2010 Fifth IEEE International Symposium on Service Oriented System Engineering

GOS: A Global Optimal Selection Approach for QoS-Aware Web Services Composition

Mu Li, Ting Deng, Hailong Sun, Huipeng Guo, Xudong Liu
School of Computer Science and Engineering
Beihang University
Beijing, China
{ limu, dengting, sunhl, guohp, Liuxd }@act.buaa.edu.cn

***Abstract*—Services composition technology provides a promising way to create a new service in services-oriented architecture (SOA). However, some challenges are hindering the application of services composition. One of the greatest challenges for composite service developer is how to select a set of services to instantiate composite service with quality of service (QoS) assurance across different autonomous region (e.g. organization or business). To solve QoS-aware Web service composition problem, this paper proposes a global optimization selection (GOS) based on prediction mechanism for QoS values of local services. The GOS includes two parts. Fisrt, local service selection algorithm can be used to predict the change of service quality information. Second, GOS aims at enhancing the run-time performance of global selection by reducing to aggregation operation of QoS. The simulation results show that the GOS has lower execution cost than existing approaches.**

I. INTRODUCTION

How to integrate data, applications and systems under distributed, dynamic and heterogeneous environment is an important issue in distributed computing technology. Web service technology, as a new scheme for distributed software development, deployment and integration, is a possible way to solve this problem. Web services based on SOAP、WSDL、UDDI [1]and BPEL [2] have advantages in building business process with standard, loosely-coupled, reusable, flexible and effective nature. Composition approaches have been a hot spot of researching focus in the field of Web services. There are two major Web services composition techniques: semi-automatic composition approaches [3, 4], including workflow scheme based on orchestration [5] and interaction protocol/session scheme based on choreography [6], and automatic composition approaches that are based on semantic [7-11]. Generally, the composition process is theoretically composed of following two steps: tasks plan and QoS-aware services selection [17] which are responsible for finding service instances that match the functionality for each abstract service in composite services by using services composition approaches and selecting an optimal solution and execute it under the constraints of users' QoS by synthetically evaluating the candidate services, respectively. Our paper mainly focuses on the last step on the basis of workflow composition technology. Current related works include QoS aggregation [12, 13] (compute the overall QoS of execution plans based on QoS of candidate services), semantic QoS matchmaking [14-16] (find plan model that satisfies the overall users' non-functional requirements) and global optimal decision making [17-22] (find the optimal solution).

However, existing QoS aggregation algorithms and decision making methods have a common assumption that, QoS values of local services remain changeless during the process of QoS aggregation and globally optimal decision making. This assumption can no longer satisfy the business application requirement for timeliness. More and more researches have shifted their focus to the fields of QoS values prediction [23-25]. QoS is composed of multi-dimensional quality attributes and user concerns. User concerns are users' definition of interests in various QoS quality attributes. The nature of autonomy and distribution over Internet determines that the QoS attributes of Web services (such as response time and availability) change dynamically by the network environments. From the view of fairness the evaluation of QoS (like reputation) should be dynamically updated after every services execution. Moreover weight of QoS attributes could be adjusted by different user, so current services selection methods can not effectively support this dynamic change of QoS. Thus this paper introduces a prediction-based composition algorithm, called GOS. It has two sub-processes: local service selection and global selection. Local service selection is an evaluating algorithm for QoS, consisting of three major tasks: non-dimensional-normalization, prediction and evaluation of candidate solutions. Global selection algorithm is based on structure optimization, aiming at looking for solution satisfied user QoS's constraints (such as ''the total price of the composite service execution should be at most $500'').

The paper contributions are: 1) local service selection approach to solve the decision-making of candidate solutions based on dynamic prediction; 2) global selection to solve the solution of compositional services by analyzing structure of workflow; 3) Simulation results show that GOS has advantage in efficiency to the optimal solution over other composition methods.

The remainder of this paper is organized as follows: Section 2 introduces QoS model of composition service. Section 3 gives an overview of the GOS architecture. Section 4 discusses the details of predicted mechanism of local service selection. The implementation strategy of GOS is presented in section 5. Section 6 presents the simulation

978-0-7695-4081-8/10 $26.00 © 2010 IEEE
DOI 10.1109/SOSE.2010.47

IEEE computer society

results and analysis. Section 7 summarizes the related work in this field. Finally, we conclude the paper and outlines our future works in Section 8.

II. QoS MODEL OF COMPOSITION SERVICE

In this section, we give a QoS model of composite service based on the workflow model. Mainstream description languages of service composition, like BPEL, are developed from traditional workflow researches. The basic structures used in BPEL to create compositions include [26]: Sequence (S), Loop (L), Exclusive Choice (C) and Parallel (P), as shown in Fig- 1, which are the basis of our discussion in this paper.

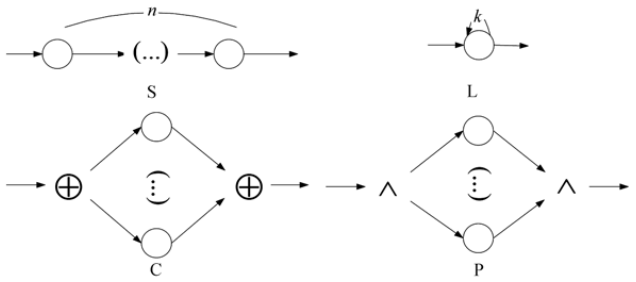


Fig- 1 Basic structure of workflow

Task is an atomic element of composite service. Sequence contains n tasks that are performed sequentially. Loop provides repeated execution of a given task. Exclusive Choice consists of several conditional branches, but only one task will be executed when the corresponding condition is true. All the branches in a parallel structure must be executed at the same time. These four workflow structures are defined as a blocks {Type: t_1, t_2, …, t_n}, Type∈{S, L, C, P}. For example, b_1={S: t_1, t_2} denotes a composite service composed of two task t_1 and t_2 executed sequentially. All the tasks are connected by defined structures. These atomic structures can be nested and combined in arbitrary ways to form complex composition of services. Thus we can define composite services as blocks in a recursive structure. For example, b_2={C: t_3, b_1} denotes exclusive choice between task t_3 and block b_1.

In Table 1, k denotes the number of times of task in loop, $|P|$ denotes the number of nodes in sequence structure, and c_i denotes the possibility of executing the ith branch in exclusive choice structure. Fig- 2 is an example of travel agent from [12], in which $t_2$and t_3 are parallel and can execute in arbitrary order. Using the definition we give in the above paragraph, this composite service can be described as {S, t_1, {P, t_2, t_3}, t_4, t_5}. We can use the aggregation strategy in Table 1 to first aggregate t_2 and t_3 to obtain the QoS of this parallel structure and then aggregate the t_1、t_4 and t_5 sequentially to get the overall QoS of this execution plan.

TABLE 1 QoS AGGREGATION STRATEGIES

	S	L	C	P
Price	$\sum p_i$	$k \times p_i$	$\sum c_i \times p_i$, $c_i \in (0,1], \sum c_i = 1$	$\sum p_i$
Available	$\prod a_i$	a_i^k	$\sum c_i \times a_i$, $c_i \in (0,1], \sum c_i = 1$	$\prod a_i$
Response time	$\sum t_i$	$k \times t_i$	$\sum c_i \times t_i$, $c_i \in (0,1], \sum c_i = 1$	$\max(t_i)$
Reputation	$\frac{1}{\lvert P \rvert}\sum r_i$	r_i	$\sum c_i \times r_i$, $c_i \in (0,1], \sum c_i = 1$	$\min(r_i)$

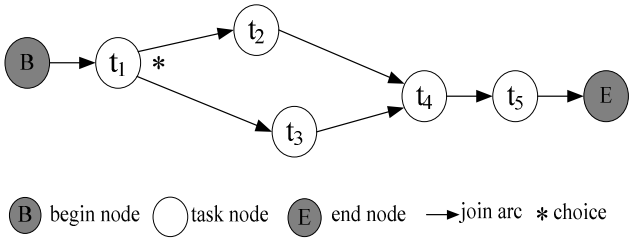


Fig- 2 Travel agent

A composite service *wsc* has a set of QoS parameters, $Q(wsc)=\{q_1,q_2,\ldots,q_z\}$. $Q(wsc)$ is the aggregated result of all the single attribute with Table 1 in a plan. We weighted average to rank final plans, as illustrated in eq.(1), where w_i is the user assigned weight of the ith QoS parameter.

$$QoS(wsc) = \sum_{1 \le i \le n} w_i q_i \ , \ (1 \le w_i \le 1, \ \sum_{1 \le i \le n} w_i = 1) \qquad (1)$$

III. GOS ARCHITECTURE

In this section we first introduce the system architecture of GOS. It continues to use definition of QoS-aware Web services composition in [17]. As shown in [17], QoS-aware Web services composition has two situations including local optimization and global selection. In this paper, we combine the local optimization into global selection as shown in Fig 3. For global optimal selection, firstly, the workflow needs to be transformed into a block. Next, we select best service for each task in the block, in which use recommend service by local prediction approach. Last, GOS aggregate QoS of the block and verify whether selection result satisfies user QoS's constraint. Local selection approaches recommend service in runtime by prediction in two steps: the first step is

normalization of QoS values and the second one is prediction of QoS value in runtime.

As we know from above, less QoS aggregation times and maximum usage of local task plan strategy results lead to more efficient selection of optimal global selection. So we describe GOS from local predicting selection approach.

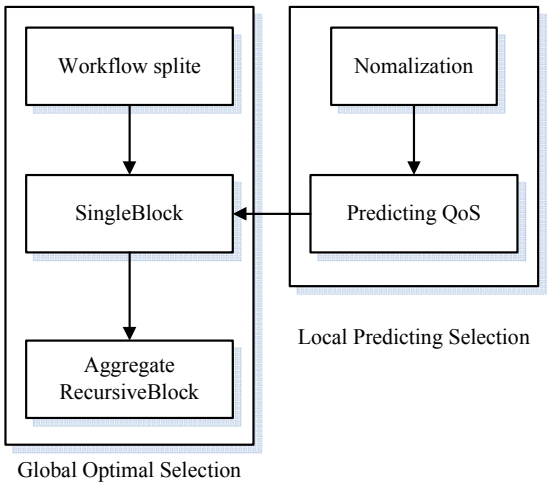


Fig- 3 Process of QoS-aware for Web services composition

IV. PREDICTION FOR QOS OF LOCAL SERVICES

A. Normalization of QoS values

As we've discussed, the turbulence of QoS makes it different at executing time from selecting time, thus results in non-optimal solution in solving local and global optimal solutions. To avoid this problem, we predict the QoS at executing time and use this predicted value to solve local service selection. Let q be the set of quality attributes, $M = (m_{ij})_{n\times m}$ is a data matrix, in which m_{ij} is the value of q_j (q_1= price, q_2 =availability, q_3= response time, q_{j4} =reputation,) at time point i. For convenience we assume that a service contains the four QoS attributes we've defined (*i.e.* price, availability, response time and reputation). Next we give the detailed decision steps of algorithm.

Different QoS attributes usually have different domains, e.g., price is given as dollars, but reputation is an integer number between 1 and 10. For assessing and comparing these data in a uniform criterion, the attribute values listed in $\tilde{M}$ need to be normalized. We extend the normalization method introduced in [17] with situation when the corresponding QoS value is missing. Formula (2) is used for computing benefit style data and (3) is used for cost style data, where $\tilde{Q}_{i,j}$ is set to -1 when $\tilde{M}_{i,j} = 0$, *i.e.*, the corresponding QoS is missing. Using (2) and (3) we get the normalized matrix $\tilde{Q}$.

$$\tilde{Q}_{i,j} = \begin{cases} -1 & \tilde{M}_{i,j} = 0 \\ \dfrac{\tilde{M}_j^{\max} - \tilde{M}_{i,j}}{\tilde{M}_j^{\max} - \tilde{M}_j^{\min}} & \tilde{M}_j^{\max} - \tilde{M}_j^{\min} \neq 0 \\ 1 & \tilde{M}_j^{\max} - \tilde{M}_j^{\min} = 0 \end{cases} \quad (2)$$

$$\tilde{Q}_{i,j} = \begin{cases} -1 & \tilde{M}_{i,j} = 0 \\ \dfrac{\tilde{M}_{i,j} - \tilde{M}_j^{\min}}{\tilde{M}_j^{\max} - \tilde{M}_j^{\min}} & \tilde{M}_j^{\max} - \tilde{M}_j^{\min} \neq 0 \\ 1 & \tilde{M}_j^{\max} - \tilde{M}_j^{\min} = 0 \end{cases} \quad (3)$$

B. Services Selection by local prediction algorithm

We constructed a SEM model and quantitatively solved QoS value in future time [25]. Forecasting future QoS will help users choose the best service. By using SEMSS, we can get matrix Λ , which is eigenvalue matrix of M, and matrix G', which is orthogonal matrix of M. The λ_i is element of the matrix Λ . Eigenvalue λ_i^{T+l} at time $T+l$ can be forecasted if there is regression equation for aigenvalues $\lambda_i^t = f_i(t)$, i=1,2, …, p. Orthogonal transformation matrix can be seen as the rotation in multi-dimensional space, so each element G_{ij}^t of orthogonal matrix G^t denotes an angle, $-\frac{\pi}{2} \le \theta_{ij}^t \le \frac{\pi}{2}$. After making a regression equation $\theta_{ij}^t = f_{ij}(t) + \varepsilon_{ij}^t$, for each time series θ_{ij}^t , $1 \le i < j \le p$, $t = 1,2,\cdots,T$, we can forecast at time $T+l$.

The predict select algorithm step is :

(1) Predict covariance matrix M^{t+l} at time point $T+l$.

(2) Calculate the goodness-of-fit index and adjusted goodness-of-fit index (GFI) of M^{t+l} and sort them in ascend order by goodness-of-fit index.

(3) Calculate the GFI of each service by the sorting result.

(4) Choose the optimal service, i.e., with largest GFI, at the time point $T+l$.

Fitting goodness index GFI is the metric to examine the match degree of sample covariance matrix of M and the covariance matrix gotten by structural equation model. If sample covariance matrix of M equals to predicted covariance matrix then GFI is set to 1, showing complete fitness between predicted results and actual changes.

V. GLOBAL OPTIMAL SELECTION FOR WEB SERVICES COMPOSITION

A. Reduction tree splitting strategy

This section, we construct a reduction tree (*RT*) [26] from a block (help us analyzing nature of workflow) and provide a splitting strategy that can improve the efficiency of global selection, in which reduce the aggregation operation, by analyzing to structure of reduction tree.

Conceptually, less QoS aggregation times and maximum usage of results of local task plan strategy lead to more efficient selection of optimal global selection. So, we will partition nodes of *RT* into two categories that are Traversal Structure (*TS*) and Aggregate structure (*AS*) for distinguish aggregation. By prove theorem 1, we explain that we can reduce aggregate operation and maximum usage of local task plan in GOS.

A reduction tree (R-Tree) of a composite service *CS* is a binary tree with degree for each node, and can be expressed by a triple RT::=<N, *type*, η>, where $N=LN \cup NLN$. Here each node of *LN* is a leaf node representing a task and each node of NLV is a non leaf node representing structure; $type \subseteq \{S,C,L,P\}$ is set of structure types in *CS*.

Using this definition the compositional service model in Fig- 2 can be transformed to a tree structure as shown in Fig- 4. Follows we examine the workflow nature of a reduction tree. We divide the structure nodes in Fig- 4 into traversal type and aggregate type.

Traversal Structure (*TS*). If n leaf nodes (task) on a reduction tree are executed sequentially, the structure nodes of these n leaf nodes are traversal type (e.g., S and L).

Structure C with *m* branches equals to a *TS* with *m*-1 structure nodes S connected in right order, where each structure node s denotes a branch. The weight of structure node C is transferred to the corresponding structure node S.It is easy to understand that S and L is *TS*, so the proving is ignored. Now, we give a brief proof that C is *TS*.

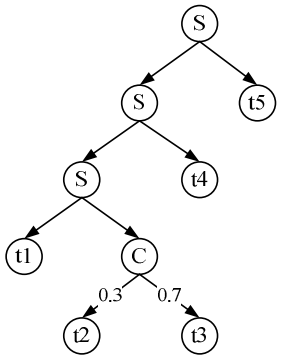


Fig- 4 Reduction Tree of travel agent

Assume that C is not *TS*, then at least two branches can be executed in parallel. Because a reduction tree is executed in a middle order, for a non-null reduction tree, the left sub-tree is executed in middle order first. The structure node is accessed and the node attribute is read. If the node is sequential then the right sub-tree is executed in middle order sequentially otherwise the right sub-tree is executed in middle order in parallel. This conflicts the definition of exclusive choice that one and only one branch is executed at one time. So the assumption is incorrect.

Aggregate structure (*AS*). Structure node P is a AS.

In *RT*, functions of a complex structure node include add, multiply, ave (average) function, max function and min function.

There are axiom 1 and lemma 1 in *RT*.

Axiom 1. Local predicting selection approach does not change the better/worse relations of any QoS attribute of any two candidate plans (as the normalization and prediction strategy does not change the better/worse relations of any QoS attribute).

Lemma 1. The linear operations (+−*/constants) on the corresponding QoS attributes does not change the better/worse relations of candidate Web services (because the linear operations do not change the better/worse relations of QoS attributes).

Now, we define theorem 1 and prove it.

Theorem 1. Assuming that a compositional logic only consists of traversal type structures, then its global selection solution is the combination of local optimal solution of each task.

Assuming that Theorem 1 is not sound, there is at least one service in the global selection solution, which is not the local optimal solution of corresponding task *T*. We assume the global selection solution is (ws^1,ws^2,…,ws^n), where ws^j denotes one candidate service of task T_j. Further we assume ws_i^m is the local optimal solution of T^m. $ws_i^m \neq ws_j^m, 1 \leqslant m \leqslant n$. According to the conditions of Theorem 1, the QoS aggregation rules are Σ, Π and Ava. In addition, we assume the workflow consists of n tasks and each candidate service has p QoS attributes. Then we only need to prove that (ws^1, ws^2,…, ws_i^m,…, ws^n) (*i* solution) is worse than (ws^1, ws^2,…, ws_j^m,…, ws^n) (*j* solution). Based on the above assumptions, the overall QoS of *j* solution $Q = \sum_{k=1}^{p} a_k q_k$, where $q_k = q^k{}_j + \sum_{l=1 \wedge l \neq j}^{n} q_l^k$ or $q_k = q^k{}_j \times \prod_{l=1 \wedge l \neq j}^{n} q_l^k$. Also the overall QoS of *i* solution $Q' = \sum_{k=1}^{p} a_k (q_k)'$, where $q_k = q^k{}_i + \sum_{l=1 \wedge l \neq i}^{n} q_l^k$ or $q_k = q^k{}_i \times \prod_{l=1 \wedge l \neq i}^{n} q_l^k$.

Further by Axiom 1, we know a_k, $\sum_{l=1 \wedge l \neq m}^{n} q_l^k$ and $\prod_{l=1 \wedge l \neq m}^{n} q_l^k$ can all be seen as constants. Based on the assumption that ws_i^m is worse than ws_j^m, i.e. *i*th solution is worse than *j*th solution, so the assumption that *i*th solution is the global selection solution is incorrect. So the assumption that there is at least one service in the global selection solution such that it is not the local optimal solution of corresponding task *T* is incorrect. □

Theorem 1 means that there is no need to aggregate the QoS for traversal type when finding the global selection solution for the compositional service. For aggregate type structure, aggregation is needed because the response time and max operator do not satisfy the conditions of Lemma 1.

B. Services Selection by global optimization

In this section, we describe a global optimization selection (GOS) based on the result of Theorem 1.GOS is used to find selection solution for compositional service plan, *i.e.*, bringing candidate services into the leaf nodes of a reduction tree generates task plans of compositional services. To satisfy the user-defined non-functional requirements, GOF need local predicted optimal selection.

After composite service engine generating the composite service *RT*, service adaptor can find a set of candidate services matching the functions and starts services selection. Each service adaptor owns a virtual topology same as the one of the compositional service reduction tree, and the corresponding task is replaced with the QoS information set of the candidate services. So though lack of the entire workflow of the compositional service, a node can still knows the service topology of his own part. Using the example in Fig- 2, the candidate services of the task may e located in three nodes as shown in Fig-6.

GOS traversals the workflow, invokes different blocks according to different structure node types, and return the planning solution after the iterative computation. The input parameters of algorithm include reduction tree structure *RT* and user constraints *C*. TaskPlan(*t*) is part of *RT* path which end with *t*. Plan(*t*) is the traversal nodes set from task starts to *t*. *T* contains the structure information of structure nodes. In line 15 we use theorem 1 as a judge to improve plan efficiency. Assume that RT has *m* task nodes, in the worst case the complexity of GOS is $O(Kn^2)+ O(m)= O(Kn^2)$. In the best case the time cost is only the plan time from the start node to the ending node.

Global Optimization Selection

```
1.  Algorithm: GOS
2.  initialize split workflow to RT
3.  predict best service of each t RT and selected;
4.
5.  Input: RT, C
6.  Output: P';
7.  Begin
8.    If the segmentResult is NOT empty then
9.      P'=TaskPlan(RT);
10.   Else
11.     p'=Plan(RT);
12.
13.   If RT is AS node then  // AS is defined in 4.1
14.     GOS (RT, C);
15.     aggregation(result.Plan);
16.     Pop resultPlan to P';
17.   Else if t is TS node then
18.     GOS (RT, C);;
19.     Pop resultPlan to P';
20.   End
```

VI. EXPERIMENTS

In this section, we perform initial simulation-based experiments to evaluate prediction accuracy, approximation ratio and success ratio of GOS algorithm and compare it with existing algorithm (Integer Programming (IP) [17], Backtracking (BT) [21] and Heuristics in Multi-choice, Multi-dimension 0-1 knapsack (H-MMKP)) [22]. For test, the Services Composition Engine of QoSOArWare deployed on Xeon(TM) 4.30 GHz CPU and 2 G RAM, and Services Adaptor of QoSOArWare deployed on 2 Intel(R) Xeon(TM) 4.30 GHz CPU and 1 G RAM. They interact through the Services Bus of QoSOArWare that is our key project.

Next, we test the run-time performance of GOS and compare it with exist algorithm under restrictive conditions in the number of tasks, per-task candidates and QoS constraints. They are the influence factor of the performance of QoS-aware web service composition algorithms.

A. Evaluation methodology

Proposing, the difference between the QoS prediction mechanism and the actual execution results of the web services, under the assumption that the weights of QoS attributes are fixed or updated with a run time of 100 days and four attributes, namely the price, credit, availability and ratio of successful execution. We assume that the weights of QoS attributes change in the different time period. For example in the initial phase users may prefer services lower price, but as the use frequency and service quality increase, the weight of price decrease and the weight of execution success increase.

In the static environment, the QoS attributes and user's concerns do not change. While in the dynamic environment. QoS values of candidate services may fluctuate dramatically or services even get off-line and on-line at any time. Before every selection for a task, 30% QoS parameters of all services would be changed to a new random value, and user concerns would random change. In this scenario, we select and plan composite service at run-time.

. All the composite service topology is the same as that shown in Figure 3, the structure between t1 and t5 is a switch structure that has equal possibility to both branches. We apply GOS and set *K*=1 in this scenario. Because the best QoS result is hard to define in this scenario, only success ratio presented.

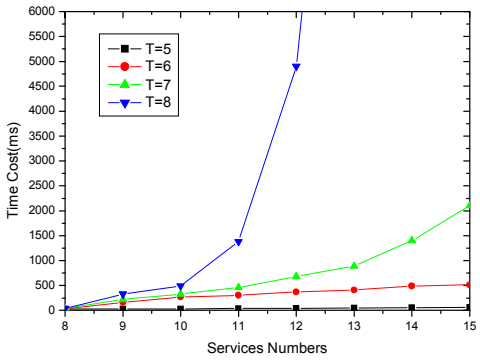
a) Last, we compare GOS with existing algorithm, and use composite service topology in Fig 2. For $<t_1,\ldots, t_5>$*, each task has 20 candidate services and K set from 1 to 5.*

B. results and analysis

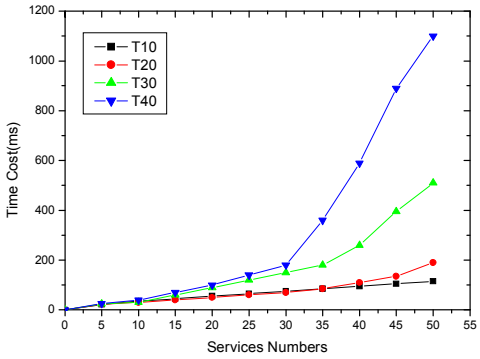
Fig- 5 shows performance comparison of basic Non-GOS and GOS, where confidence intervals with 0.95 confidence level are illustrated in figures. In Fig- 5 (a), with the growing number of candidate services and task number, the execution time of basic Non-GOS rapidly increases. The algorithm performance is unbearable when total candidate

service number reaches 150. Moreover in Fig- 5 (b), the GOS algorithm is far scalable than the basic one. For the same task number and service number in Fig- 5 (a), the GOS can finish computing in several milliseconds. When the total number of candidate service is approaching 2000, the execution time is still under 1 s and grows gently. The algorithm performance is comparable to previous work by Zeng et al [18]. We believe that this scalability is enough for most of the practical scenarios.

The second experiment is to analyze computation cost between prediction selection algorithm and IP (Integer Programming). The results in Fig- 6 show that GOS and existing algorithm have insignificant difference in computation cost. In third experiment, we increase number of task from 10 to 70, and record of computation cost in difference web services selection algorithm (IP, H-MMKP and BT) in dynamic environment.



(a) Time cost of Non-GOS



(b) Time cost of GOS

Fig- 5 Compare time cost with GOS and Non-GOS

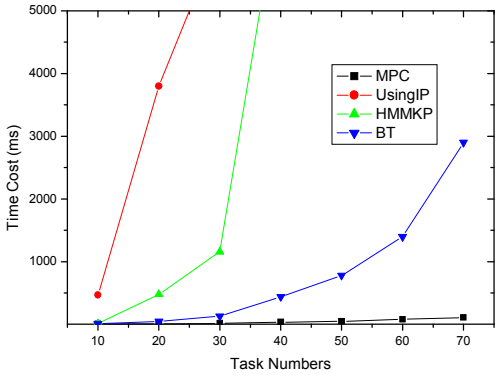


Fig- 6 Experimental results (computation cost) in a dynamic environment,varying the number of services per task

VII. RELATED WORK

To achieve selection result (satisfied user constraints), many efforts have been made. Related work mainly involves service composition, dynamic selection, dynamic predicting and adaptation technology, monitoring and recovery technology, evaluation, replication technology and so forth. Dynamic service composition method is proposed by and L. Zeng [17, 18]. In the definition period only required func tions are defined. Component services are bound and instantiated at runtime. Composite service communicates with service registry dynamically according to the pre-established strategies. These strategies include choice based on QoS [19], or based on semantic [13] and so on. These methods improve the flexibility and adaptability in selection. But they fail to solve dynamically searching. The multiple remote interactions needed and the efficiency will be low. Because the existing service registries cannot guarantee the authenticity of data and the state of registered services, the quality of services cannot be guaranteed.

H.P. Guo [15] proposed a method for adaptive maintenance of composite services in dynamic environment. The method use modeling the control process as a Markov decision process (MDP), and then predict service state with algorithm based on Kalman-Filter. L.S. Shao [23] proposed predictive methods based on making similarity mining and prediction from consumers' experiences. Their approach can make significant improvement on the effectiveness of QoS prediction for web services. However, they fail to support the adaptive concerns of attributes in dynamically environment.

S. Guinea [19] presented a self-healing method through service state monitoring and recovery of fails. V. Issarny and F. Tartanoglu [20] proposed a method to achieve fault tolerance and dependable composite service for forward error recovery and based on the concept of cooperative atomic action and web service composition action. But they unconsidered monitoring and recovery efficiency, costs and

rewards, and also they do not assess the effect of the monitoring and recovery in quantitative forms. In addition, In ref.[21], the composition problem was modeled as a multi-dimension multi-choice 0-1 knapsack problem as well as a multi-constraint optimal path algorithm. Berbner et al.[22] introduced a set of QoS aggregation algorithms and a backtracking based plan selection algorithm. For all, heuristics were presented. Their experiment results revealed that heuristic algorithms show a better performance and scalability than linear integer programming with increasing numbers of candidate web services and tasks. In section 5, we performed initial simulation-based experiments to compare GOS with the mathematical programming and heuristic approaches. The conclusion is GOS had the lowest computation time and highest approximation ratio and success ratio when the number of tasks, candidates and QoS constraints changed.

VIII. Conclusion and Future Work

Service selection is essential to user's feeling of the compositional service. Our paper summarizes related works and introduces a novel iterative algorithm to service selection. GOS has two sub-processes: local service selection and global selection. Local service selection is an evaluating algorithm for QoS, consisting of three major tasks: non-dimensional-normalization, prediction and evaluation of candidate solutions. Global optimal selection algorithm is based on structure optimization, aiming at looking for solution satisfied user QoS's constraints (such as "the total price of the composite service execution should be at most $500").This paper major contributions are: ① local service selection approach to solve the decision-making of candidate solutions based on dynamic prediction; ② global optimal selection to solve the optimal solution of compositional services by analyzing structure of workflow; Other contributions of our work are constructed a QoS model for multi-dimensional non-function attributes and simulation results show that GOS has advantage in efficiency to the optimal solution over other composition methods. Though GOS achieves better performance than other existing method, further improvement is needed. An efficient ontology management framework is under design to perform knowledge reasoning and ontology evolution to enhance the automatic degree of GOS. In addition we will design a dynamic scheduling algorithm which will allocate the task of computing local optimal solution to the compositional service engine based on the real-time resources of the service adaptor, to maximize the resource utility.

Acknowledgment

This research was supported in part by China 863 High-tech Program (SN: 2007AA010301 and SN: 2009AA01Z419) and the National Natural Science Foundation of China (SN: 2005CB321803).

References

[1] F. Curbera, et al, IEEE Internet Computing: Spotlight - Unraveling t he Web Services Web: An Introduction to SOAP, WSDL, and UDDI. IEEE Distributed Systems Online 3(4): (2002)

[2] T. Audrews, F. Curbera. Business Process Execution Language for Web Services 1.1. ftp://www6.software.ibm.com/software/developer /library/ws-bpel.pdf

[3] M. E. C.Gregorio Dyaz, J. Pardo Juan, Valentin Valero, F. Cuartero. Automatic generation of Correct Web Services Choreographies and Orchestrations with Model Checking Techniques [A]. In Proceeding s of Proceedings of the Advanced Int'l Conference on Telecommunic ations and Int'l Conference on Internet and Web Applications and Se rvices (AICT-ICIW)[C]. 2006.

[4] S.Battle, A.Bernstein, H.Boley, et al. Semantic Web Services Frame work (SWSF) Overview[EB/OL]. http://www.w3.org/Submission/S WSF/

[5] Business Process Execution Language for Web Services version1.1, May, 2005

[6] Web Services Choreography Description Language Version 1.0, Nov ember, 2005

[7] S. A.Mcllraith, T.Son, H.Zeng, Semantic Web Services[J]. IEEE Inte lligent Systems, 2001, 46-53.

[8] S. A.Mcllraith, T. Son. Adapting Golog for Programming the Seman tic Web[A]. In Proceedings of the Fifth Symposium on Logical Form alizations of Commonsense Reasoning[C]. 2001:195-202.

[9] T. Li, J.Z. Li, K.H.WANG , Heterogeneous Messages Match and Re use in Web Services [J]. CHINESE JOURNAL OF COMPUTERS , 2006, 29(7):1038-1047

[10] D. L. McGuinness, F. Harmelen. OWL web ontology language over view. 2004: http://www.w3.org/TR/2004/REC-owl-features-200402 10/.

[11] D. Martin, et al. The OWL Services Coalition. OWL-S 1.0 Release. 2003: http://www.daml.org/services/owl-s/1.0/.

[12] H.Guo, J. P.Huai, et al. ANGEL: Optimal Configuration for High Av ailable Service Composition. Proceedings of the IEEE International Conference on Web Services, Salt Lake City, Utah, USA, July, 2007.

[13] F.C.YANG, S. SU, Z. LI Hybrid QoS-aware semantic web service c omposition strategies. Science in China SeriresF. 2008 Vol. 51 (11): 1822-1840

[14] E M. Maximilien, M P.Singh, A framework and ontology for dynami c web services selection. IEEE Internet Comput, 2003, 8: 84—93

[15] C. Zhou, L. T. Chia, B S.Lee DAML-QoS ontology for web services. In: Zhamkng L J, ed. Proc ICWS'04. California: IEEE Computer So ciety, 2004. 472—479

[16] S. Ran. A Model for Web Services Discovery with Qos. ACM SIGec om Exchanges. V4, I 1, pp. 1-10. 2003

[17] L.Z. Zeng, et al, QoS-Aware Middleware for Web Services Composition. IEEE TRANSACTION ON SOFTWARE ENGINEERING, VOL 30, NO. 5. MAY 2004

[18] L.Z. Zeng, et al,. Monitoring the QoS for Web Services. ICSOC, 2007.

[19] S. Guinea. Self-healing web service compostions. proceedings of the ICSE., pp. 655-655, 2005

[20] V. Issarny, F. Tartanoglu, A. Romanovsky, and N. Levy. Coordinated forward error recovery for composite web services. Proceedings of the SRDS, pp. 167-176, 2003

[21] T. Yu, K. J. Lin. Service Selection algorithms for composing complex services with multiple QoS constraints. In: Benatallah B, Casati F, Traverso P, eds. Proc. of ICSOC'05. Heidelberg: Springer-Verlag, 2005. 130－143

[22] R. Berbner, M. Spahn, N. Repp, et al. Heuristics for QoS-aware web service composition. In: Leymann F, ed. Proc. of the ICWS'06. Chicago: IEEE Computer Society, 2006. 72－82

[23] L.S. Shao, et al. Personalized QoS Prediction forWeb Services via Collaborative Filtering. ICWS 2007

[24] J.X. Wu, F.C. Yang. QoS Prediction for Composite Web Services with Transactions.

[25] M. Li, J.P. Huai, H.P. Guo. An Adaptive Web Services Selection Method Based on the QoS Prediction Mechanism. IEEE/WIC/ACM International Conference on Web Intelligence 2009. (Accepted)

[26] N. Mulyar, W.M.P. van der Aalst, A.H.M. ter Hofstede and N. Russell. Towards a WPSL: A Critical Analysis of the 20 Classical Workflow Control-flow Patterns, BPM Center Report BPM-06-18, BPMcenter.org, 2006.

[27] S.Su, F. Li, F. C.Yang. Iterative selection algorithm for service composition in distributed environments. Science in China SeriresF. 2008 Vol. 51 (11): 1841-1856