Modeling and Exploiting QoS Prediction in Cloud and Service Computing

Yilei Zhang

Thesis Committee
Chair: Prof. Pak Ching LEE
Supervisor: Prof. Michael R. LYU
Member: Prof. Fung Yu YOUNG
External Examiner: Prof. Qing LI

Sep. 12, 2013

Outlines

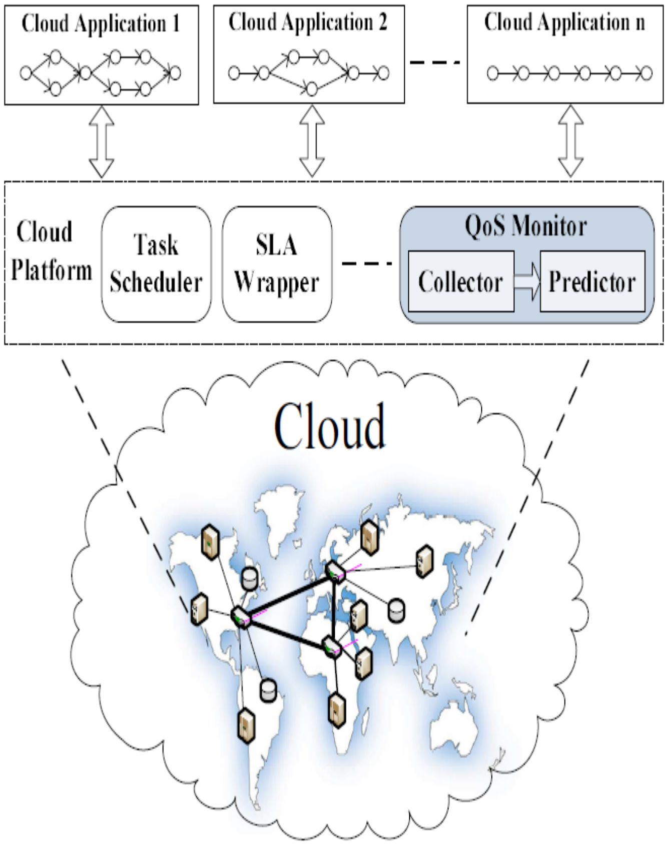
- Introduction
- Part1: QoS Prediction Approaches
  - Neighborhood-Based Approach
  - Time-aware Model-Based Approach
  - Online Approach
- Part2: QoS-Aware Web Service Searching
- Part3: QoS-Aware Byzantine Fault Tolerance
- Conclusion

Cloud Computing

- Cloud component
  - Software, server, database, etc.
- On-demand

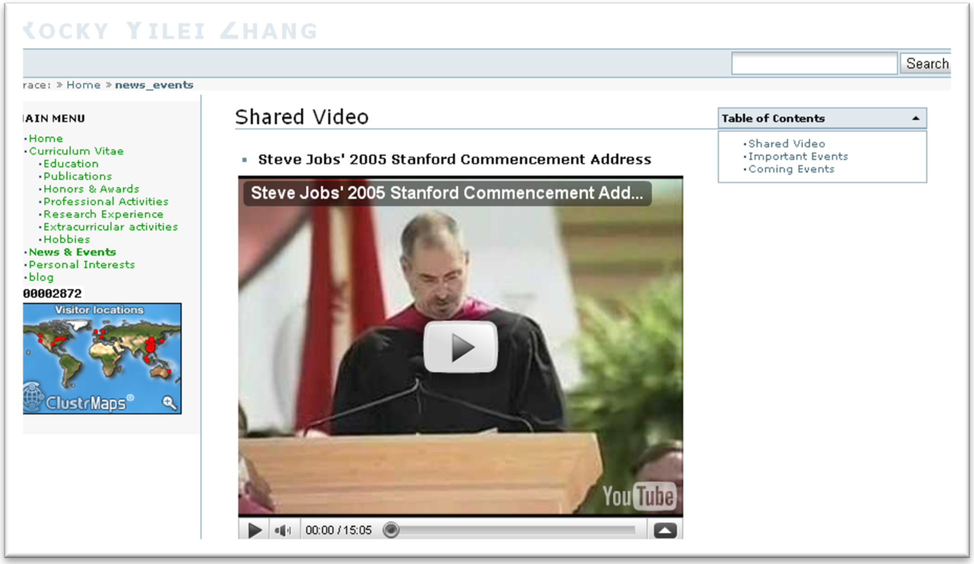
Cloud Applications

- Software-as-a-Service (SaaS)
  - Large-scale, complicated, time sensitive, high-quality
- Case 1: New York Times
  - Convert scanned articles to PDF
  - 15 million files, 4TB data
  - EC2 & S3, 100 computers 24 hours
- Case 2: Nasdaq
  - Stock and fund information
  - Millions of files, per 10 minutes
  - S3



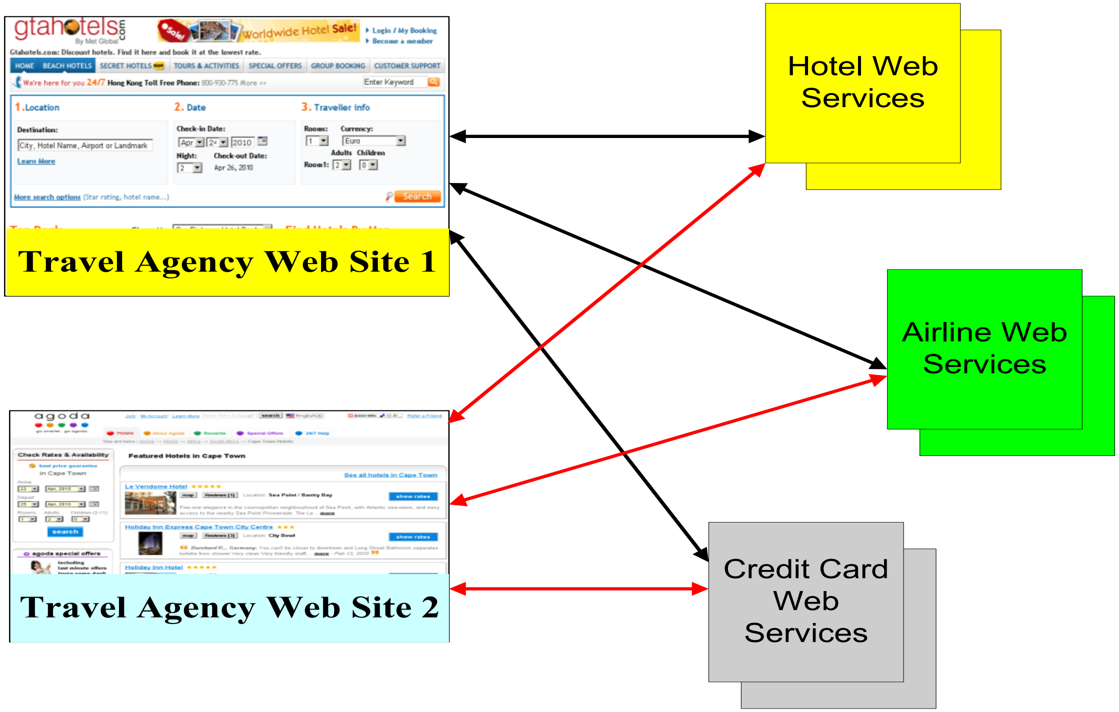
Web Services

- Web APIs
  - Accessed over a network,
  - Executed on remote systems
  - Loosely-coupled
  - Compositional nature



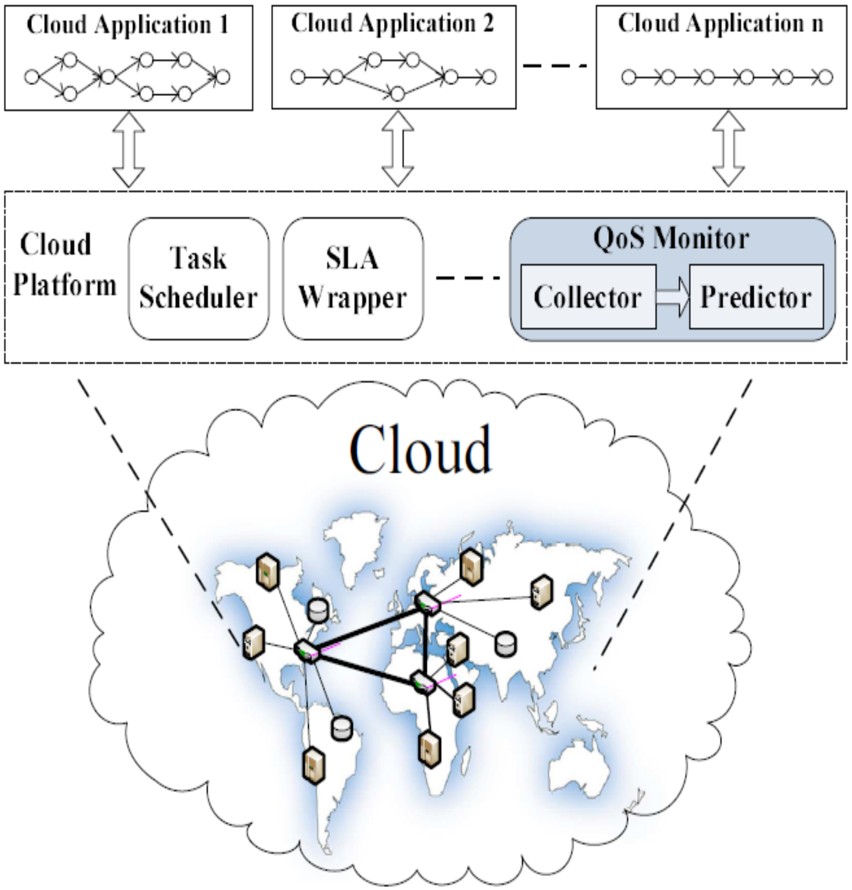
Service-Oriented-Architecture

- Service-Oriented-Architecture (SOA)
  - Distributed Web services
  - 30,000 services, 200,000 documents (seekda.com)



Performance of Services

- Service
  - Web service
  - Cloud component
- User observed performance.
  - Remote network access
  - Location
  - Invocation time



Quality-of-Service

- Quality-of-Service (QoS): non-functional performance
  - User/Time-independent QoS properties
    - price, popularity
  - User/Time-dependent QoS properties
    - failure probability, response time, throughput
- High quality applications depends on high quality services
  - Service selection, service searching, fault tolerance, service composition, etc.

Challenge 1: How to Obtain QoS?

- Conducting real-world evaluations?
- Drawbacks
  - Expensive (charge for real invocations)
  - Time-consuming (thousands of services)
  - Personalized evaluation (users' perspective)
  - Expertise (extra cost and effort)

Challenge 1: How to Obtain QoS?

- Solution: QoS Prediction (Part 1)
  - Collect users' usage experiences
  - Analyze historical QoS data
  - Predict QoS values
- Advantages
  - Economical (no additional invocation)
  - Precise (personalized QoS)
  - Effective (no extra expertise)
  - Efficient (provided as a service)

Challenge 2: How to Search Appropriate Services?

- Problems
  - Thousands of services
  - Different QoS Performance
- Solution: QoS-aware searching (Part 2)

Challenge 3: How to Build Reliable Service-Oriented Systems

- Problems
  - Services may contain various faults
  - QoS of remote services may not be stable, e.g., unavailability problem
- Solution: QoS-aware fault tolerance (Part 3)

Thesis Structure

Part 2: QoS-Aware Searching

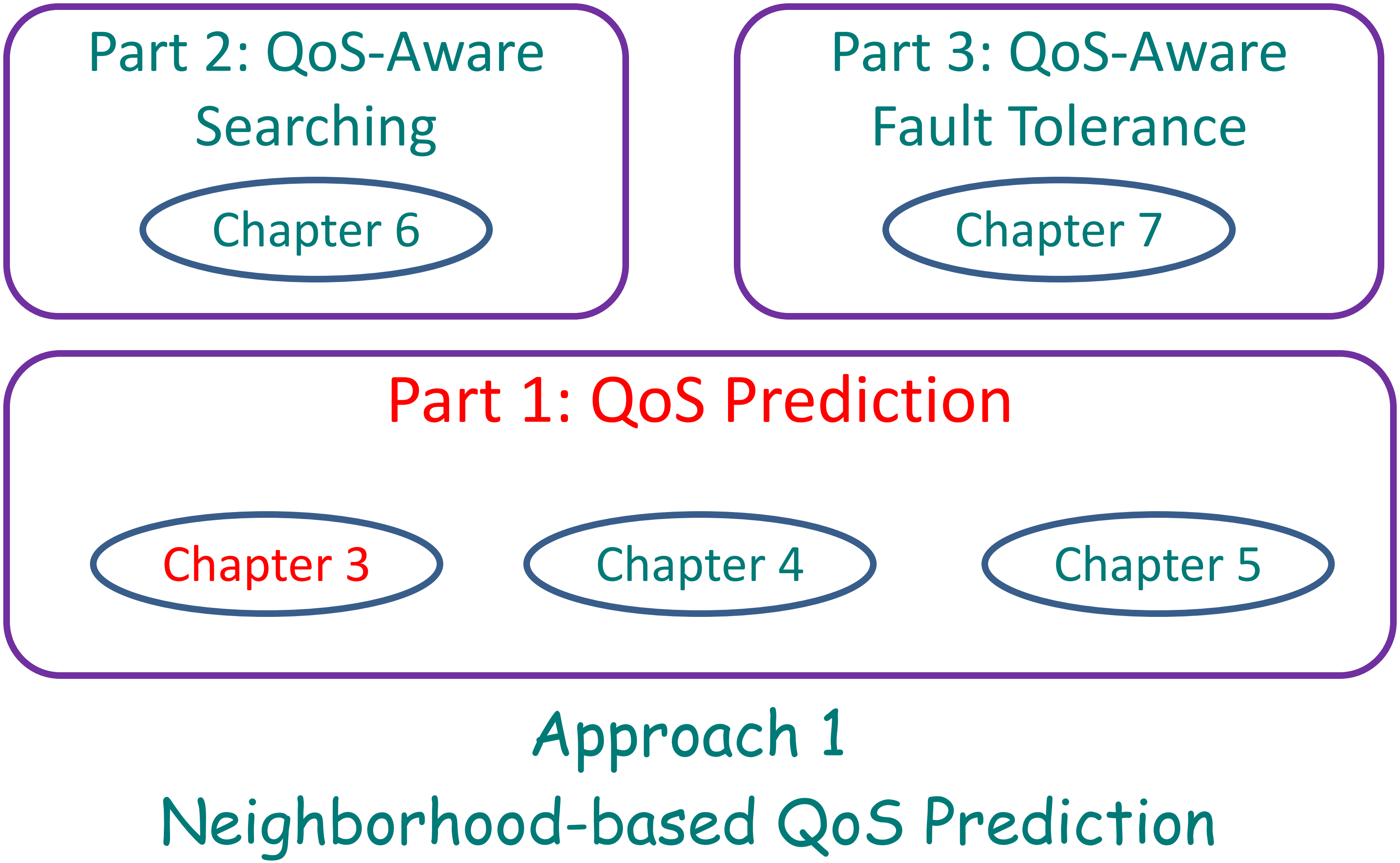
- Chapter 6

Part 3: QoS-Aware Fault Tolerance

- Chapter 7

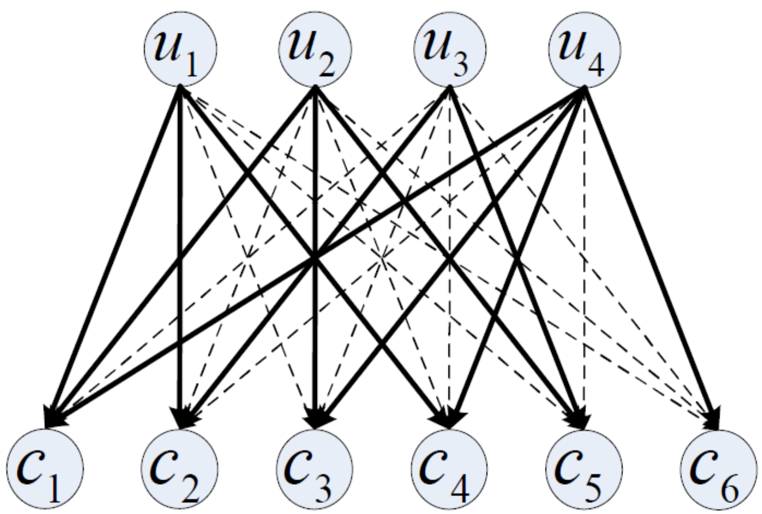
Part 1: QoS Prediction

- Chapter 3
- Chapter 4
- Chapter 5


Part 2: QoS-Aware Searching
Chapter 6
Part 3: QoS-Aware Fault Tolerance
Chapter 7
Part 1: QoS Prediction
Chapter 3
Chapter 4
Chapter 5
Approach 1
Neighborhood-based QoS Prediction

Toy Example

- User-component matrix: $m \times n$, each entry is a QoS value.
  - Sparse
  - Prediction accuracy is greatly influenced by similarity computation.



	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

Latent Features Learning

	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		$\boxed{0.37}$			0.36	
u_4	0.69		0.22	0.22		0.34

$$\begin{array}{cccc} u1 & u2 & u3 & u4 \\ \end{array}$$
$$\begin{bmatrix} 0.32 & 0.15 & 0.31 & 0.33 \\ 0.23 & 0.15 & 0.26 & 0.28 \\ 0.30 & 0.20 & 0.24 & 0.34 \\ 0.47 & 0.23 & 0.59 & 0.21 \end{bmatrix}$$

Latent-user matrix V

$$\begin{array}{cccccc} c1 & c2 & c3 & c4 & c5 & c6 \\ \end{array}$$
$$\begin{bmatrix} 0.73 & 0.35 & 0.31 & 0.26 & 0.32 & 0.42 \\ 0.60 & 0.31 & 0.27 & 0.22 & 0.28 & 0.36 \\ 0.69 & 0.37 & 0.32 & 0.27 & 0.33 & 0.45 \\ 0.95 & 0.46 & 0.42 & 0.35 & 0.41 & 0.54 \end{bmatrix}$$

Latent-component matrix H

$$W = V^T x\ H$$

Similarity Computation

- Pearson Correlation Coefficient (PCC)
- Similarity between users:

$$S(u_i,u_j)=\frac{\sum_{k=1}^{l}(v_{ik}-\overline{v}_i)(v_{jk}-\overline{v}_j)}{\sqrt{\sum_{k=1}^{l}(v_{ik}-\overline{v}_i)^2}\sqrt{\sum_{k=1}^{l}(v_{jk}-\overline{v}_j)^2}}$$

$$\begin{array}{cccc} u1 & u2 & u3 & u4 \end{array}$$
$$\begin{bmatrix} 0.32 & 0.15 & 0.31 & 0.33 \\ 0.23 & 0.15 & 0.26 & 0.28 \\ 0.30 & 0.20 & 0.24 & 0.34 \\ 0.47 & 0.23 & 0.59 & 0.21 \end{bmatrix}$$

Latent-user matrix V

- Similarity between components:

$$S(c_i,c_j)=\frac{\sum_{k=1}^{l}(h_{ik}-\overline{h}_i)(h_{jk}-\overline{h}_j)}{\sqrt{\sum_{k=1}^{l}(h_{ik}-\overline{h}_i)^2}\sqrt{\sum_{k=1}^{l}(h_{jk}-\overline{h}_j)^2}}$$

$$\begin{array}{cccccc} c1 & c2 & c3 & c4 & c5 & c6 \end{array}$$
$$\begin{bmatrix} 0.73 & 0.35 & 0.31 & 0.26 & 0.32 & 0.42 \\ 0.60 & 0.31 & 0.27 & 0.22 & 0.28 & 0.36 \\ 0.69 & 0.37 & 0.32 & 0.27 & 0.33 & 0.45 \\ 0.95 & 0.46 & 0.42 & 0.35 & 0.41 & 0.54 \end{bmatrix}$$

Latent-component matrix H

Neighbors Selection

- For every entry $w_{i,j}$ in the matrix, a set of similar users Ψ_i towards user u_i can be found by:

$$\Psi_i = \{u_k | S(u_i, u_k) > 0, rank_i(k) \leq K, k \neq i\}$$

- *A* set of similar items Φ_j towards component c_j can be found by:

$$\Phi_j = \{c_k | S(c_j, c_k) > 0, rank_p(k) \leq K, k \neq j\}$$

Missing Value Prediction

- Similar User-based:

$$w_{ij} = \overline{w}_i + \sum_{k \in \Psi_i} \frac{S(u_i, u_k)}{\sum_{a \in \Psi_i} S(u_i, u_a)} (w_{kj} - \overline{w}_k)$$

- Similar Component-based:

$$w_{ij} = \overline{w}_j + \sum_{k \in \Phi_j} \frac{S(i_j, i_k)}{\sum_{a \in \Phi_j} S(i_j, i_a)} (w_{ik} - \overline{w}_k)$$

- Hybrid:

$$w_{ij}^* = \lambda \times w_{ij}^u + (1 - \lambda) \times w_{ij}^c$$

Experiments

- QoS Dataset

STATISTICS OF WS QOS DATASET

Statistics	Response-Time	Throughput
Scale	0-20s	0-1000kbps
Mean	0.910s	47.386kbps
Num. of Users	339	339
Num. of Web Services	5,825	5,825
Num. of Records	1,974,675	1,974,675

- Metrices
  - Mean Absolute Error (MAE)
  - Root Mean Squared Error (RMSE)

$$MAE = \frac{\sum_{i,j} |w_{ij} - w^*_{ij}|}{N} \qquad RMSE = \sqrt{\frac{\sum_{i,j} (w_{ij} - w^*_{ij})^2}{N}}$$

  - w_{ij} : the real QoS value.
  - w^*_{ij} : the predicted QoS value
  - N: the number of predicted values.

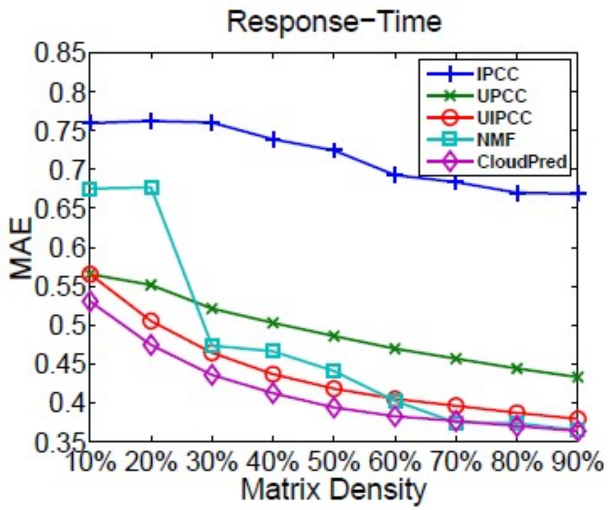
Performance Comparisons

- IPCC
  - similar item
- UPCC
  - similar user
- UIPCC
  - similar item + similar user
- NMF
  - matrix factorization
- CloudPred
  - matrix factorization + similar item + similar user

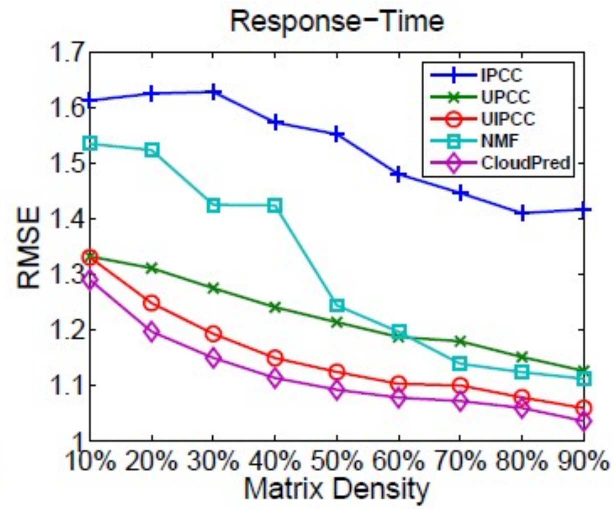
Experimental Results

Matrix Density	Metrics	Response-Time (seconds)					Throughput (kbps)				
		IPCC	UPCC	UIPCC	NMF	CloudPred	IPCC	UPCC	UIPCC	NMF	CloudPred
10%	MAE	0.759	0.565	0.565	0.675	**0.530**	31.672	26.201	22.656	19.770	**19.000**
	RMSE	1.613	1.332	1.330	1.535	**1.290**	65.522	61.965	57.465	57.376	**51.823**
20%	MAE	0.762	0.551	0.505	0.677	**0.474**	35.178	21.933	18.123	15.779	**15.420**
	RMSE	1.625	1.311	1.248	1.524	**1.197**	66.602	56.544	50.0435	50.140	**44.897**
80%	MAE	0.670	0.444	0.387	0.374	**0.370**	29.914	14.549	12.488	12.510	**10.788**
	RMSE	1.410	1.151	1.078	1.124	**1.059**	64.307	44.373	39.601	39.202	**36.850**
90%	MAE	0.668	0.433	0.379	0.364	**0.363**	29.940	13.876	12.066	11.696	**10.472**
	RMSE	1.417	1.126	1.059	1.112	**1.035**	63.714	42.553	38.076	36.755	**35.922**

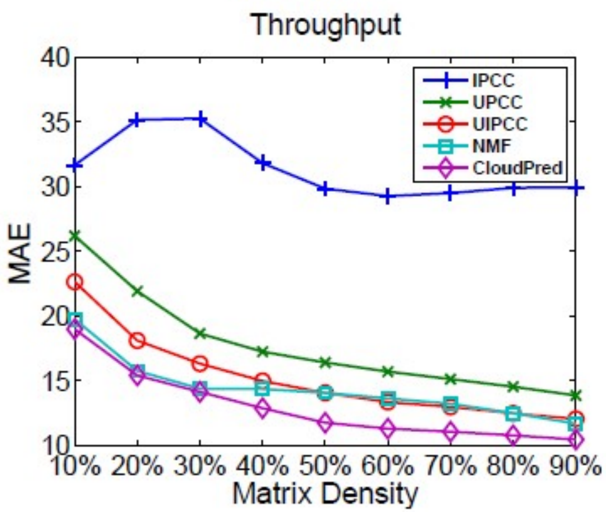
Impact of Matrix Density



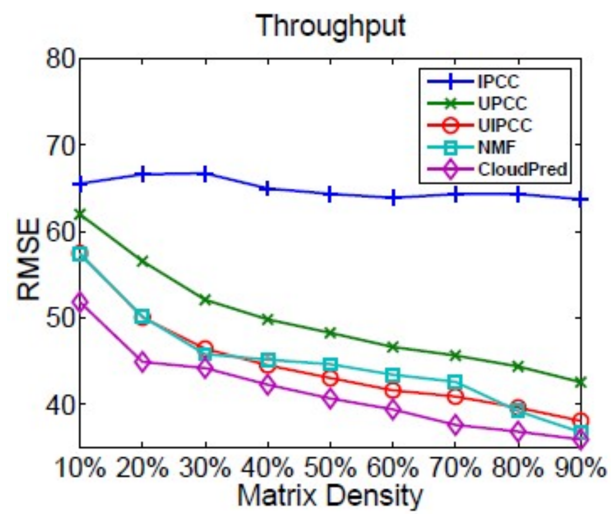
(a)



(b)

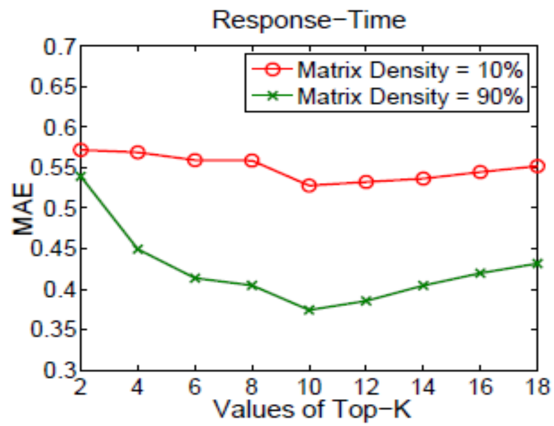


(c)

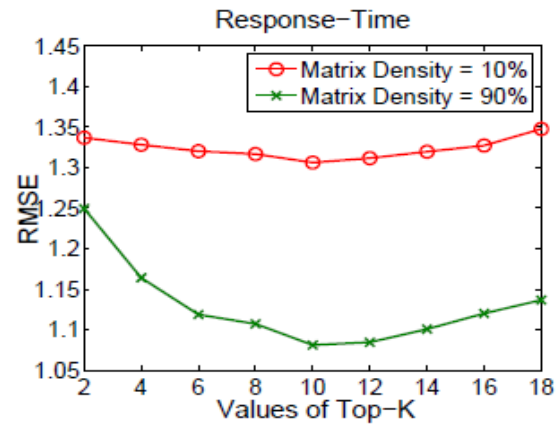


(d)

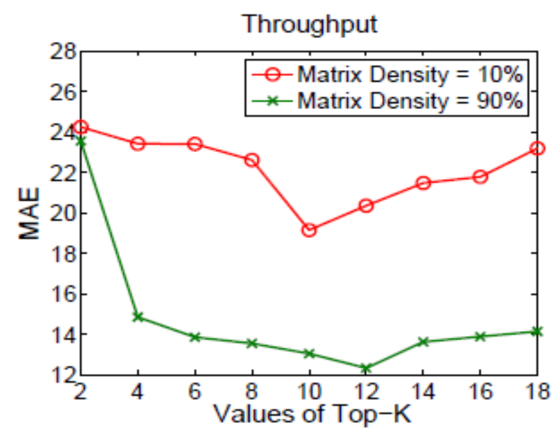
Impact of Top-K



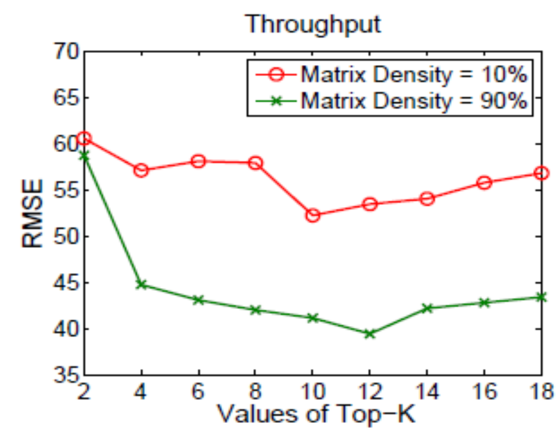
(a)



(b)

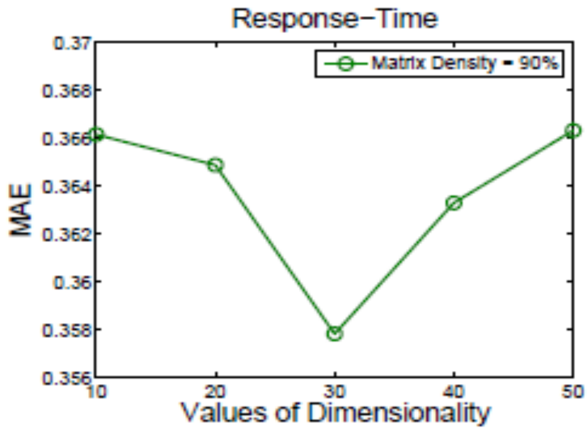


(c)

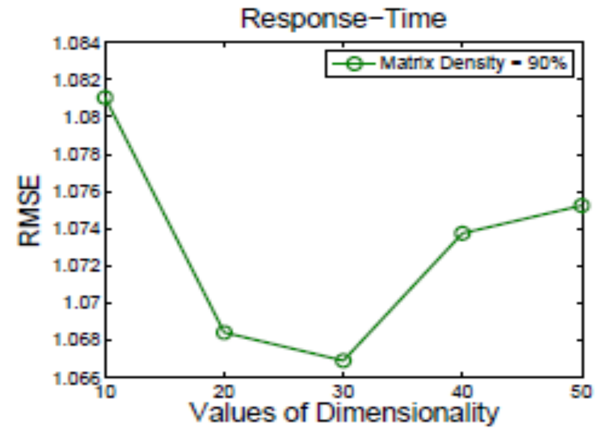


(d)

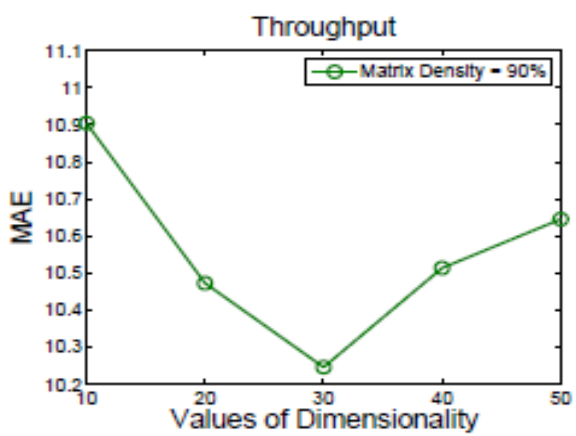
Impact of Dimensionality



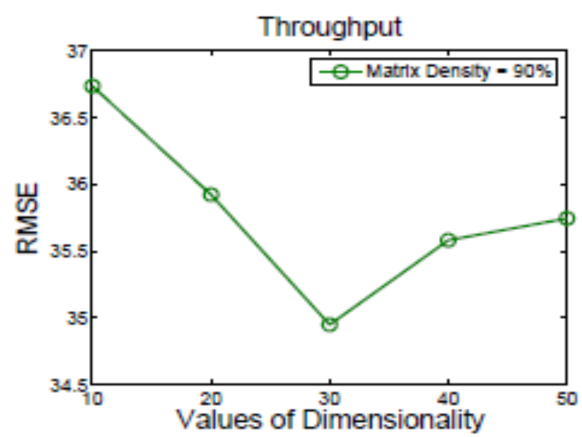
(a)



(b)

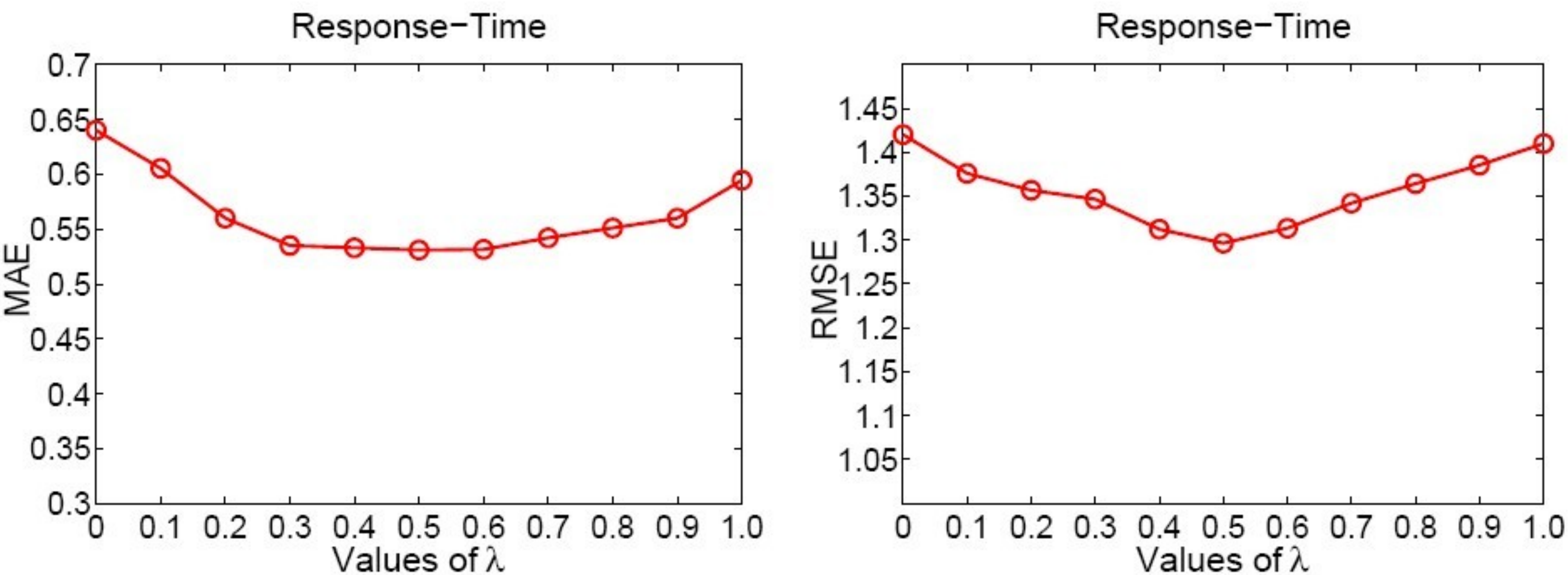


(c)

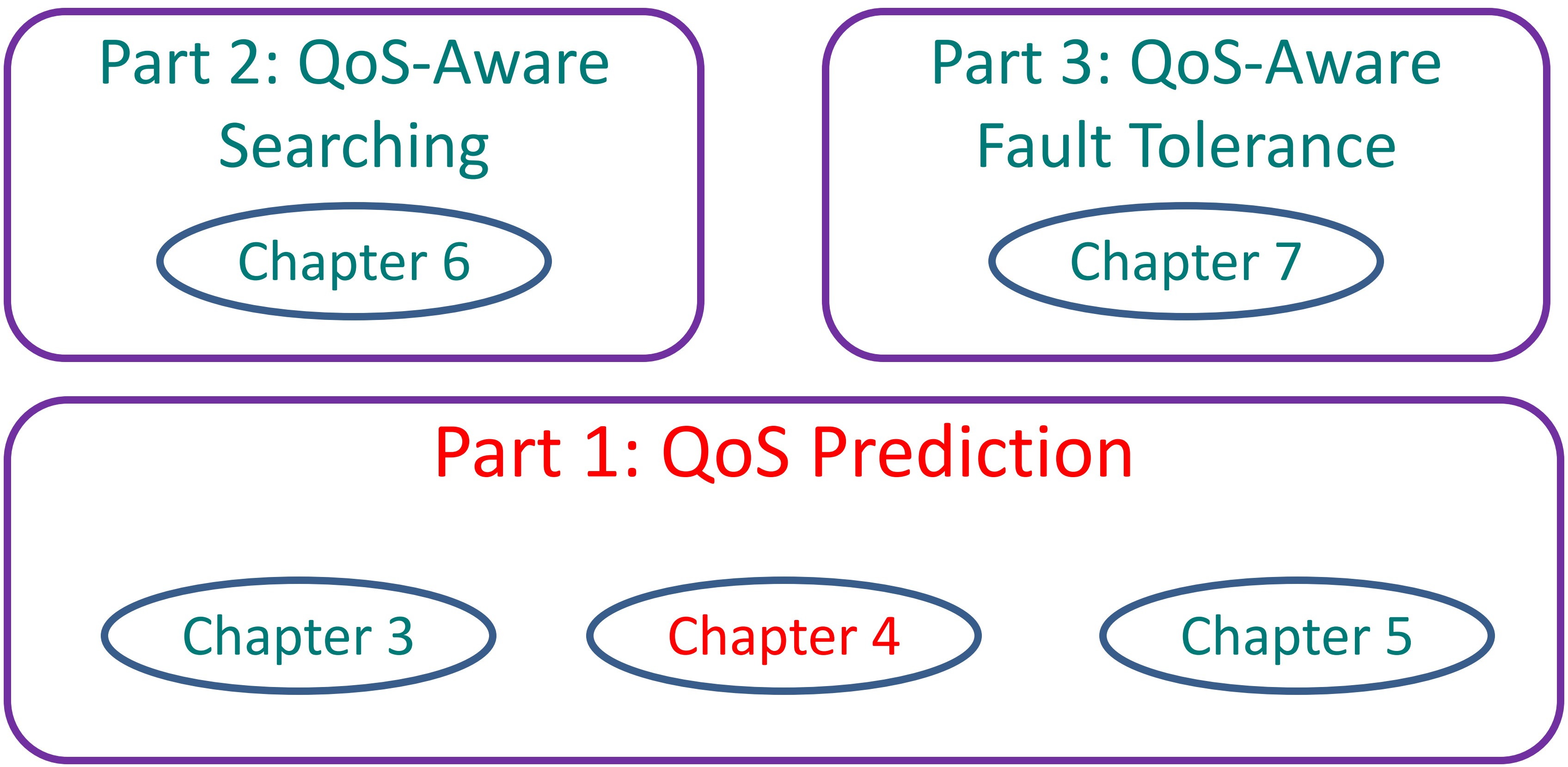


(d)

Impact of Lambda



(a) (b)



Approach 2

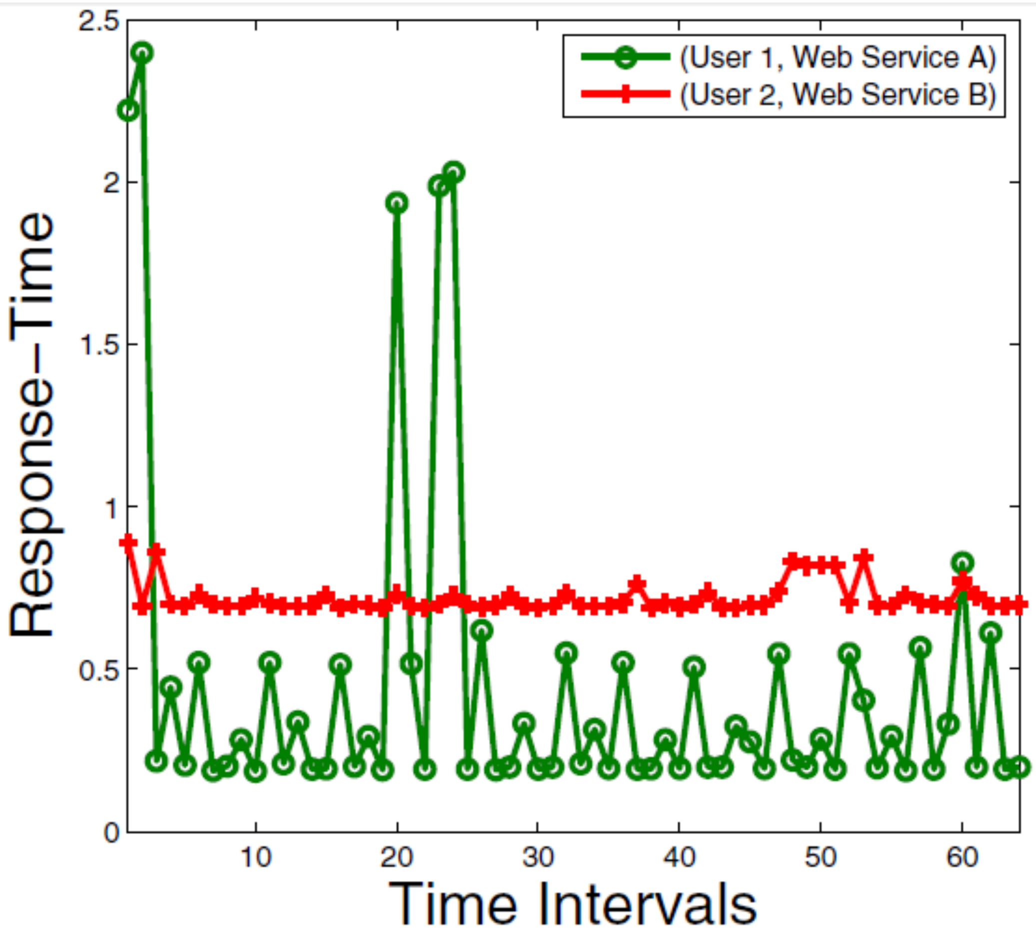
Time-Aware Model-Based QoS Prediction

Time-Aware QoS Performance

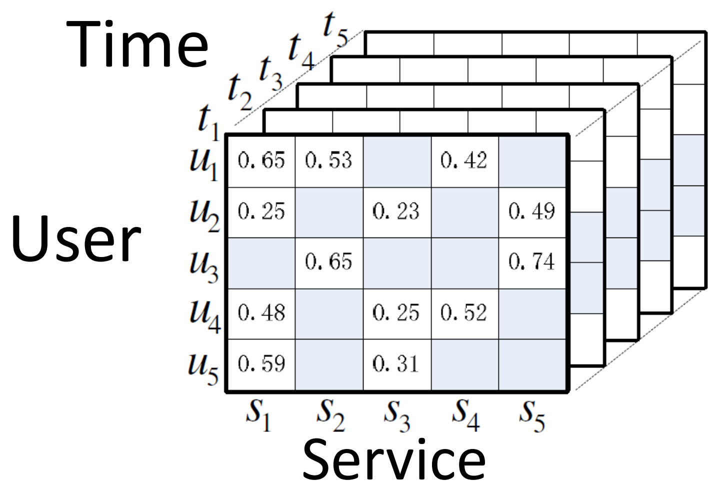
- Time-aware personalized QoS prediction is essential for:
  - Automatic selection
  - Dynamic composition

Case Study

- Periodic feature + average performance



Tensor Factorization



$$\begin{array}{ccccc} u1 & u2 & u3 & u4 & u5 \\ \end{array}$$
$$\begin{bmatrix} 0.32 & 0.15 & 0.31 & 0.33 & 0.51 \\ 0.23 & 0.15 & 0.26 & 0.28 & 0.54 \\ 0.30 & 0.20 & 0.24 & 0.34 & 0.13 \\ 0.47 & 0.23 & 0.59 & 0.21 & 0.21 \end{bmatrix}$$

latent-user matrix

$$\begin{array}{ccccc} s1 & s2 & s3 & s4 & s5 \\ \end{array}$$
$$\begin{bmatrix} 0.73 & 0.35 & 0.31 & 0.26 & 0.32 \\ 0.60 & 0.25 & 0.42 & 0.22 & 0.28 \\ 0.68 & 0.31 & 0.51 & 0.27 & 0.33 \\ 0.95 & 0.33 & 0.27 & 0.41 & 0.35 \end{bmatrix}$$

latent-service matrix

$$\begin{array}{ccccc} t1 & t2 & t3 & t4 & t5 \\ \end{array}$$
$$\begin{bmatrix} 0.84 & 0.32 & 0.26 & 0.19 & 0.13 \\ 0.26 & 0.23 & 0.74 & 0.67 & 0.25 \\ 0.29 & 0.35 & 0.28 & 0.98 & 0.55 \\ 0.94 & 0.85 & 0.49 & 0.52 & 0.57 \end{bmatrix}$$

latent-time matrix

$$\hat{Y}_{ijk} = \sum_{f=1}^{l} U_{if} S_{jf} T_{kf}.$$

Latent Features Learning

- Objective function

The error between estimated tensor and the original tensor

$$\min_{U,S,T} \mathcal{L}_{\mathcal{A}}(Y,U,S,T) = \boxed{\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{c} I_{ijk}(Y_{ijk}-\hat{Y}_{ijk})^2}$$
$$+ \boxed{\frac{\lambda_1}{2}||U||_F^2 + \frac{\lambda_2}{2}||S||_F^2 + \frac{\lambda_3}{2}||T||_F^2}$$
$$+ \boxed{\frac{\eta}{2}\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{c} I_{ijk}(\hat{Y}_{ijk}-\bar{Y}_{ij})^2,}$$

Regularization terms which constrain the norms of U, S and Ts, to avoid overfitting problem

Regularization term which preserves the predicted QoS values from varying a lot against the average QoS value

- Local optimal solution is found by incremental gradient descent

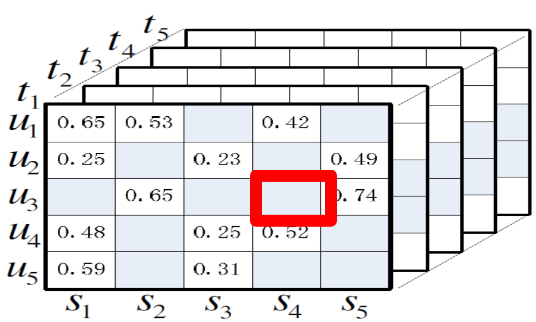
$$\frac{\partial \mathcal{L}_{\mathcal{A}}}{\partial U_{if}} = \sum_{j=1}^{n}\sum_{k=1}^{c} I_{ijk}(\hat{Y}_{ijk}-Y_{ijk})S_j^T T_k + \lambda_1 U_{if}$$
$$+ \eta\sum_{j=1}^{n}\sum_{k=1}^{c} I_{ijk}(\hat{Y}_{ijk}-\bar{Y}_{ij})S_j^T T_k,$$

Missing Value Prediction

- Given feature spaces U, S and T

$$\hat{Y}_{ijk} = I_{ijk} \sum_{f=1}^{l} U_{if} S_{jf} T_{kf}.$$

- Example:



$$\hat{Y}_{341}$$

$$\begin{array}{ccccc} u1 & u2 & u3 & u4 & u5 \\ \end{array}$$
$$\begin{bmatrix} 0.32 & 0.15 & 0.31 & 0.33 & 0.51 \\ 0.23 & 0.15 & 0.26 & 0.28 & 0.54 \\ 0.30 & 0.20 & 0.24 & 0.34 & 0.13 \\ 0.47 & 0.23 & 0.59 & 0.21 & 0.21 \end{bmatrix}$$

latent-user matrix

$$\begin{array}{ccccc} s1 & s2 & s3 & s4 & s5 \\ \end{array}$$
$$\begin{bmatrix} 0.73 & 0.35 & 0.31 & 0.26 & 0.32 \\ 0.60 & 0.25 & 0.42 & 0.22 & 0.28 \\ 0.68 & 0.31 & 0.51 & 0.27 & 0.33 \\ 0.95 & 0.33 & 0.27 & 0.41 & 0.35 \end{bmatrix}$$

latent-service matrix

$$\begin{array}{ccccc} t1 & t2 & t3 & t4 & t5 \\ \end{array}$$
$$\begin{bmatrix} 0.84 & 0.32 & 0.26 & 0.19 & 0.13 \\ 0.26 & 0.23 & 0.74 & 0.67 & 0.25 \\ 0.29 & 0.35 & 0.28 & 0.98 & 0.55 \\ 0.94 & 0.85 & 0.49 & 0.52 & 0.57 \end{bmatrix}$$

latent-time matrix

Experiments

- Time-Aware Web Service QoS Dataset

STATISTICS OF WS QOS DATASET

Statistics	Response-Time	Throughput
Scale	0-20s	0-1000kbps
Mean	3.165s	9.609kbps
Num. of Users	142	142
Num. of Web Services	4,532	4,532
Num. of Time Intervals	64	64
Num. of Records	30,287,611	30,287,611

Performance Comparisons

- Matrix Factorization extended methods
  - MF1: a set of user-service matrix slices in terms of time
  - MF2: compresses the user-service-time tensor into an user-service matrix
- Tensor Factorization methods
  - TF :Tensor factorization-based prediction method.
  - WSPred :Tensor factorization-based prediction method with average QoS value constraints.

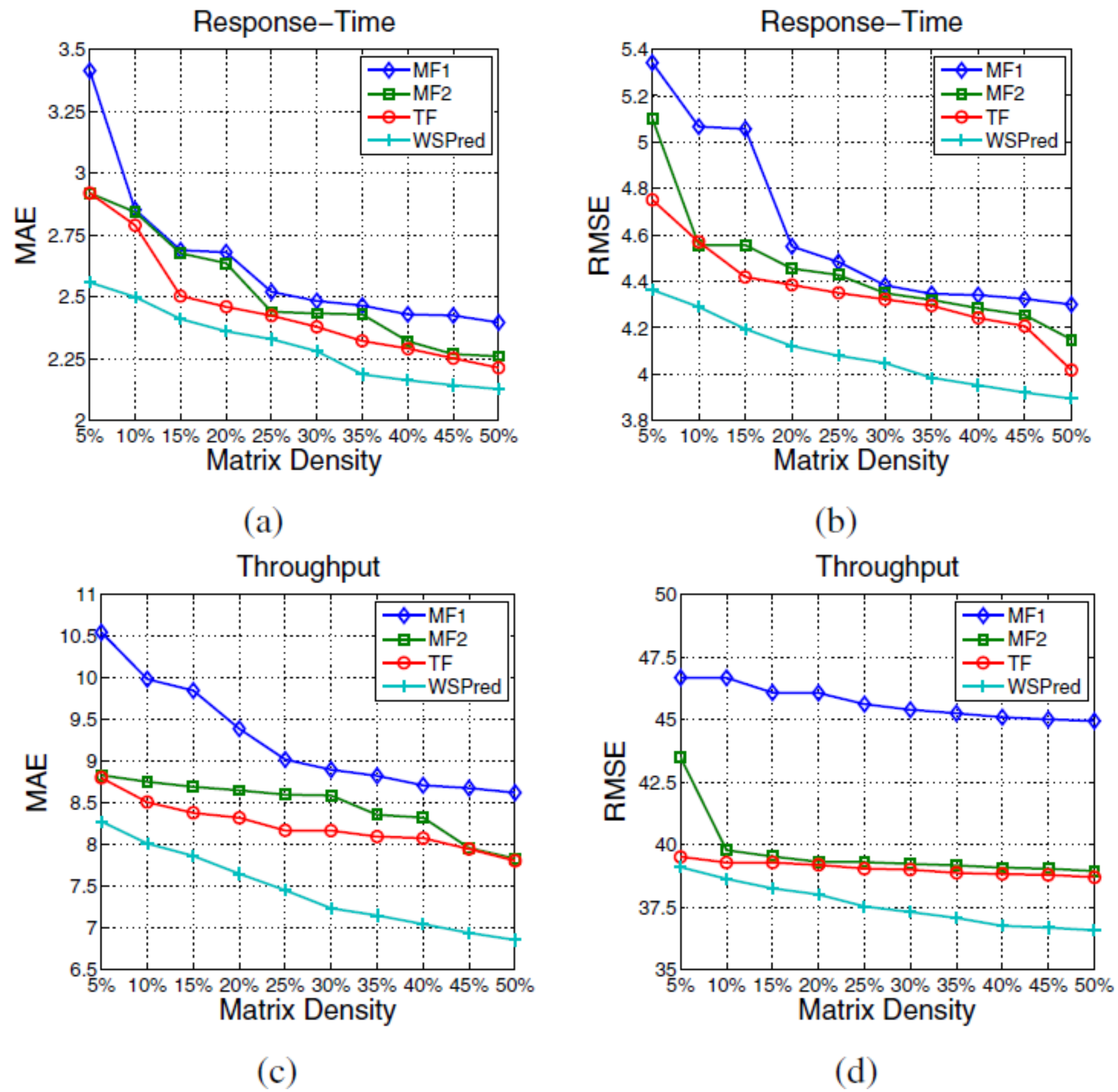
Experimental Results

- A smaller MAE or RMSE value means a better performance

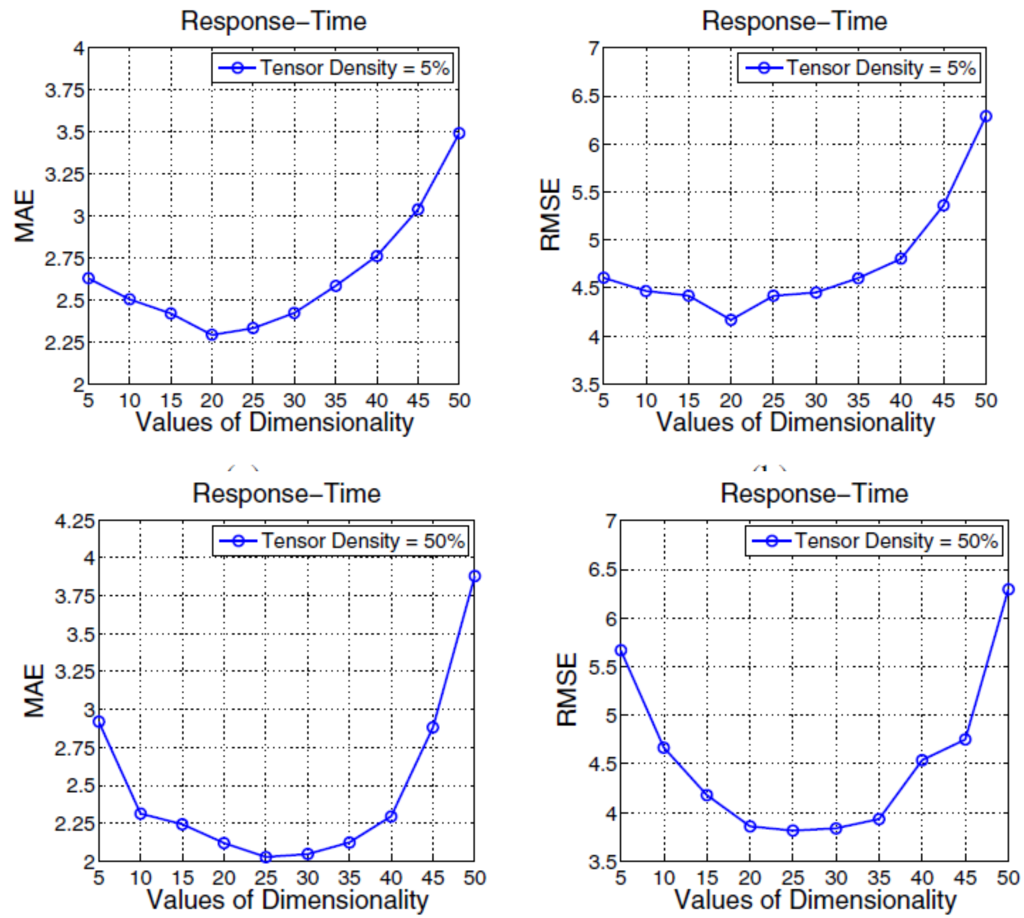
Tensor Density	Metrics	Response-Time (seconds)				Throughput (kbps)			
		MF1	MF2	TF	WSPred	MF1	MF2	TF	WSPred
5%	MAE	3.4137	2.9187	2.9184	**2.5580**	10.5460	8.8317	8.7997	**8.2761**
	RMSE	5.3423	5.1024	4.7508	**4.3626**	46.6735	43.4769	39.5133	**39.0962**
10%	MAE	2.8518	2.8421	2.7888	**2.4990**	9.9839	8.7522	8.5080	**8.0131**
	RMSE	5.0667	4.5563	4.5696	**4.2892**	46.6656	39.7740	39.2792	**38.6251**
45%	MAE	2.4241	2.2679	2.2511	**2.1462**	8.6773	7.9590	7.9471	**6.9398**
	RMSE	4.3240	4.2541	4.2071	**3.9200**	45.0077	39.9388	38.6964	**36.5724**
50%	MAE	2.3959	2.2596	2.2127	**2.1266**	8.6224	7.8306	7.8045	**6.8558**
	RMSE	4.2996	4.1490	4.0169	**3.8943**	44.9407	38.9388	38.6964	**36.5724**
		9~25%	5~15%	3~12%		16~22%	3~13%	1~12%	

Performance improvement of WSPred

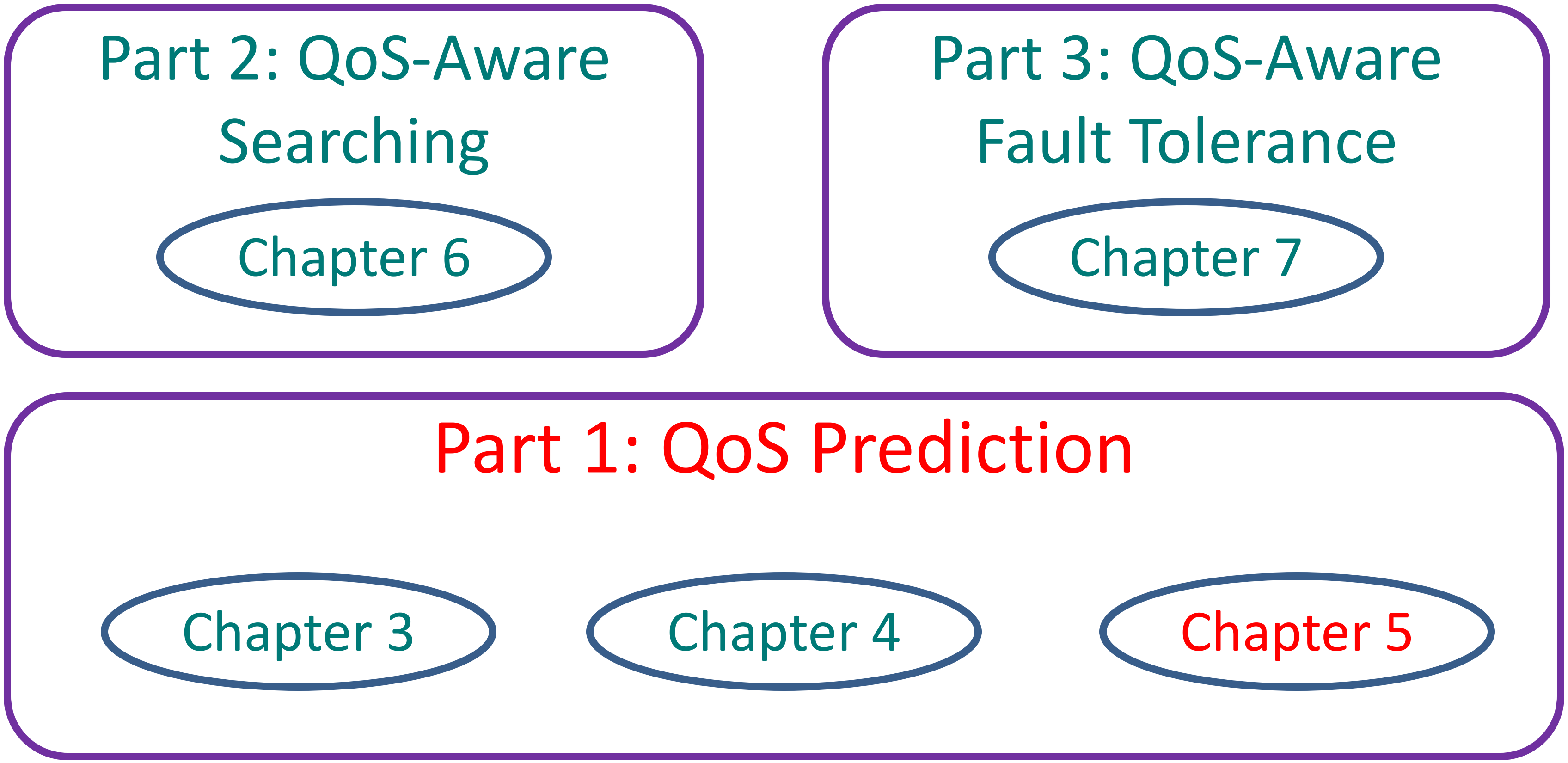
Impact of Tensor Density



Impact of Dimensionality

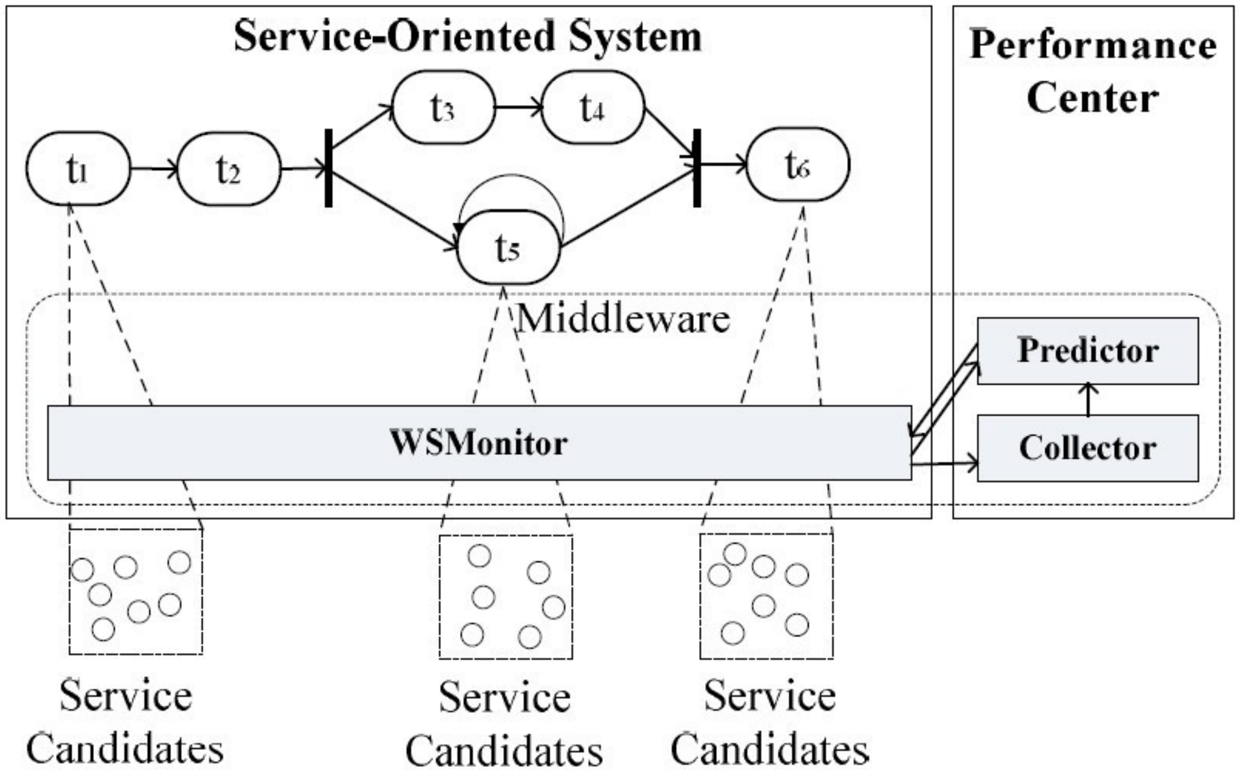


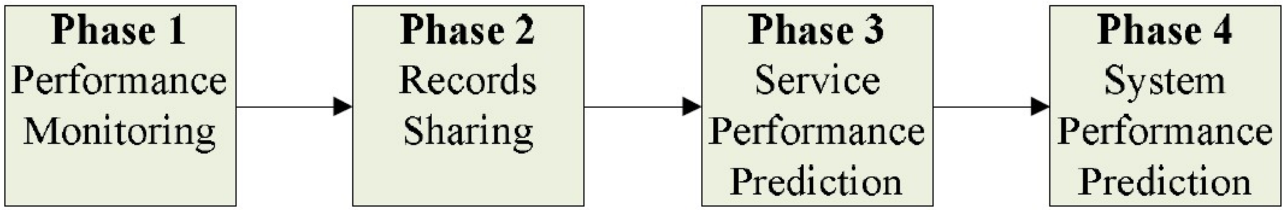
(c) (d)



Approach 3
Online QoS Prediction

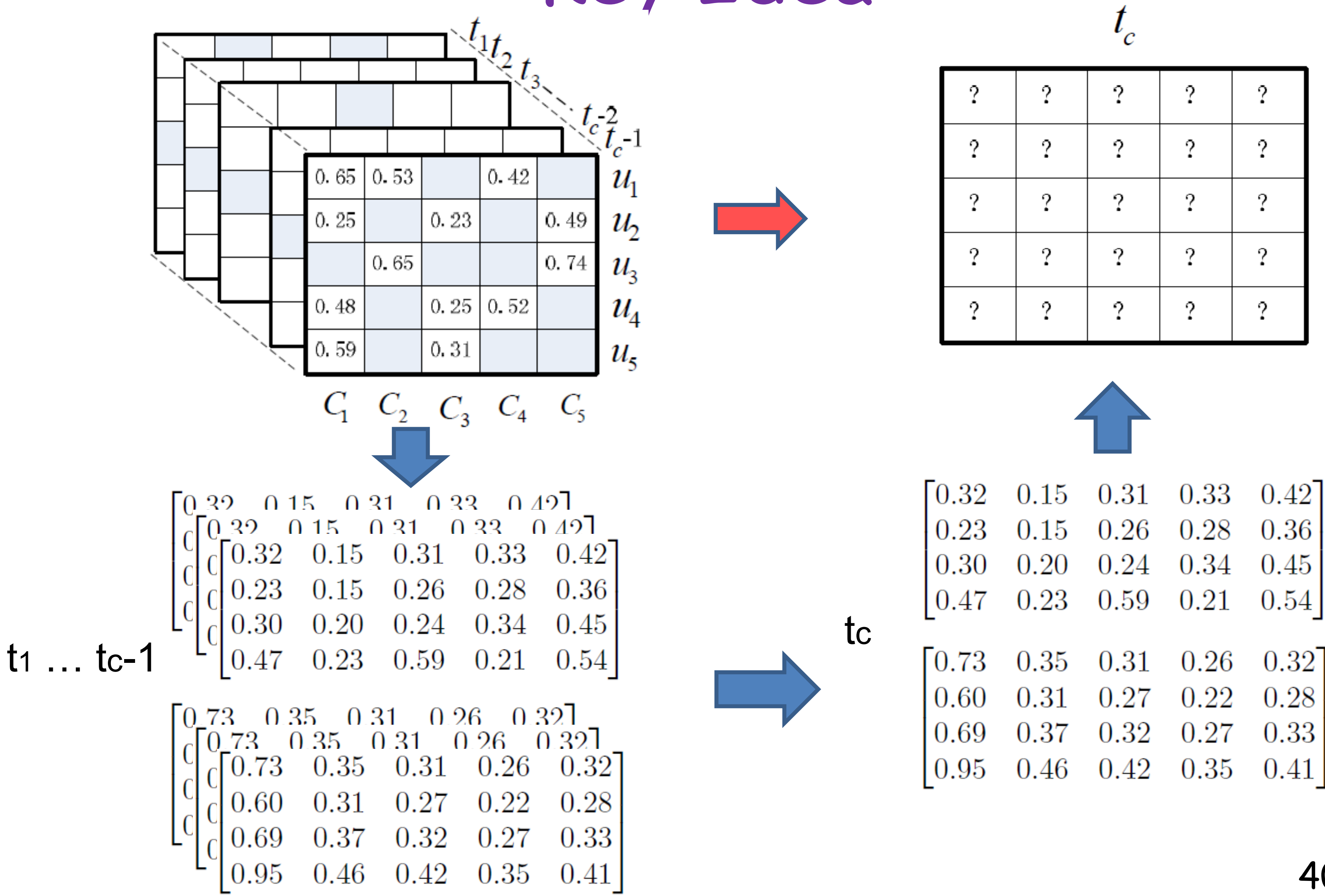
System Architecture





Online QoS Prediction Procedures

Key Idea



t_1 t_2 t_3 t_c-2 t_c-1
0.65 0.53 0.42
0.25 0.23 0.49
0.65 0.74
0.48 0.25 0.52
0.59 0.31
u_1 u_2 u_3 u_4 u_5
C_1 C_2 C_3 C_4 C_5
t_c
? ? ? ? ?
? ? ? ? ?
? ? ? ? ?
? ? ? ? ?
? ? ? ? ?
t1 … tc-1
0.32 0.15 0.31 0.33 0.42
0.23 0.15 0.26 0.28 0.36
0.30 0.20 0.24 0.34 0.45
0.47 0.23 0.59 0.21 0.54
0.73 0.35 0.31 0.26 0.32
0.60 0.31 0.27 0.22 0.28
0.69 0.37 0.32 0.27 0.33
0.95 0.46 0.42 0.35 0.41
tc
0.32 0.15 0.31 0.33 0.42
0.23 0.15 0.26 0.28 0.36
0.30 0.20 0.24 0.34 0.45
0.47 0.23 0.59 0.21 0.54
0.73 0.35 0.31 0.26 0.32
0.60 0.31 0.27 0.22 0.28
0.69 0.37 0.32 0.27 0.33
0.95 0.46 0.42 0.35 0.41

Step 1: Time-Aware Latent Feature Learning

- Objective function:

$$\min \mathcal{L}(p_u(t), p_i(t))$$
$$= \boxed{\frac{1}{2}\sum_{u=1}^{m}\sum_{i=1}^{n} I_{ui}(r_{ui}(t) - g(\hat{r}_{ui}(t)))^2}$$
$$+ \boxed{\frac{\lambda_1}{2}||p(t)||^2 + \frac{\lambda_2}{2}||q(t)||^2,}$$

The error between estimated matrix and the original matrix

Regularization terms which constrain the norms of p(t) and q(t), to avoid overfitting problem

Algorithm 4: Time-Aware Latent Features Learning

Input: $R(t), l, \lambda_1, \lambda_2$
Output: $p(t), q(t)$

Initialize $p(t) \in \mathbb{R}^{l \times m}$ and $q(t) \in \mathbb{R}^{l \times n}$ with small random numbers;
Load the performance records from matrix $R(t)$;
Calculate the objective function value $\mathcal{L}(p_u(t), q_i(t))$ by Eq. (5.1) and Eq. (5.2);
repeat
    Calculate the gradient of feature vectors $\frac{\partial L}{p_u(t)}$ and $\frac{\partial L}{q_i(t)}$ according Eq. (5.3) and Eq. (5.4), respectively;
    Update the latent user and service feature matrices $p(t)$ and $q(t)$;
    $p_u(t) \leftarrow p_u(t) - \frac{\partial L}{p_u(t)}$;
    $q_i(t) \leftarrow q_i(t) - \frac{\partial L}{q_i(t)}$;
    Update the objective function value $\mathcal{L}(p_u(t), p_i(t))$ by Eq. (5.1) and Eq. (5.2);
until *Converge* ;

Step 1: Time-Aware Latent Feature Learning

- Iterative Process :
  - gradient descent

$$
\begin{aligned}
\frac{\partial L}{p_u(t)} &= I_{ui}(g(\hat{r}_{ui}(t)) - r_{ui}(t))g'(\hat{r}_{ui}(t))q_i(t) \\
&\quad +\lambda_1 p_u(t), \\
\frac{\partial L}{q_i(t)} &= I_{ui}(g(\hat{r}_{ui}(t)) - r_{ui}(t))g'(\hat{r}_{ui}(t))p_u(t) \\
&\quad +\lambda_2 q_i(t).
\end{aligned}
$$

Algorithm 4: Time-Aware Latent Features Learning

Input: $R(t), l, \lambda_1, \lambda_2$

Output: $p(t), q(t)$

Initialize $p(t) \in \mathbb{R}^{l \times m}$ and $q(t) \in \mathbb{R}^{l \times n}$ with small random numbers;
Load the performance records from matrix $R(t)$;
Calculate the objective function value $\mathcal{L}(p_u(t), q_i(t))$ by Eq. (5.1) and Eq. (5.2);
repeat
    Calculate the gradient of feature vectors $\frac{\partial L}{p_u(t)}$ and $\frac{\partial L}{q_i(t)}$ according Eq. (5.3) and Eq. (5.4), respectively;
    Update the latent user and service feature matrices $p(t)$ and $q(t)$;
    $p_u(t) \leftarrow p_u(t) - \frac{\partial L}{p_u(t)}$;
    $q_i(t) \leftarrow q_i(t) - \frac{\partial L}{q_i(t)}$;
    Update the objective function value $\mathcal{L}(p_u(t), p_i(t))$ by Eq. (5.1) and Eq. (5.2);
until *Converge* ;

Step 2 & Step 3 (Offline Phase)

$$\hat{p}_u(t_c) = \frac{\sum_{k=1}^{w} p_u(t_c - k) f(k)}{\sum_{k=1}^{w} f(k)},$$

$$\hat{q}_i(t_c) = \frac{\sum_{k=1}^{w} q_i(t_c - k) f(k)}{\sum_{k=1}^{w} f(k)},$$

$$f(k) = e^{-\alpha k}$$

$$\hat{r}_{ui}(t_c) = \hat{p}_u^T(t_c)\hat{q}_i(t_c)$$

$$\begin{array}{c}
\begin{array}{ccccc} \text{u1} & \text{u2} & \text{u3} & \text{u4} & \text{u5} \end{array} \\
\begin{bmatrix} 0.32 & 0.15 & 0.31 & 0.33 & 0.42 \\ 0.23 & 0.15 & 0.26 & 0.28 & 0.36 \\ 0.30 & 0.20 & 0.24 & 0.34 & 0.45 \\ 0.47 & 0.23 & 0.59 & 0.21 & 0.54 \end{bmatrix}
\end{array}$$

$$\begin{array}{c}
\begin{array}{ccccc} \text{c1} & \text{c2} & \text{c3} & \text{c4} & \text{c5} \end{array} \\
\begin{bmatrix} 0.73 & 0.35 & 0.31 & 0.26 & 0.32 \\ 0.60 & 0.31 & 0.27 & 0.22 & 0.28 \\ 0.69 & 0.37 & 0.32 & 0.27 & 0.33 \\ 0.95 & 0.46 & 0.42 & 0.35 & 0.41 \end{bmatrix}
\end{array}$$

t_c

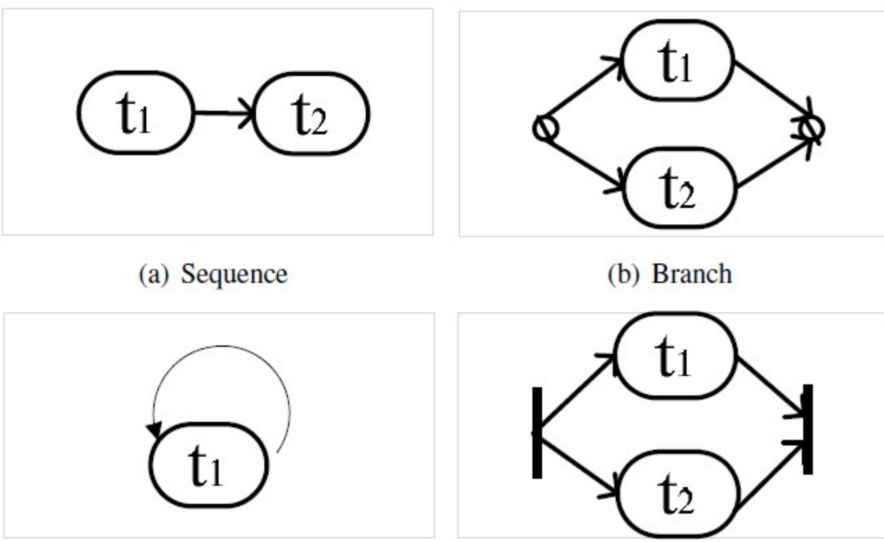
?	?	?	?	?
?	?	?	?	?
?	?	?	?	?
?	?	?	?	?
?	?	?	?	?

Step 2 & Step 3 (Online Phase)

$$\hat{p}_u(t_c) = e^{-\alpha}\Big(\frac{\boxed{p_u(t_{c-1})}}{\sum_{k=1}^{w} f(k)} + \boxed{\hat{p}_u(t_{c-1})} - \frac{\boxed{p_u(t_{c-1-w})} f(w)}{\sum_{k=1}^{w} f(k)}\Big),$$

$$\hat{q}_i(t_c) = e^{-\alpha}\Big(\frac{q_i(t_{c-1})}{\sum_{k=1}^{w} f(k)} + \hat{q}_i(t_{c-1}) - \frac{q_i(t_{c-1-w}) f(w)}{\sum_{k=1}^{w} f(k)}\Big),$$

System Level Performance



(a) Sequence (b) Branch

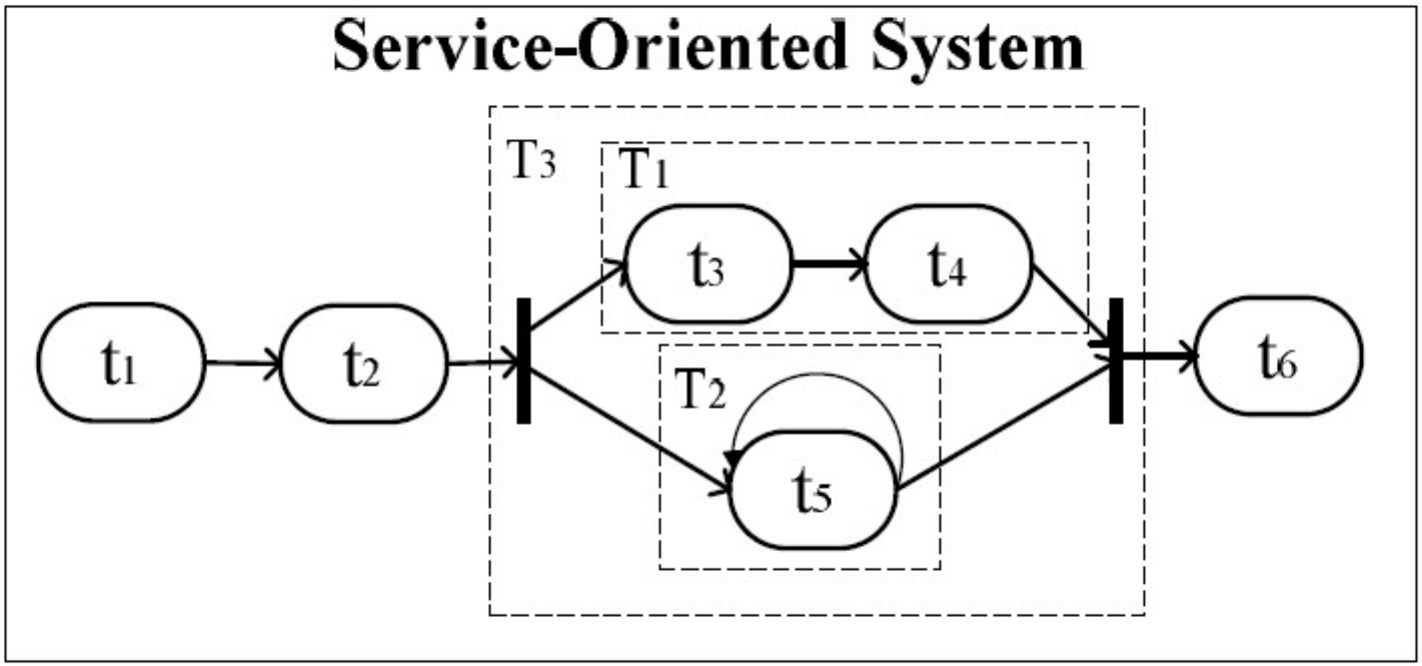
(c) Loop (d) Parallel

Basic Compositional Structures

Calculation of Aggregated Response Time

Structure	Calculation Method	Meaning of Notation
Sequence	$r = \sum_{i=1}^{n} r_i$	n: number of sequential sub-tasks r_i: response time of the i^{th} sub-task
Branch	$r = \sum_{i=1}^{n} p_i r_i$	n: number of branches r_i: response time of the i^{th} branch p_i: probability of the i^{th} branch to be executed
Loop	$r = \sum_{i=1}^{n} p_i r_i i$	n: maximum looping times r_i: response time of the i^{th} sub-task p_i: probability of executing the sub-task for i times
Parallel	$r = \max_{i=1}^{n} r_i$	n: number of branches r_i: response time of the i^{th} branch

System Level Performance Prediction



Comparison with Other Methods

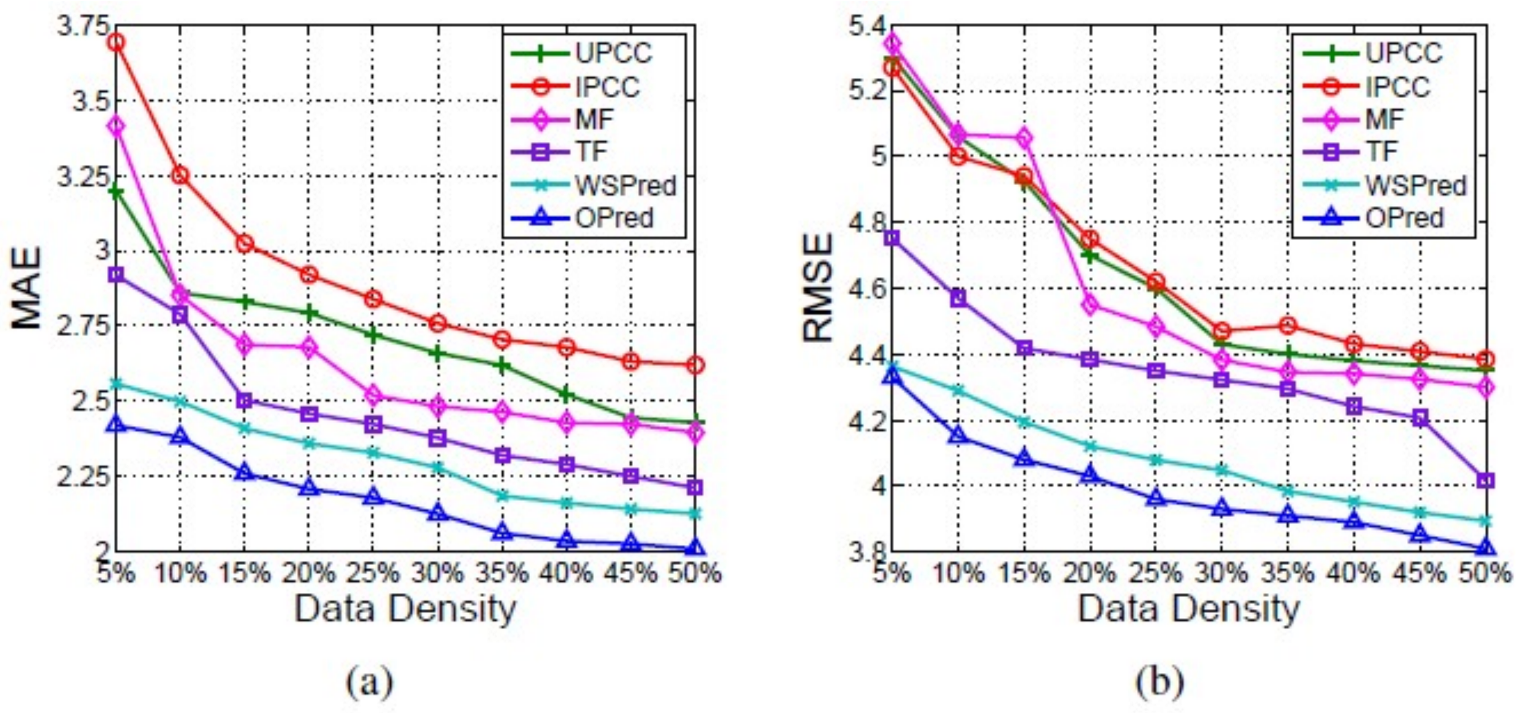
- UPCC (time-insensitive) - Mean
- IPCC (time-insensitive) - Mean
- MF (time-insensitive) - Mean
- TF (time-sensitive) - Periodic
- WSPred (time-sensitive) – Periodic + Mean
- OPred (time-sensitive) - Periodic + Mean + timely trend

Experimental Results

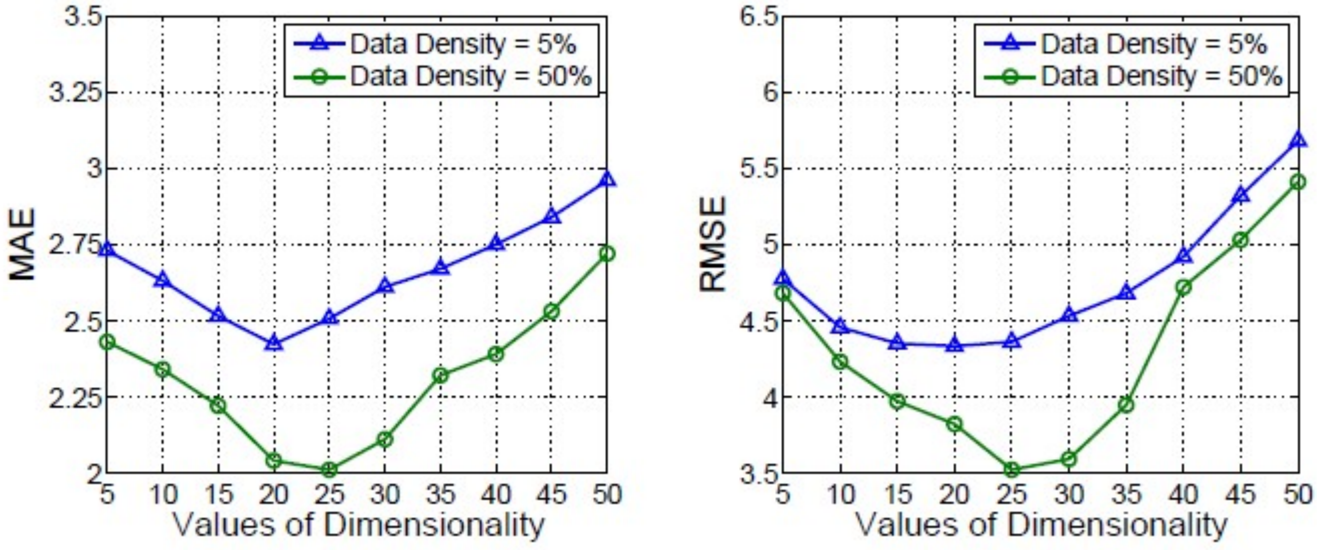
Data Density	RMSE	Response Time (seconds)					
		UPCC	IPCC	MF	TF	WSPred	OPred
5%	Mean	5.312	5.289	5.329	4.751	4.362	**4.330**
	Best	5.263	5.276	5.321	4.747	4.358	**4.327**
10%	Mean	5.043	4.972	5.079	4.567	4.287	**4.151**
	Best	4.962	4.946	5.063	4.563	4.283	**4.148**
45%	Mean	4.425	4.371	4.337	4.208	3.923	**3.855**
	Best	4.388	4.342	4.318	4.202	3.918	**3.851**
50%	Mean	4.352	4.354	4.298	4.016	3.899	**3.809**
	Best	4.331	4.336	4.274	4.012	3.894	**3.808**

Data Density	MAE	Response Time (seconds)					
		UPCC	IPCC	MF	TF	WSPred	OPred
5%	Mean	3.720	3.213	3.387	2.915	2.559	**2.417**
	Best	3.687	3.207	3.381	2.911	2.555	**2.413**
10%	Mean	3.264	2.841	2.873	2.786	2.495	**2.376**
	Best	3.243	2.812	2.851	2.782	2.488	**2.374**
45%	Mean	2.627	2.455	2.436	2.253	2.141	**2.029**
	Best	2.613	2.431	2.423	2.247	2.137	**2.026**
50%	Mean	2.619	2.417	2.391	2.211	2.130	**2.011**
	Best	2.609	2.404	2.384	2.207	2.125	**2.008**

Impact of Density

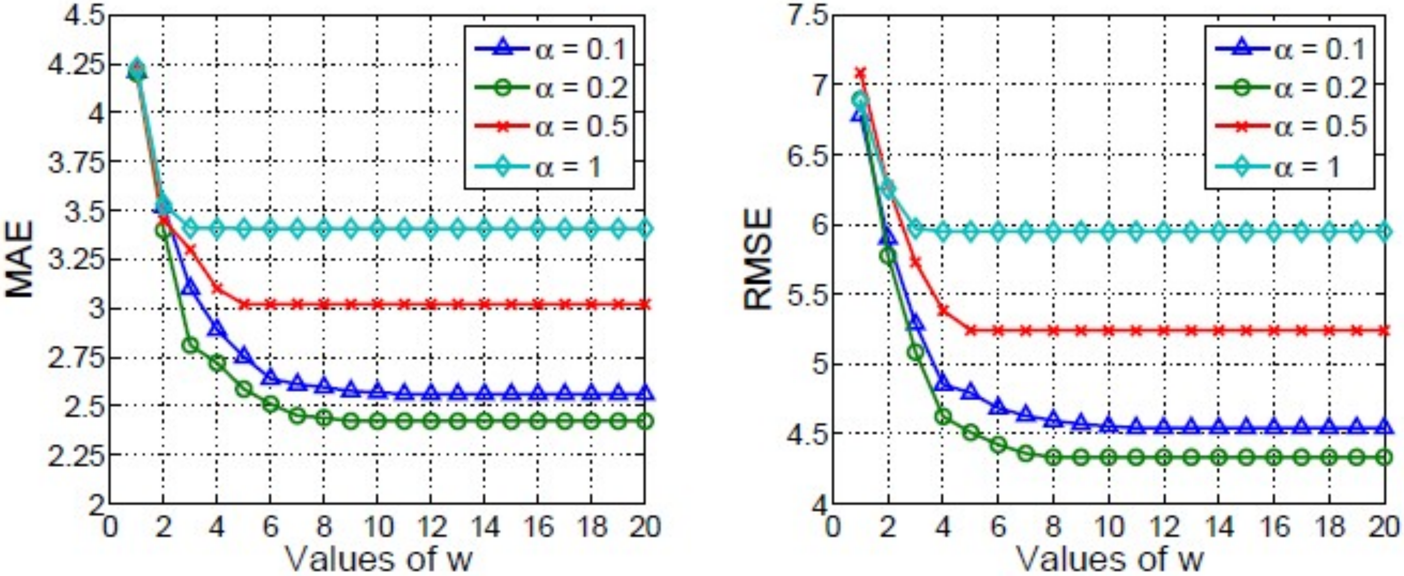


Impact of Dimensionality



(a) (b)

Impact of a and w

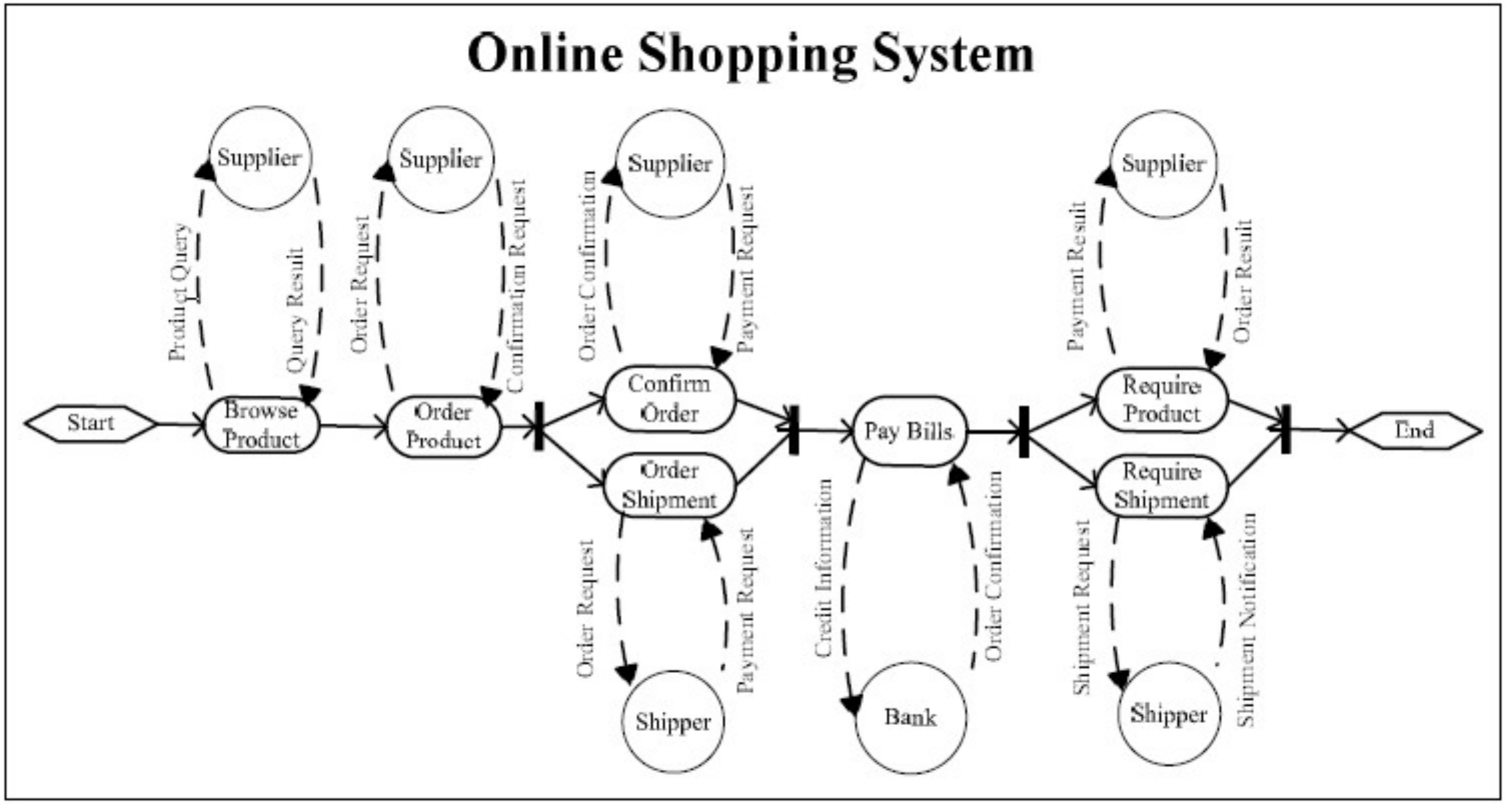


(a) (b)

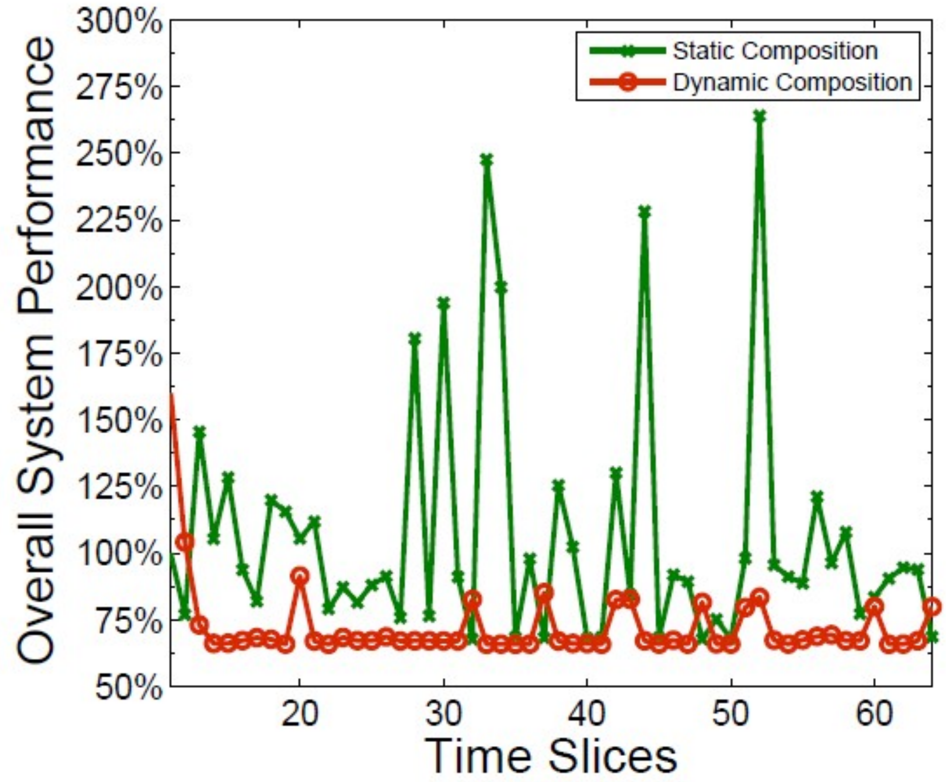
Average Computational Time

Approach	Computational Time	Percentage of A Time Slice
UPCC	10.095m	67.3%
IPCC	9.735m	64.9%
MF	1.575m	10.5%
TF	1.860m	12.4%
WSPred	2.055m	13.7%
OPred	0.240m	1.6%

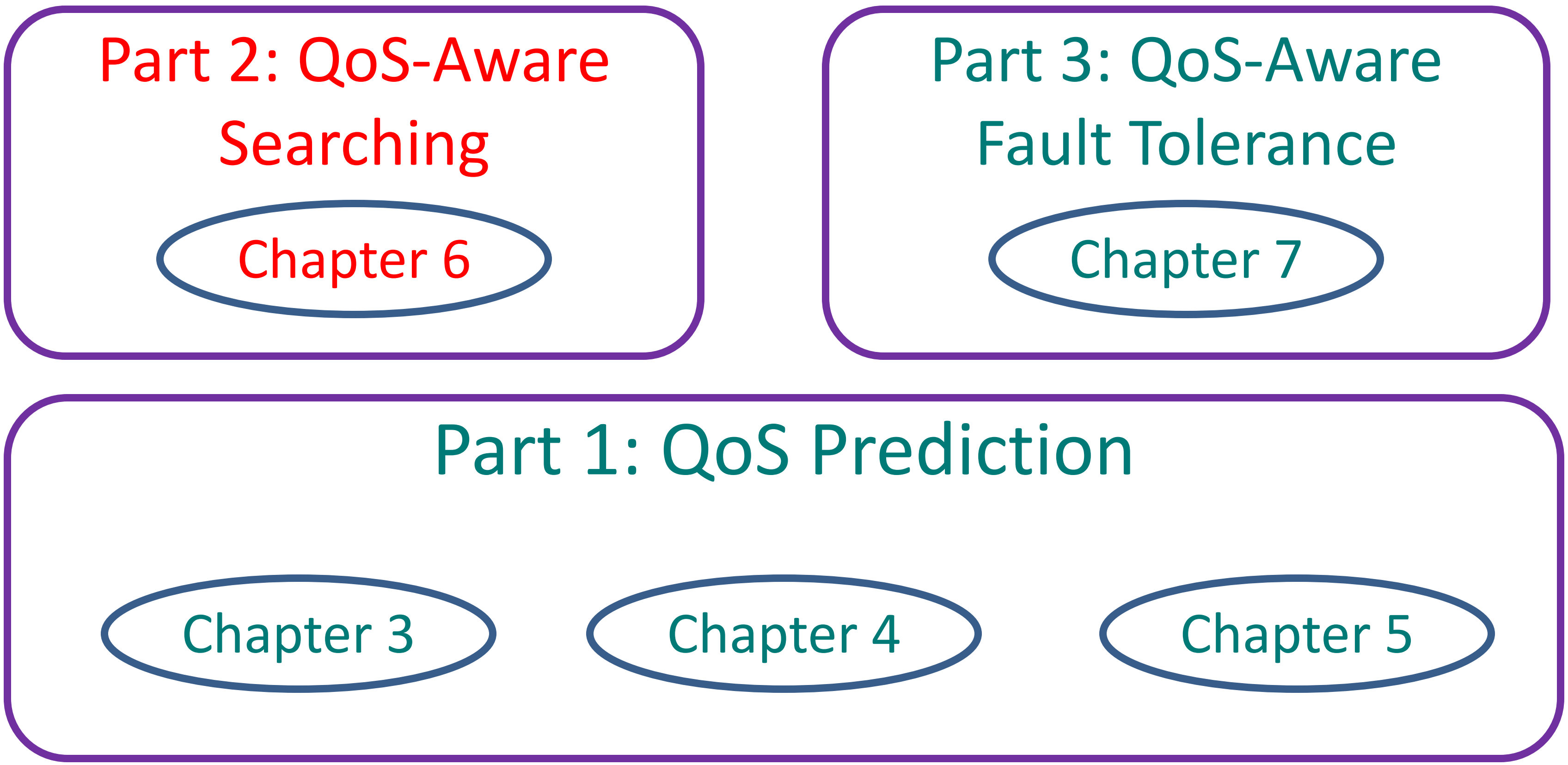
System Level Performance Case Study



System Level Performance Case Study



System Performance Improvement of Dynamic Service Composition

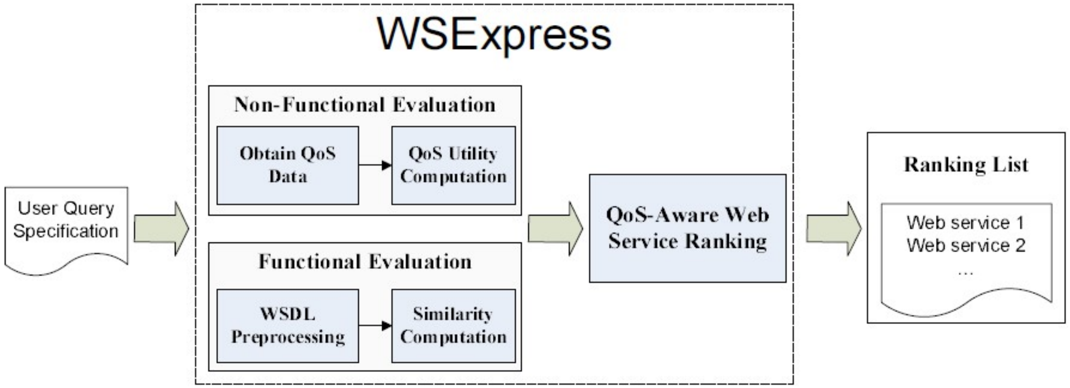

Part 2: QoS-Aware Searching
Chapter 6
Part 3: QoS-Aware Fault Tolerance
Chapter 7
Part 1: QoS Prediction
Chapter 3
Chapter 4
Chapter 5

Problems

- How to find a service? → Functionality
  - Many Web services
- How to find the best one? → QoS
  - Different QoS Performance

WSExpress

- Functional attributes & non-functional features.
  - Non-functional evaluation
    - Obtains QoS criteria values
    - QoS utility computation
  - Functional evaluation
    - WSDL preprocessing
    - Similarity computation



Combination

$$r_i = \lambda \cdot \frac{1}{\log(p_{s_i} + 1)} + (1 - \lambda) \cdot \frac{1}{\log(p_{u_i} + 1)}$$

$$\lambda \in [0, 1]$$

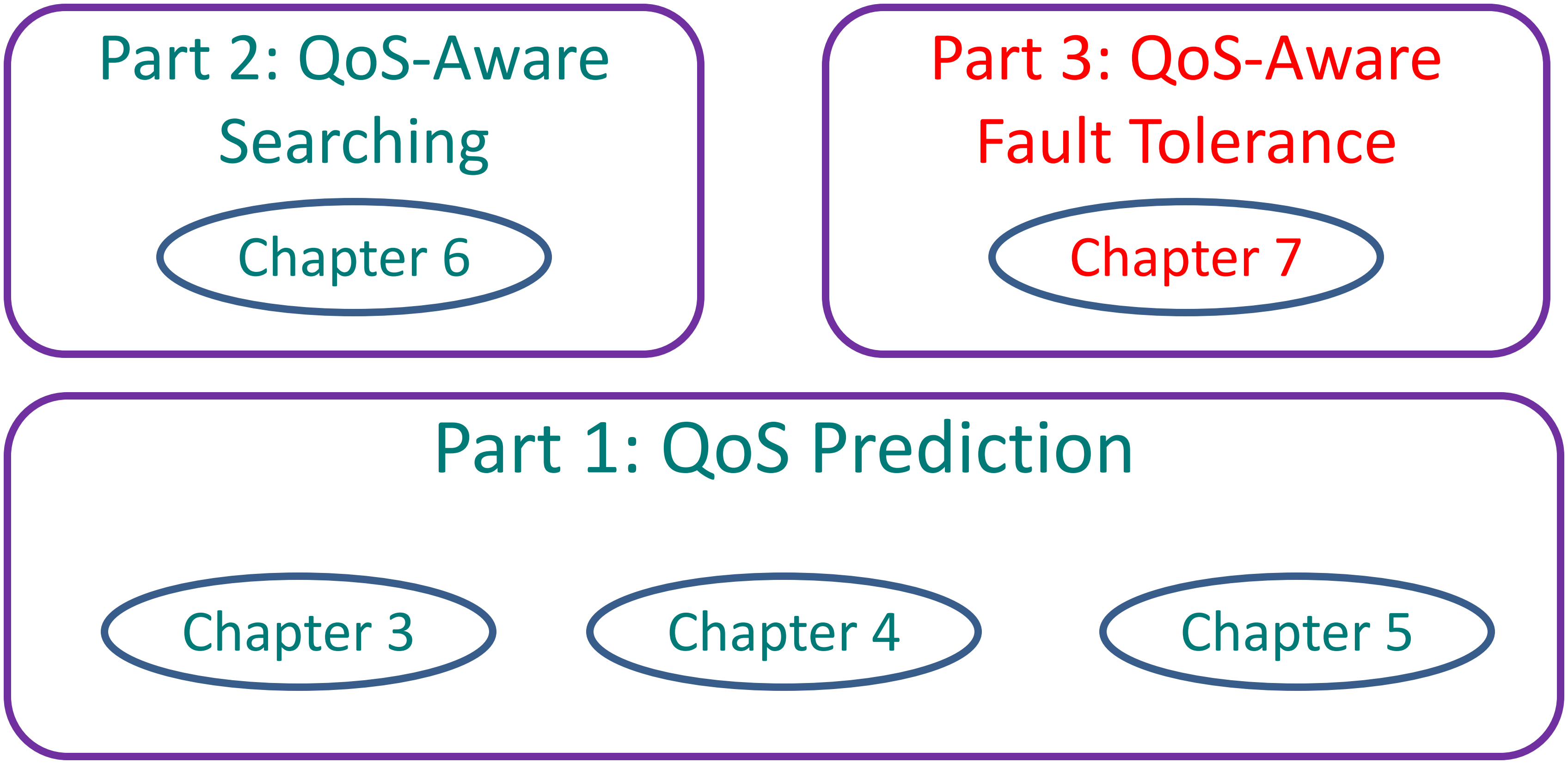
Performance Comparisons

Domain	Query ID	Top5		Top10		Top20		Top40	
		URBE	WSExpress	URBE	WSExpress	URBE	WSExpress	URBE	WSExpress
Business	1	0.437	0.661	0.444	0.599	0.439	0.633	0.527	0.659
	2	0.653	0.653	0.668	0.721	0.657	0.666	0.634	0.645
	3	0.402	0.502	0.456	0.512	0.502	0.544	0.574	0.603
	4	0.200	0.767	0.303	0.697	0.399	0.667	0.496	0.699
Education	5	0.603	0.742	0.604	0.753	0.598	0.664	0.631	0.717
	6	0.621	0.732	0.571	0.715	0.574	0.675	0.598	0.696
	7	0.645	0.688	0.579	0.671	0.560	0.643	0.632	0.662
	8	0.509	0.642	0.562	0.642	0.575	0.633	0.600	0.672
Science	9	0.423	0.538	0.478	0.549	0.495	0.572	0.502	0.578
	10	0.573	0.731	0.525	0.717	0.546	0.693	0.602	0.702
	11	0.632	0.819	0.613	0.823	0.583	0.757	0.628	0.774
	12	0.622	0.754	0.593	0.728	0.582	0.681	0.597	0.734
Weather	13	0.214	0.574	0.245	0.551	0.243	0.559	0.259	0.581
	14	0.713	0.825	0.701	0.814	0.687	0.802	0.725	0.824
	15	0.431	0.581	0.346	0.566	0.465	0.566	0.530	0.606
	16	0.475	0.611	0.485	0.519	0.501	0.529	0.525	0.543
Media	17	0.409	0.516	0.419	0.485	0.403	0.496	0.589	0.530
	18	0.393	0.519	0.373	0.488	0.450	0.527	0.532	0.567
	19	0.544	0.740	0.554	0.683	0.512	0.642	0.551	0.683
	20	0.504	0.678	0.473	0.613	0.451	0.559	0.497	0.602

A larger NDCG value means a better performance

Contributions

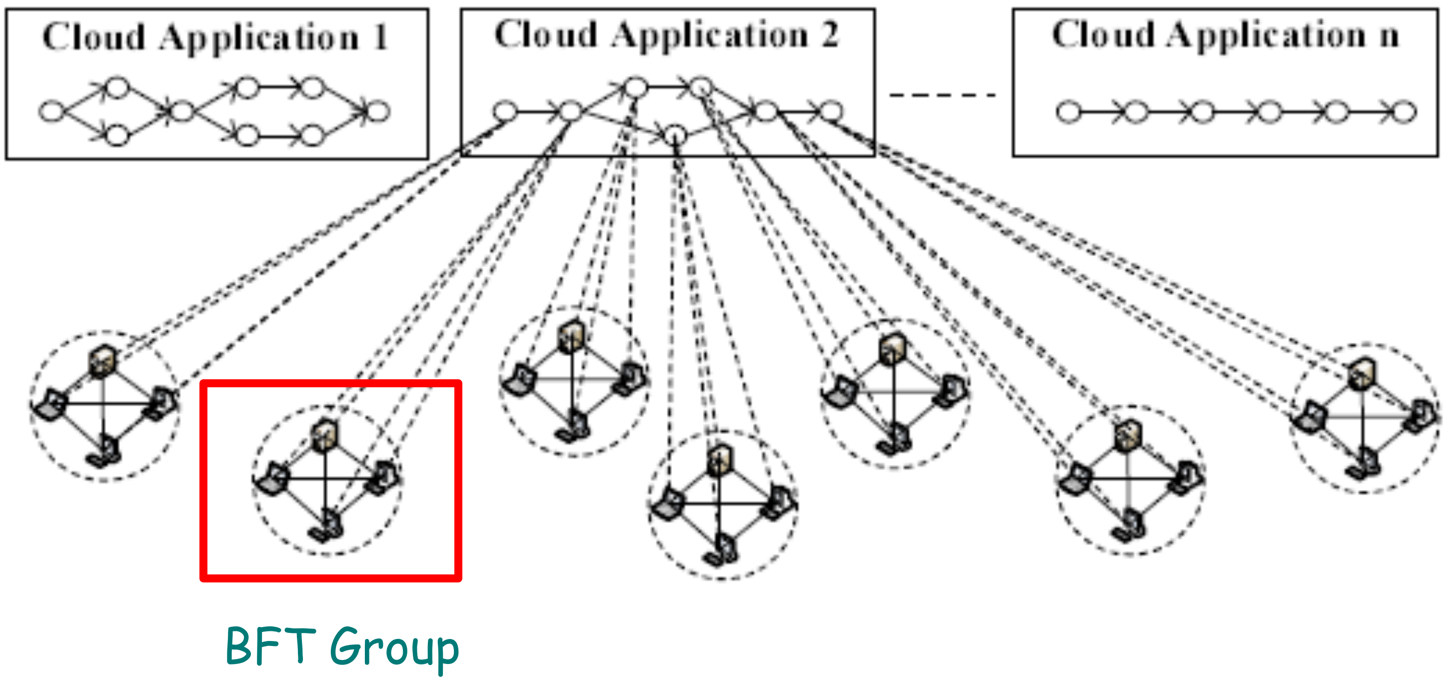
- Functionality and non-functionality
- A large-scale distributed experimental evaluation
  - 3738 Web services
  - 69 countries
- real-world WSDL dataset and QoS dataset
  - 30+ institutes


Part 2: QoS-Aware Searching
Chapter 6
Part 3: QoS-Aware Fault Tolerance
Chapter 7
Part 1: QoS Prediction
Chapter 3
Chapter 4
Chapter 5

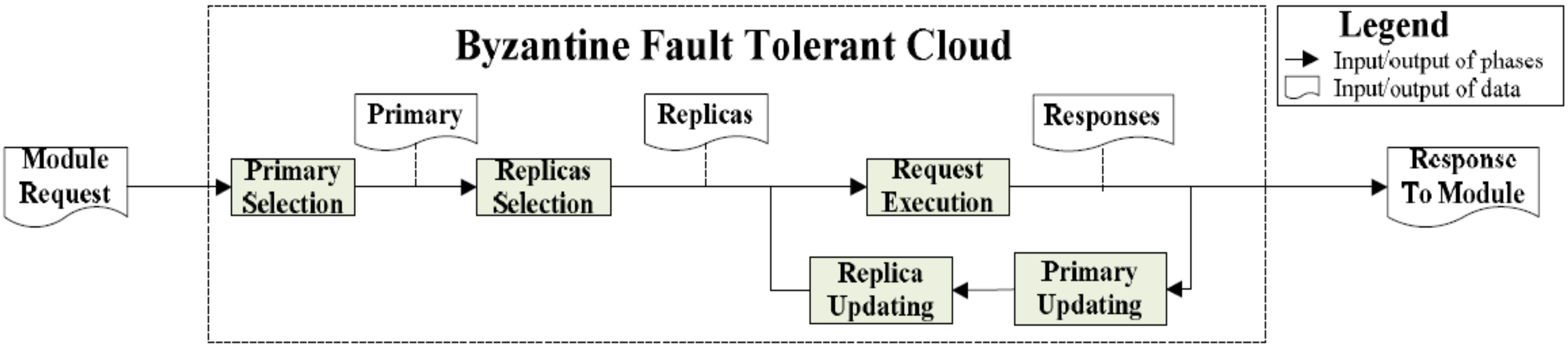
How to Build Reliable Service-Oriented Systems

- Problems
  - Services may contain various faults
  - QoS of remote services may not be stable, e.g., unavailability problem
- Solution: QoS-aware fault tolerance

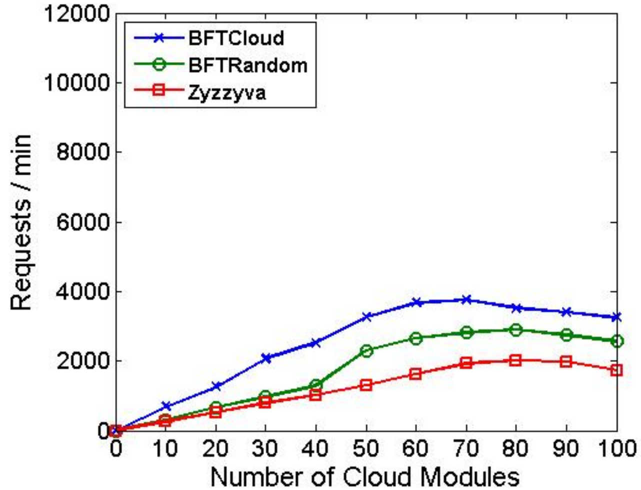
System Architecture



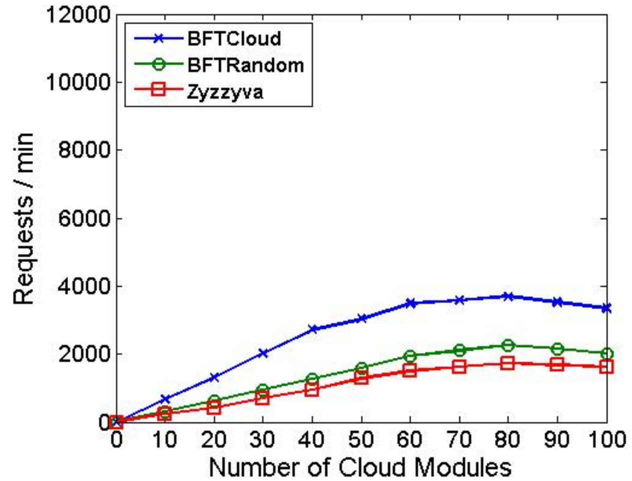
Work Procedures of BFTCloud



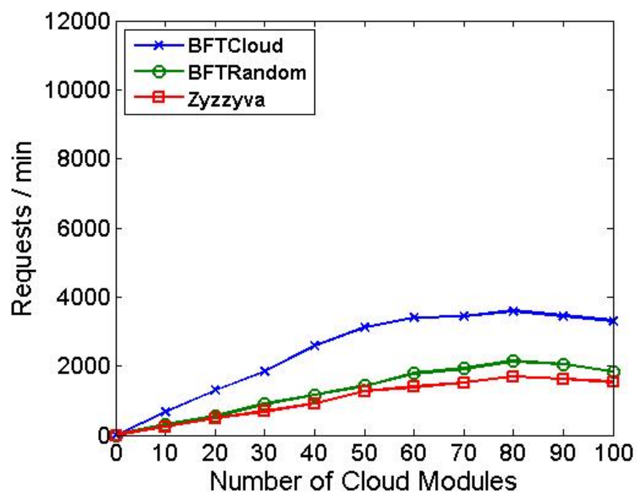
Experimental Results



Request/Response Size: 0/0 KB



Request/Response Size: 4/0 KB



Request/Response Size: 0/4 KB

Contributions

- A Byzantine fault tolerance framework
  - QoS-aware
- A prototype system
- Large-scale real-world experiments

Thesis Structure

Part 2: QoS-Aware Searching

Chapter 6

Part 3: QoS-Aware Fault Tolerance

Chapter 7

Part 1: QoS Prediction

Chapter 3

Chapter 4

Chapter 5

Conclusion

- Part 1: QoS prediction
  - Three prediction approaches
  - Several following up works on this topic using the released datasets
- Prat 2: QoS-aware service searching
  - Searching qualities are significantly improved
- Part 3: QoS-aware fault tolerance
  - Byzantine fault tolerance
  - dynamic QoS information

Publications

Paper published

- **Yilei Zhang**, Zibin Zheng and Michael R. Lyu. "Real-Time Performance Prediction for Cloud Components", in Proceedings of the 5th International Workshop on Real-Time Service-Oriented Architecture and Applications (RTSOAA 2012), Shenzhen, China, Apr. 11-Apr. 13, 2012, pp.106-111.
- **Yilei Zhang**, Zibin Zheng and Michael R. Lyu. "WSPred: A Time-Aware Personalized QoS Prediction Framework for Web Services", in Proceedings of the 22th IEEE Symposium on Software Reliability Engineering (ISSRE 2011), Hiroshima, Japan, Nov. 29-Dec. 2, 2011, pp.210-219.
- **Yilei Zhang**, Zibin Zheng and Michael R. Lyu. "Exploring Latent Features for Memory-Based QoS Prediction in Cloud Computing", in Proceedings of the 30th IEEE Symposium on Reliable Distributed Systems (SRDS 2011), Madrid, Spain, Oct. 4-7, 2011, pp.1-10.
- **Yilei Zhang**, Zibin Zheng and Michael R. Lyu. "BFTCloud: A Byzantine Fault Tolerance Frame-work for Voluntary-Resource Cloud Computing", in Proceedings of the 4th IEEE International Conference on Cloud Computing (CLOUD 2011), Washington DC, USA, July 4-9, 2011, pp.444-451.
- Zibin Zheng, **Yilei Zhang**, and Michael R. Lyu, "Investigating QoS of Real-World Web Services", IEEE Transactions on Service Computing.

Publications

- **Yilei Zhang**, Zibin Zheng and Michael R. Lyu. "WSExpress: A QoS-Aware Search Engine for Web Services", in Proceedings of the 8th IEEE International Conference on Web Services (ICWS 2010), Miami, Florida, USA, July 5-10, 2010, pp. 91-98.
- Zibin Zheng, Xinmiao Wu, **Yilei Zhang**, Michael R. Lyu and Jianmin Wang. "QoS Ranking Prediction for Cloud Services", IEEE Transactions on Parallel and Distributed Systems.
- Zibin Zheng, **Yilei Zhang** and Michael R. Lyu. "Distributed QoS Evaluation for Real-World Web Services", in Proceedings of the 8th IEEE International Conference on Web Services (ICWS 2010), Miami, Florida, USA, July 5-10, 2010, pp. 83-90.
- Zibin Zheng, **Yilei Zhang** and Michael R. Lyu. "CloudRank: A QoS-Driven Component Ranking Framework for Cloud Computing", in proceedings of the 28th IEEE International Symposium on Reliable Distributed Systems (SRDS 2010), New Delhi, India, Oct.31-Nov.3, 2010.

Paper under review/preparation

- **Yilei Zhang**, Zibin Zheng and Michael R. Lyu. "An Online Performance Prediction Framework for Service-Oriented Systems", submitted to the IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans.
- **Yilei Zhang**, Zibin Zheng and Michael R. Lyu. "QoS-Aware Web Service Searching ", prepared to submit to IEEE Transactions on Service Computing.
- **Yilei Zhang**, Zibin Zheng and Michael R. Lyu. “QoS Prediction via Latent Feature Learning in Cloud Computing", prepared to submit to IEEE Transactions on Cloud Computing.

- Thank you!
- Q & A
- Email: ylzhang@cse.cuhk.edu.hk