c_5	c_6
u_1	0.98	0.23		0.22		
u_2	0.13		0.27		0.25	
u_3		0.37			0.36	
u_4	0.69		0.22	0.22		0.34

(b) User-Component Matrix

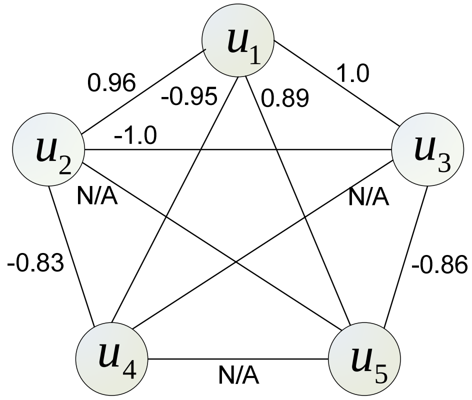
$$\min_{U,V} \mathcal{L}(R,U,V) = \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n} I_{ij}^{R}(R_{ij} - U_i^T V_j)^2 + \frac{\lambda_U}{2}\|U\|_F^2 + \frac{\lambda_V}{2}\|V\|_F^2,$$

$$\begin{bmatrix} 0.32 & 0.15 & 0.31 & 0.33 \\ 0.23 & 0.15 & 0.26 & 0.28 \\ 0.30 & 0.20 & 0.24 & 0.34 \\ 0.47 & 0.23 & 0.59 & 0.21 \end{bmatrix} \times \begin{bmatrix} 0.73 & 0.35 & 0.31 & 0.26 & 0.32 & 0.42 \\ 0.60 & 0.31 & 0.27 & 0.22 & 0.28 & 0.36 \\ 0.69 & 0.37 & 0.32 & 0.27 & 0.33 & 0.45 \\ 0.95 & 0.46 & 0.42 & 0.35 & 0.41 & 0.54 \end{bmatrix}$$

NIMF: Neighborhood–Integrated Matrix Factorization

	i_1	i_2	i_3	i_4	i_5	i_6
u_1	0.5	1.2		0.3		0.4
u_2		0.8		0.6	0.5	
u_3	0.4		0.3		0.9	
u_4		0.6		0.7		
u_5	0.5		0.7			0.3

(a) User-Item Matrix

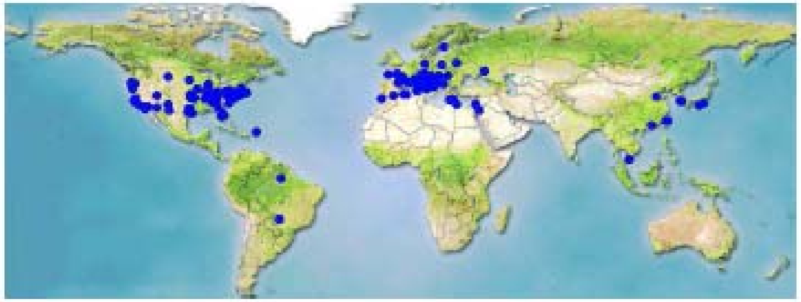


(b) PCC Values

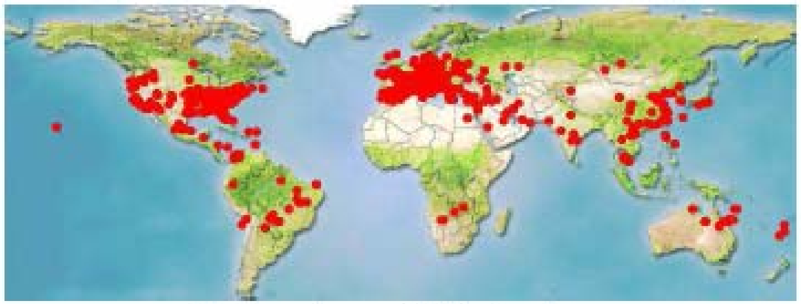
$$
\begin{aligned}
&\mathcal{L}(R, S, U, V) \\
&= \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{n} I_{ij}^{R} \Big(R_{ij} - \big(\alpha U_i^T V_j + (1-\alpha) \sum_{k \in \mathcal{T}(i)} S_{ik} U_k^T V_j\big)\Big)^2 \\
&+ \frac{\lambda_U}{2} \|U\|_F^2 + \frac{\lambda_V}{2} \|V\|_F^2,
\end{aligned}
$$

$$
S_{ik} = \frac{PCC(i,k)}{\sum_{k \in \mathcal{T}(i)} PCC(i,k)}
$$

Location Information



(a) Locations of Service Users

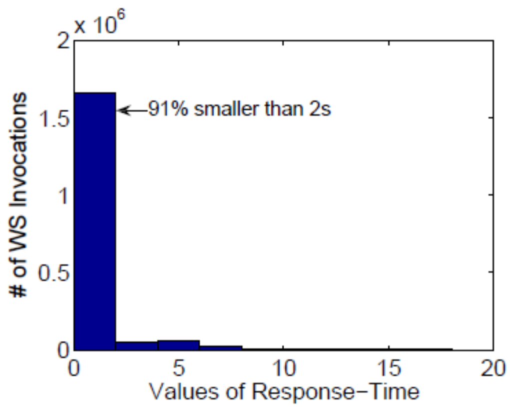


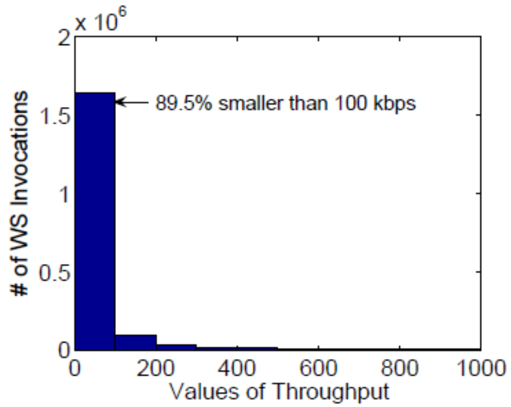
(b) Locations of Web Services

Location Information: (a) locations of service users, totally 339 service users from 30 countries are plotted; (b) locations of Web services, totally 5,825 real-world Web services from 73 countries are plotted. Each user in (a) invoked all the Web services in (b). Totally 1,974,675 Web service invocation results are collected in our experiments.

Statistics of the WS QoS Dataset

Statistics	Values
Num. of Service Users	339
Num. of Web Services	5,825
Num. of Web Service Invocations	1,974,675
Range of Response-time	1-20 s
Avg. Value of Response-time	0.9085 s
Range of Throughput	1-1000 kbps
Avg. Value of Throughput	47.5616 kbps



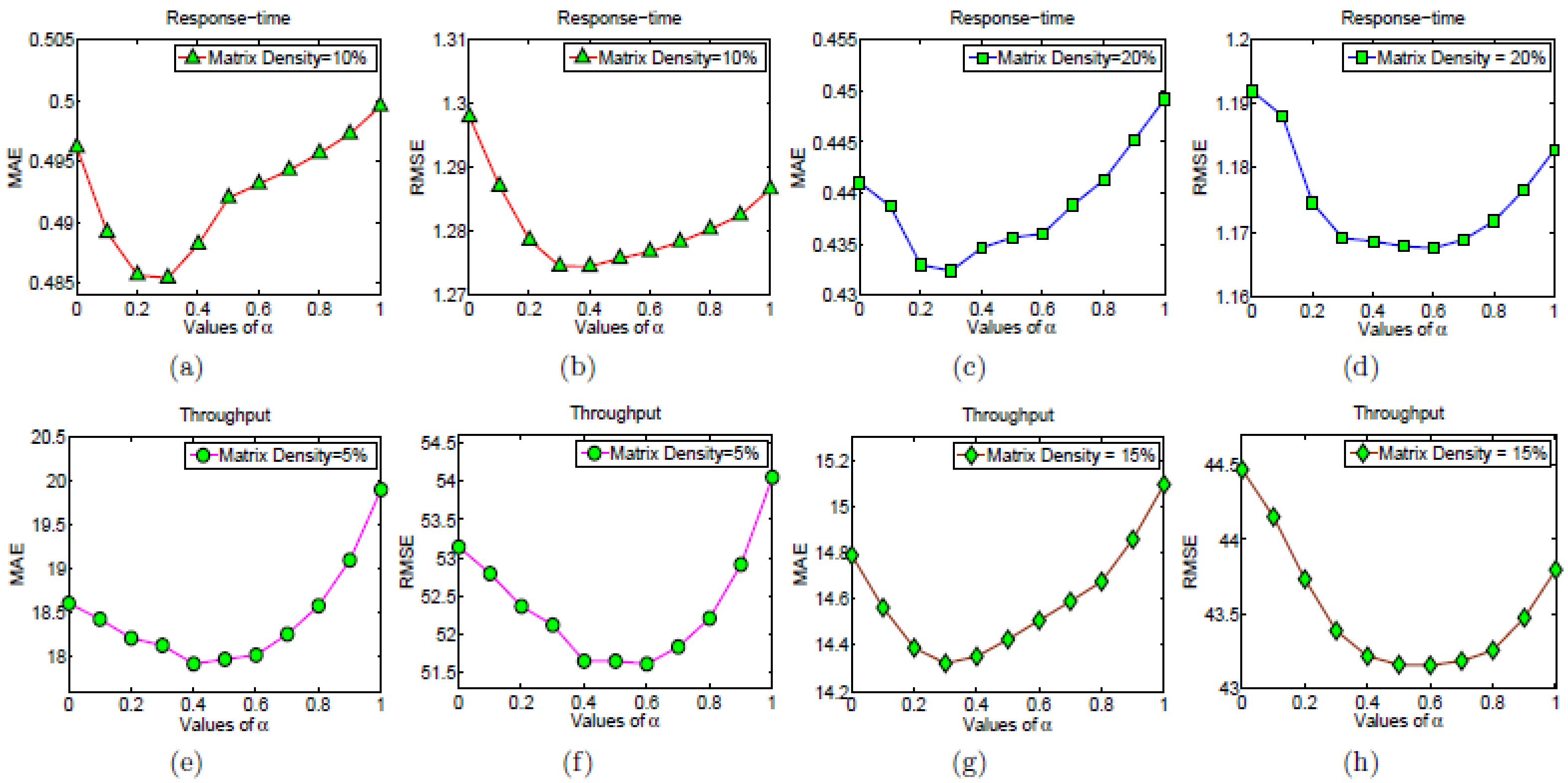


Performance Comparison

Performance Comparison (A Smaller MAE or RMSE Value Means a Better Performance)

QoS	Methods	Matrix Density=5%		Matrix Density=10%		Matrix Density=15%		Matrix Density=20%	
		MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
Response-time (0-20 s)	UMEAN	0.8785	1.8591	0.8783	1.8555	0.8768	1.8548	0.8747	1.8557
	IMEAN	0.7015	1.5813	0.6918	1.5440	0.6867	1.5342	0.6818	1.5311
	UPCC	0.6261	1.4078	0.5517	1.3151	0.5159	1.2680	0.4884	1.2334
	IPCC	0.6897	1.4296	0.5917	1.3268	0.5037	1.2552	0.4459	1.2095
	WSRec	0.6234	1.4078	0.5365	1.3043	0.4965	1.2467	0.4407	1.2012
	NMF	0.6182	1.5746	0.6040	1.5494	0.5990	1.5345	0.5982	1.5331
	PMF	0.5678	1.4735	0.4996	1.2866	0.4720	1.2163	0.4492	1.1828
	NIMF	**0.5514**	**1.4075**	**0.4854**	**1.2745**	**0.4534**	**1.1980**	**0.4357**	**1.1678**
Throughput (0-1000 kbps)	UMEAN	54.0084	110.2821	53.6700	110.2977	53.8792	110.1751	53.7114	110.1708
	IMEAN	27.3558	66.6344	26.8318	64.7674	26.6239	64.3986	26.6364	64.1082
	UPCC	26.1230	61.6108	21.2695	54.3701	18.7455	50.7768	17.5546	48.2621
	IPCC	29.2651	64.2285	27.3993	60.0825	26.4319	57.8593	25.0273	55.4970
	WSRec	25.8755	60.8685	19.9754	54.8761	17.5543	47.8235	16.0762	47.8749
	NMF	25.7529	65.8517	17.8411	53.9896	15.8939	51.7322	15.2516	48.6330
	PMF	19.9034	54.0508	16.1755	46.4439	15.0956	43.7957	14.6694	42.4855
	NIMF	**17.9297**	**51.6573**	**16.0542**	**45.9409**	**14.4363**	**43.1596**	**13.7099**	**41.1689**

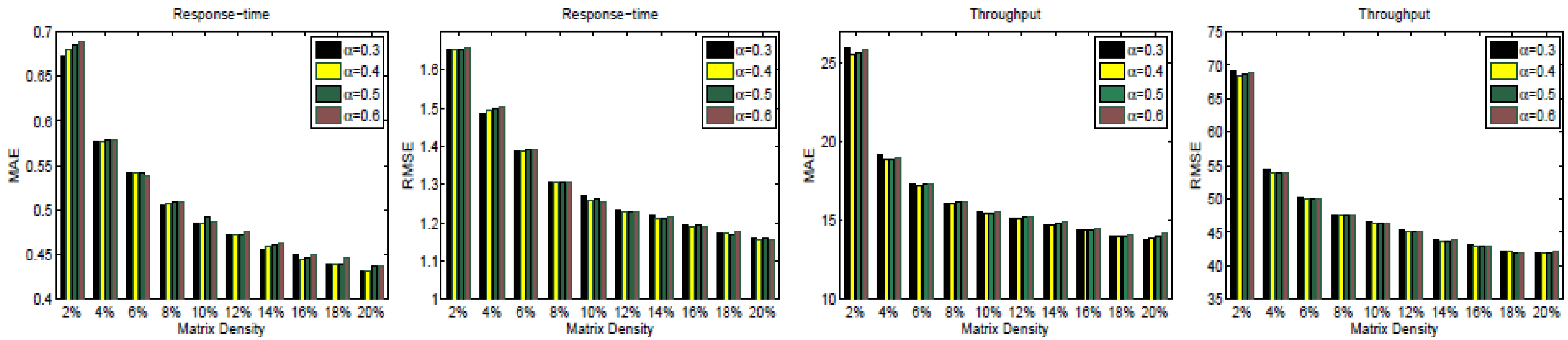
Impact of Parameter



Impact of Parameter α (Dimensionality = 10)

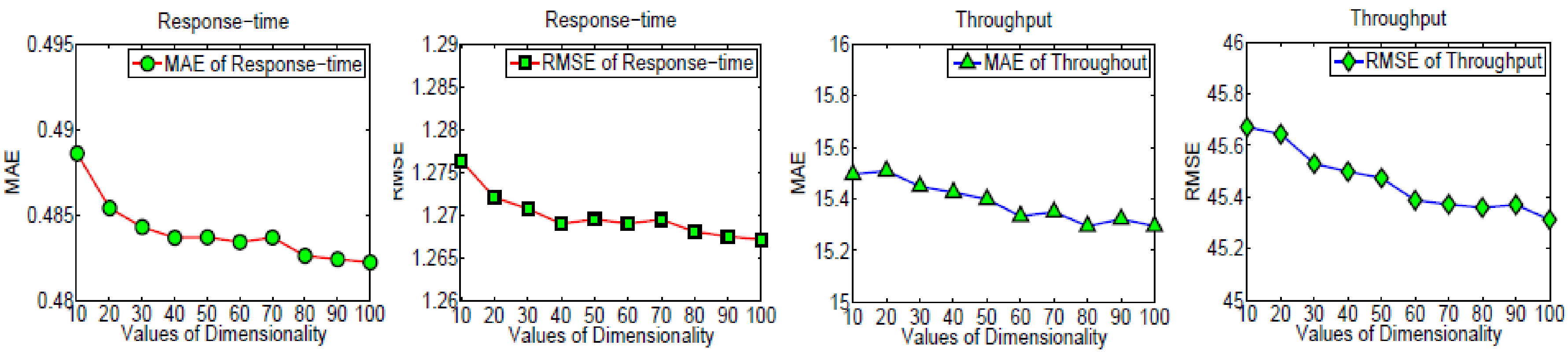
The parameter α controls how much our method relies on the users themselves and their similar users

Impact of Parameter



(a) (b) (c) (d)

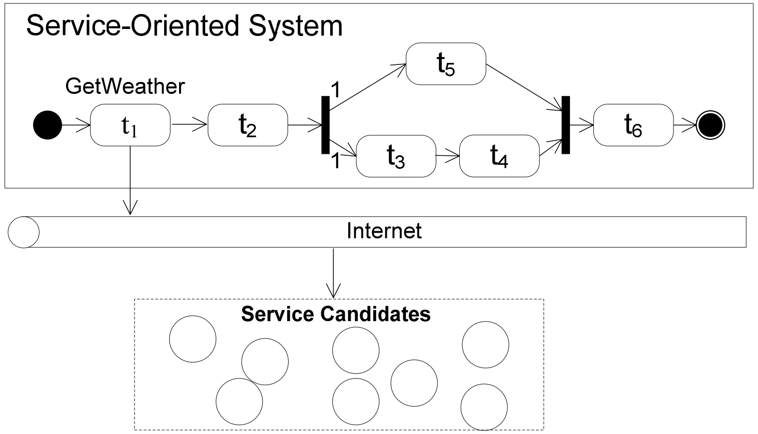
Impact of Matrix Density (Dimensionality = 10, α = 0.4)



(a) (b) (c) (d)

Impact of Dimensionality (α = 0.4, Matrix Density = 10%)

Approach 3: Ranking-Based Prediction



- Select the optimal Web service from the candidates
  - Neighborhood-based approaches:
    Predict QoS values → rank the candidates
  - Ranking-based approach:
    Rank the candidates directly without predicting QoS values

Expected values: (2, 3, 5) on (ws1, ws2, ws3)

Prediction 1: (3, 2, 4); MAE = (|2−3| + |3−2| + |5−4|)/3 = 1

Prediction 2: (1, 2, 3); MAE = (|2−1| + |3−2| + |5−3|)/3 = 1.3

Ranking-based: Expected Ranking: ws1 < ws2 < ws3

Prediction 1: ws2 < ws1 < ws3

Prediction 2: ws1 < ws2 < ws3

User Preference

- User preference on two Web services which have been invoked previously:

$$\Psi(i, j) = q_i - q_j$$

- User preference on pairs of Web services that have not both been invoked by the current user:

$$\Psi(i,j) = \sum_{v \in N(u)^{ij}} \frac{Sim(u,v)}{\sum_{v \in N(u)^{ij}} Sim(u,v)} (q_{v,i} - q_{v,j})$$

Kendall Rank Correlation Coefficient

- Kendall Rank Correlation Coefficient (KRCC)

$$Sim(u,v) = \frac{C - D}{N(N-1)/2}$$

  – N is the number of Web services.
  – C is the number of concordant pairs between two rankings.
  – D is the number of discordant pairs.

Target Web services: (ws1, ws2, ws3)
User 1 observed response-time: (2, 3, 5)
User 2 observed response-time: (1, 2, 3)

User 1: ws1< ws2, ws1< ws3, ws2 < ws3
User 2: ws1< ws2, ws1< ws3, ws2 < ws3

N = 3; C = 3; D = 0;
Sim(user1, user2) = (3-0) / (3(3-1)/2) = 1

Target Web services: (ws1, ws2, ws3)
User 1 observed response-time: (2, 3, 5)
User 2 observed response-time: (3, 2, 1)

User 1: ws1< ws2, ws1< ws3, ws2 < ws3
User 2: ws1> ws2, ws1> ws3, ws2 > ws3

N = 3; C = 0; D = 3;
Sim(user1, user2) = (0-3) / (3(3-1)/2) = -1

Problem Modeling

- Given a preference function, choose a ranking that agrees with the preferences as much as possible.
- **Object function**:

$$V^{\Psi}(\rho) = \sum_{i,j:\rho(i)>\rho(j)} \Psi(i,j)$$

- **Target**: produce a ranking that maximizes the objective function.
- **Trivial Solution**: search through possible rankings
  - *n!* possible rankings
  - NP-Complete problem

Ranking Algorithm

Algorithm 1: Greedy Order Algorithm: CloudRank

Input: an employed component set E, a full component set I, a preference function Ψ

Output: a component ranking $\hat{\rho}$

$F = E$;
while $F \neq \emptyset$ **do**
  $t = \arg\max_{i \in F} q_i$;
  $\rho_e(t) = |E| - |F| + 1$;
  $F = F - \{t\}$;
end
foreach $i \in I$ **do**
  $\pi(i) = \sum_{j \in I} \Psi(i, j)$;
end
$n = |I|$;
while $I \neq \emptyset$ **do**
  $t = \arg\max_{i \in I} \pi(i)$;
  $\hat{\rho}(t) = n - |I| + 1$;
  $I = I - \{t\}$;
  foreach $i \in I$ **do**
    $\pi(i) = \pi(i) - \Psi(i, t)$
  end
end
while $E \neq \emptyset$ **do**
  $e = \arg\min_{i \in E} \rho_e i$;
  $index = \min_{i \in E} \hat{\rho}(i)$;
  $\hat{\rho}(e) = index$;
  $E = E - \{e\}$;
end

- Step 1: for each service candidate, calculate the sum of preference values with all other candidates in the candidate set.
- Step 2: a candidate with largest preference values is more preferred by the user. Rank this candidate at highest position.
- Step 3: remove the selected candidate from the candidate set. Go to step 1.

Evaluation Metric

- Normalized Discounted Cumulative Gain (NDCG)

$$NDCG_k = \frac{DCG_k}{IDCG_k}$$

  - The $NDCG_k$ value is on the interval of 0.0 to 1.0, where larger value stands for better ranking accuracy.
  - DCG_k and $IDCG_k$ are the *DCG* values of the top-K components of the predicted ranking and ideal ranking, respectively.

$$DCG_k = rel_1 + \sum_{i=2}^{k} \frac{rel_i}{log_2 i}$$

  - rel_i is the QoS value of the component at position *i* in the ranking.

Performance Comparison

NDCG Comparison of Response Time (Larger value indicates better ranking accuracy)

	Matrix Density = 10%			Matrix Density = 30%			Matrix Density = 50%		
Methods	NDCG3	NDCG10	NDCG100	NDCG3	NDCG10	NDCG100	NDCG3	NDCG10	NDCG100
UVS:	0.9491	0.9104	0.9514	0.9689	0.9476	0.9726	0.9547	0.9408	0.9663
UPCC:	0.9347	0.8968	0.9414	0.9696	0.9489	0.9729	0.9541	0.9417	0.9666
IVS:	0.9710	0.9308	0.9637	0.9689	0.9442	0.9690	0.9548	0.9417	0.9661
IPCC:	0.9737	0.9359	0.9656	0.9688	0.9466	0.9702	0.9588	0.9484	0.9695
UIVS:	0.9719	0.9304	0.9639	0.9689	0.9441	0.9696	0.9553	0.9423	0.9663
UIPCC:	0.9730	0.9354	0.9653	0.9691	0.9477	0.9711	0.9584	0.9482	0.9695
Greedy	0.9789	0.9523	0.9755	0.9816	0.9728	0.9860	0.9939	0.9843	0.9921
CloudRank	**0.9792**	**0.9532**	**0.9763**	**0.9854**	**0.9760**	**0.9888**	**0.9959**	**0.9864**	**0.9947**
	0.63%	1.85%	1.11%	1.63%	2.85%	1.63%	3.87%	4.01%	2.60%

Part 3 (Ch. 7-8): Fault-Tolerant Web Services

Fault-Tolerant Web Services

- It is difficult to build reliable service-oriented systems.
  - Reliability of the system is highly dependent on the remote Web service components.
  - Web services are usually hosted by other organizations.
    - May contain faults
    - May become unavailable suddenly
    - Source codes of the Web services are usually unavailable
  - The Internet environment is unpredictable.

How to employ the redundant Web services and their QoS values for building fault-tolerant service-oriented systems?

Background

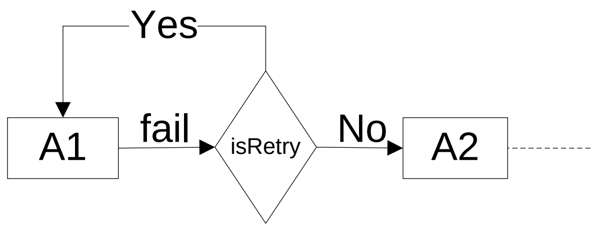
- Traditional software fault tolerance techniques:
  – Recovery block [63]
  – N-Version Programming (NVP) [11]
  – N self-checking programming [42]
  – Distributed recovery block [40]
- Fault tolerance strategies for Web services.
  – Passive strategies: FT-SOAP [28] , FT-CORBA [74]
  – Active strategies: FTWeb [70], Thema [52], WS-Replication [67], SWS [44], Perpetual [61]
- Our Contributions:
  – Adaptive fault tolerance strategy design
  – Systematic and extendable framework for building fault-tolerance Web services.

Adaptive Fault Tolerance

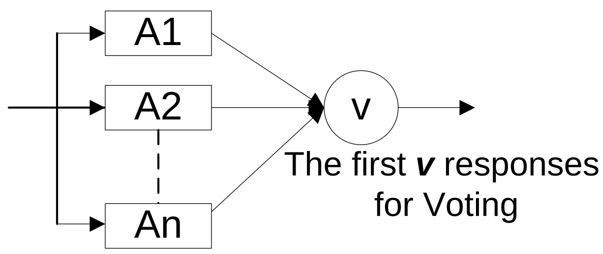
- Internet environment is highly dynamic
  - Network condition changes
  - Software/hardware updates of the Web services
  - Server workload changes
- Traditional fault tolerance strategies are too static
  - Fixed at design time
  - Cannot adaptive to the dynamic environment

Adaptive Fault Tolerance

- Idea: determine optimal fault tolerance strategy dynamically at runtime based on the Web service QoS values.

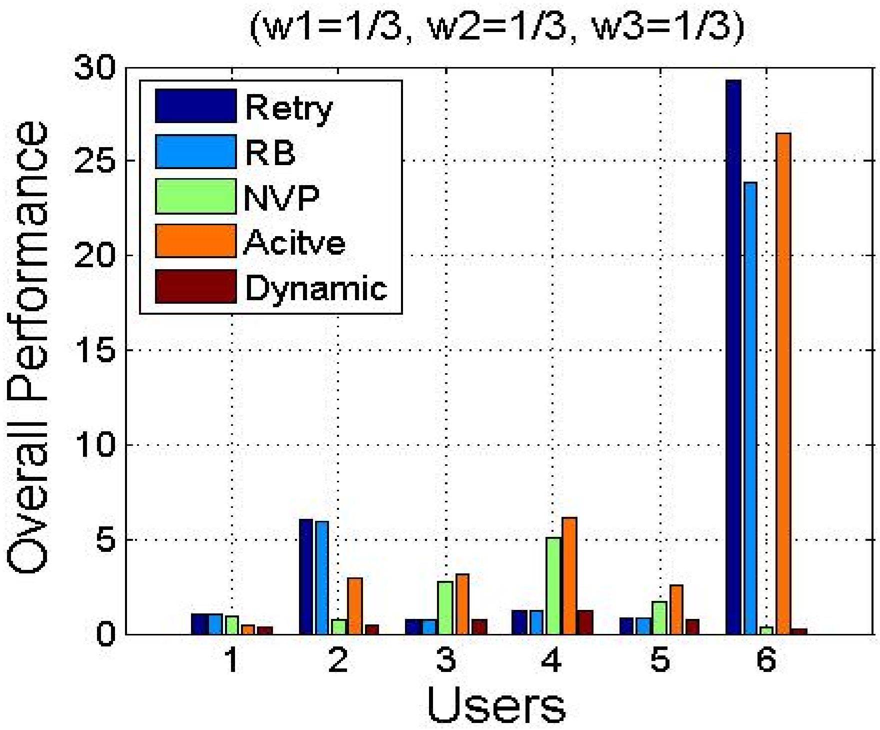


(1). Dynamic Sequential Strategy



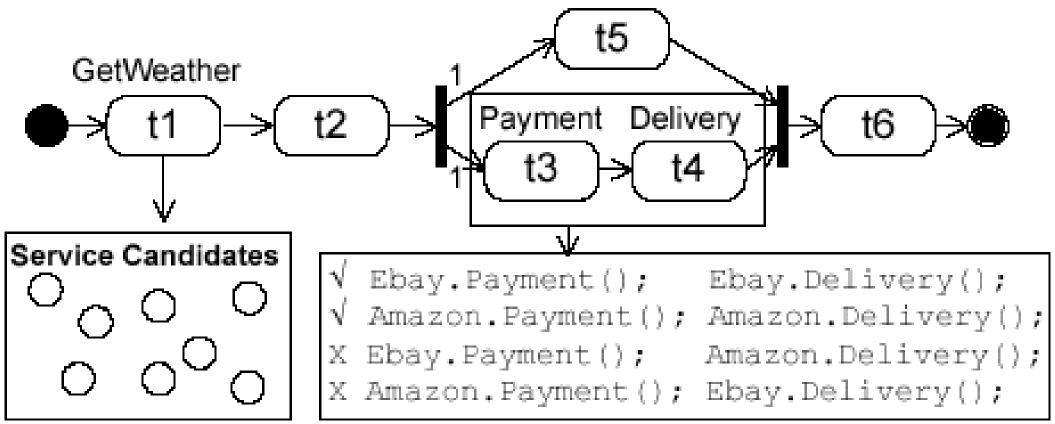
(2). Dynamic Parallel Strategy

Adaptive Fault Tolerance



- Static fault tolerance strategies have good performance in some cases, but have bad performance in others.
- The proposed strategy obtains the best overall performance for all the six users.

Fault-Tolerant Framework



- Target:
  - Optimal FT strategy selection for each task under local and global constraints
- Local constraint: Response time of t1 < 1000 ms
- Global constraint: Success-rate of the whole service plan > 99%

Fault-Tolerant Framework

- 0-1 Integer Programming Problem

Problem 2: **Minimize:**

$$\sum_{i \in ER_i} \sum_{j \in S_i} u_{ij} x_{ij}$$

Subject to:

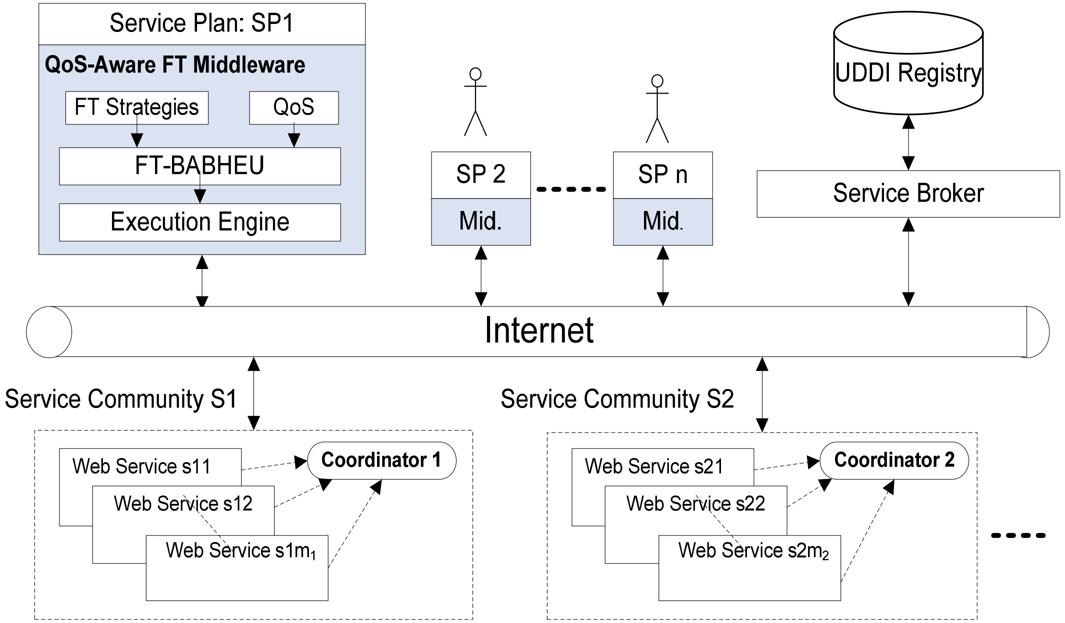
$$\sum_{i \in ER_i} \sum_{j \in S_i} q^y_{ij} x_{ij} \le gc^y (y = 2, 3, 4)$$

$$\forall k, \sum_{i \in SR_{ik}} \sum_{j \in S_i} q^y_{ij} x_{ij} \le gc^y (y = 6, 8)$$

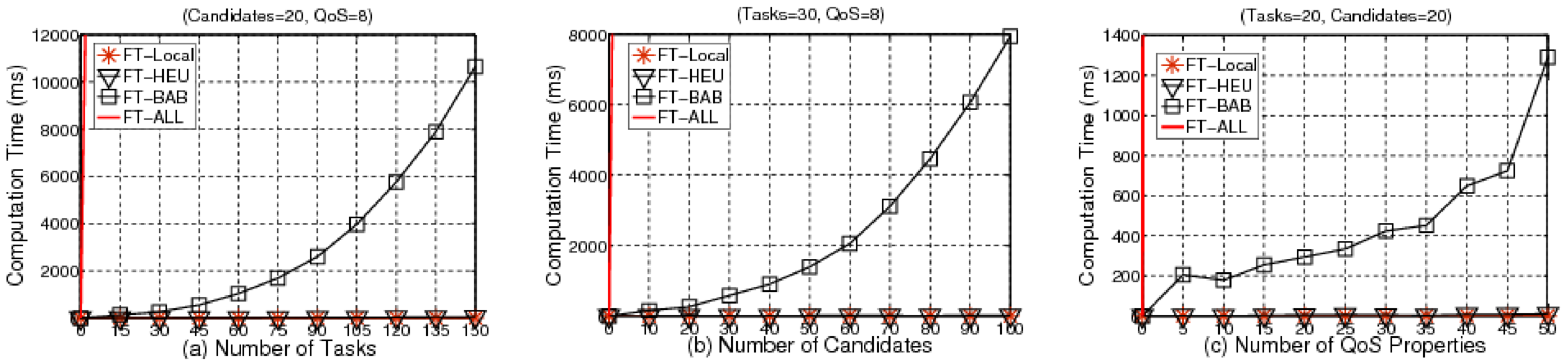
$$\prod_{i \in ER_i} \prod_{j \in S_i} (q^y_{ij})^{x_{ij}} \le gc^y (y = 1, 5, 7)$$

$$\forall SFT_i, x_{y_1 j} = x_{y_2 j} = ... = x_{y_{n_i} j} (t_{y_i} \in SFT_i)$$

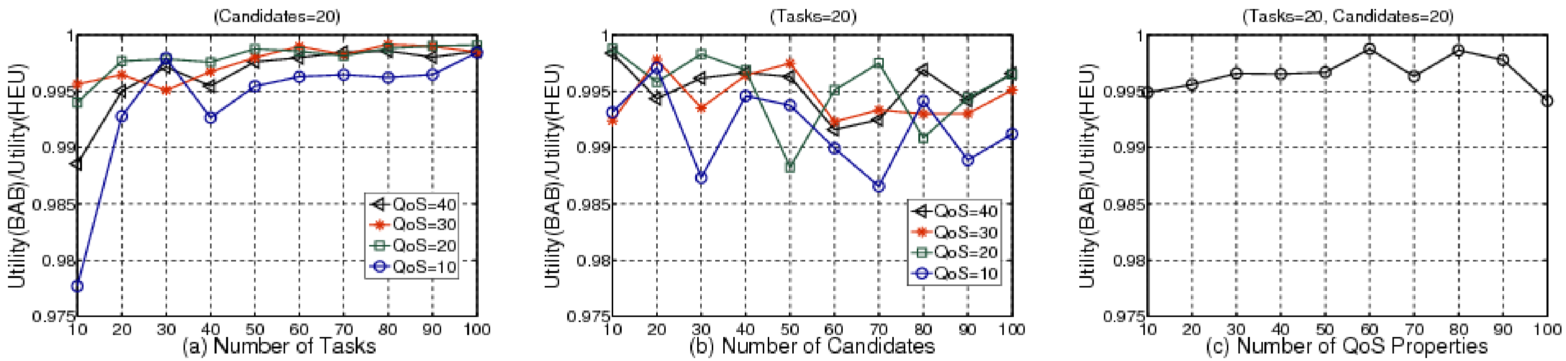
$$\forall i, \sum_{j \in S_i} x_{ij} = 1; x_{ij} \in \{0, 1\}$$



Fault-Tolerant Framework



Performance of Computation Time

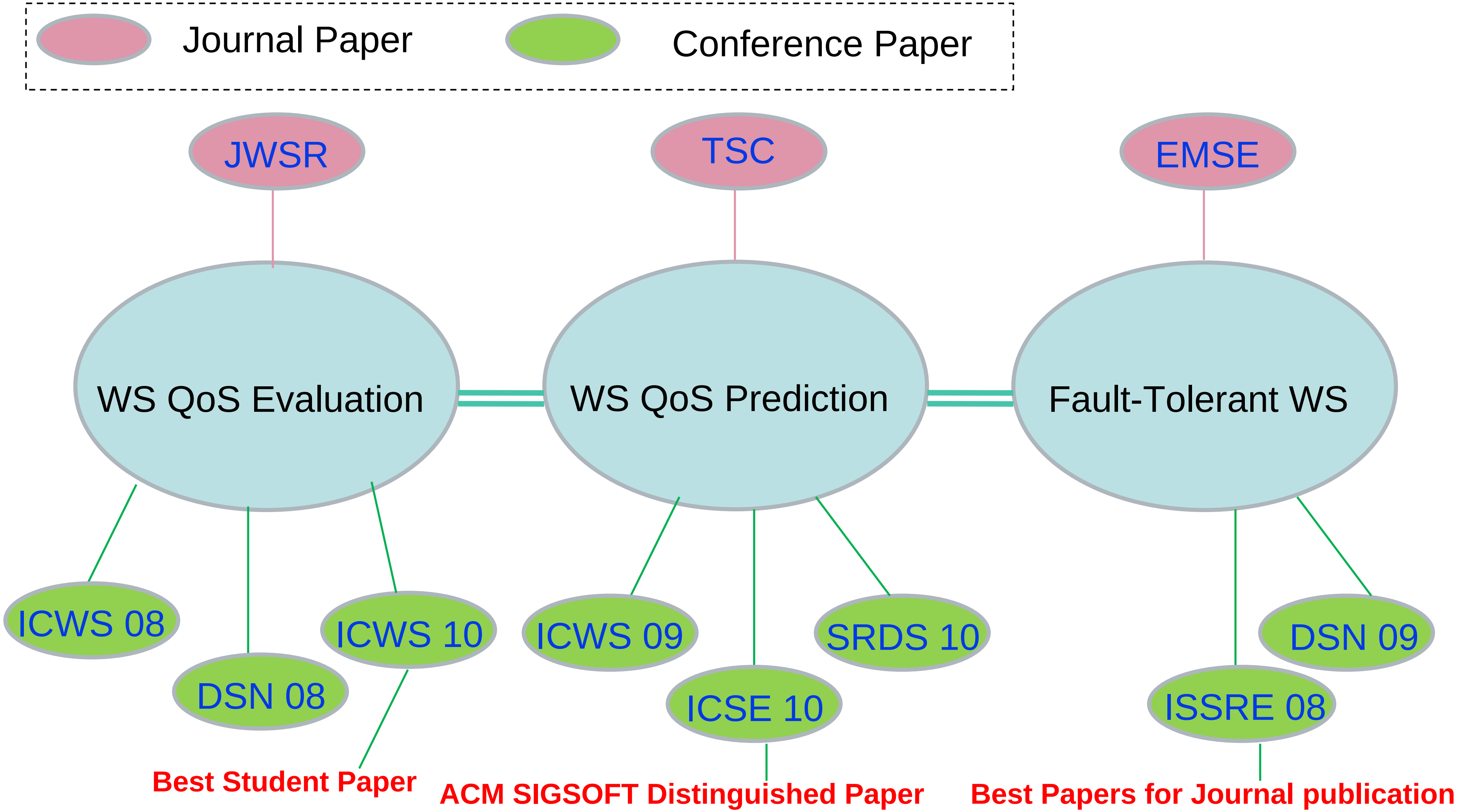


Performance of Selection Results

Conclusion

- QoS evaluation of Web services
  - Two large-scale real-world evaluations on Web services which were never attempted before
  - The published dataset website was widely accessed
- QoS prediction of Web services
  - Three prediction approaches
  - A lot of followup work on this topic using the released datasets
- Fault-tolerant Web services
  - Adaptive fault tolerance strategy
  - Systematic framework for fault-tolerant service oriented systems

Structure of My Work



A Summary of My Published Thesis Work

1. **Zibin Zheng**, Hao Ma, Michael R. Lyu, and Irwin King "QoS-Aware Web Service Recommendation by Collaborative Filtering", *IEEE Transactions on Service Computing (**TSC**)*, to be published.
2. **Zibin Zheng**, Michael R. Lyu, "Optimal Fault Tolerance Strategy Selection for Web Services," *International Journal of Web Service Research (**JWSR**)*, Vol. 7, no. 4, pp. 21-40, 2010.
3. **Zibin Zheng**, Michael R. Lyu, "An Adaptive QoS-Aware Fault Tolerance Strategy for Web Services," *Springer Journal of Empirical Software Engineering (**EMSE**)*, Vol. 15, No. 4, pp. 323-345, 2010.
4. **Zibin Zheng**, Yilei Zhang, and Michael R. Lyu, "CloudRank: A QoS-Driven Component Ranking Framework for Cloud Computing", *in proc. 28th IEEE International Symposium on Reliable Distributed Systems (**SRDS2010**)*, New Delhi, India. Oct.31-Nov.3, 2010. [Acceptance rate: 21/93 = 22.6%]
5. **Zibin Zheng**, Yilei Zhang, and Michael R. Lyu, "Distributed QoS Evaluation for Real-World Web Services,"*in Proceedings of the 8th International Conference on Web Services (**ICWS2010**)*, Miami, Florida, USA, July 5-10, 2010.
(**Best Student Paper Award**) [Acceptance rate: 39/221 = 17.6%]
6. **Zibin Zheng**, Michael R. Lyu, "Collaborative Reliability Prediction for Service-Oriented Systems", *in Proc. ACM/IEEE 32nd International Conference on Software Engineering (**ICSE2010**)*, Cape Town, South Africa, May 2-8, 2010, pp. 35-44.
(**ACM SIGSOFT Distinguished Paper Award**) [Acceptance rate: 52/380 = 13.7%]

A Summary of My Published Thesis Work

7. **Zibin Zheng**, Hao Ma, Michael R. Lyu, Irwin King, "WSRec: A Collaborative Filtering based Web Service Recommender System", *in Proceedings of the 7th IEEE International Conference on Web Services (**ICWS2009**)*, Los Angeles, CA, USA, July 6-10, 2009, pp. 437-444. [Acceptance rate: 63/404 = 15.6%]
8. **Zibin Zheng**, Michael R. Lyu, "A QoS-Aware Fault Tolerant Middleware for Dependable Service Composition", *in Proceedings of the 39th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (**DSN2009**)*, Lisbon, Portugal, June 29–July 02, 2009, pp. 239-248. [Acceptance rate: 21/96 = 21.9%]
9. **Zibin Zheng**, Michael R. Lyu, "A QoS-Aware Middleware for Fault Tolerant Web services", *in Proceedings of the IEEE International Symposium on Software Reliability Engineering (**ISSRE2008**)*, Seattle, USA, Nov. 11-14, 2008, pp.97-106. [Acceptance rate: 29/116 = 25%]
10. **Zibin Zheng**, Michael R. Lyu, "A Distributed Replication Strategy Evaluation and Selection Framework for Fault Tolerant Web Services", *in Proceedings of the 6th IEEE International Conference on Web Services (**ICWS2008**)*, Beijing, China, Sep. 23-26, 2008, pp.145-152. [Acceptance rate: 43/269 = 16%]
11. **Zibin Zheng**, Michael R. Lyu, "WS-DREAM: A Distributed Reliability Assessment Mechanism for Web Services", *in Proceedings of the 38th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (**DSN2008**)*, Anchorage, Alaska, USA, June 24-27, 2008, pp.392-397. [Acceptance rate: 23/87 = 26%]

QoS Management of Web Services

Zibin Zheng (Ben)
Supervisor: Prof. Michael R. Lyu

Thank You!

The Chinese University of Hong Kong
Dec. 10, 2010