Applied Mathematics & Information Sciences
An International Journal

Research on Transaction Web Service Selection Algorithm in WSC

Hu jingjing[1], *Zhao xing*[2], *and Cao yuanda*[1]

[1] School of Software, Beijing Institute of Technology, Beijing 100081 P. R. China
[2] School of Mathematics, Capital Normal University, Beijing 100037 P. R. China

Received: 20 Sep. 2012; Revised 28 Nov. 2012; Accepted 9 Dec. 2012
Published online: 1 Mar. 2013

Abstract: Web service composition (WSC) needs transactional support to guarantee its consistent and reliable execution. To improve the validity of service selection, a transaction service composition model (TSM) is proposed. The model associates the quality of service (QoS) with dependencies between transactional states and time. Based on the model, a parallel algorithmłłtransaction QoS selection algorithm (TQSA) is developed, which combines genetic algorithm (GA) and service compensation computing, and can be applied to dynamic service composition due to its mechanism of selection and evaluation performing at the same time. To improve the success of composition, an algorithm I-TQSA is further developed by optimizing evolution strategy. Experiment results show that TQSA resolves the QoS-aware selection NP-hard complete problem, and I-TQSA further improves the effectiveness of selection and reduces composition time.

Keywords: Service selection, transaction, QoS, GA.

1. Introduction

Web services provide the solution of the integration for the heterogeneous and autonomous components. However, single web service cannot satisfy the requirements for users. Therefore, how to effectively integrate several web services into a composite one has been a challenge and is attracting more attention [1]. Web services composition (WSC) is a complex process involving several steps. The significant step is the process of web services selection for each task. Some web services selection approaches such as QoS based selection algorithms [2–4] have been presented. Nevertheless, few of them consider the association of transaction in services selection, especially in dynamic WSC environment where it possibly leads to losses of cost and even failures of WSC. Advanced transactional support is required to ensure the overall consistency, integrity and reliability, and selection algorithms supporting transaction processing for web service composition have been presented [5].

This paper proposes a transaction service composition model (**TSM**) to improve the validity of service selection. Based on the model, a transaction QoS selection algorithm (**TQSA**) is developed to fit dynamic service composition and provide compensation strategy for long-running service. To improve the success of composition, an algorithm **I-TQSA** is further developed by optimizing evolution strategy. The paper is organized as follows: The next section is a brief introduction to relevant work. The third section puts forward to the problems and solutions. The fourth section describes the transaction service composition model. The fifth section gives the selection algorithm for dynamic composition. The sixth section explains the performance evaluation, and the last section presents our conclusions.

2. Related works

As for all collaborative systems, composite services need transactional support to guarantee their consistent and reliable execution. The traditional notion of transactions with ACID properties is too restrictive for the types of complex transactional activities that occur in distributed applications, primarily because locking resources during the entire execution period is not applicable for long running transactions (LRTs) that require relaxed atomicity. Advanced

* Corresponding author: e-mail: hujingjing@bit.edu.cn

transaction models (ATMs), have been defined to better support LRTs in a distributed environment. It relaxes the requirement of the entire transaction as an atomic action and creates a compensator to undo the results of the task logically. So, several Web services protocols such as WS-Business Activity [6] adopt the compensation mechanism whose basic idea is that the effect of a completed service can be semantically undone by a so-called compensation operation of the service. These protocols provide transactional support for distributed processes in service-oriented environments. But it is only unilateral channel in service selection because it focuses on the business logic of exception handling or fault prevention mechanism, and ignores the profound impact of exception or compensation for service composition, which would lead to some problems: In such cases, first, even if with higher priority, can those web service candidate providers often necessary to compensate in composition be selected in the subsequent course of business? Second, does the transaction failure in service execution have a measurable association with quality of service? Third, can the success or failure of transaction establish a uniform metric to guide service selection strategy?

WSC should ensure not only correct and reliable execution but also optimal quality of service (QoS) [7]. Indeed on one hand, WSC based on transactional properties ensures a reliable execution however an optimal QoS composite service is not guaranteed. Moreover on the other hand, composing optimal QoS services does not guarantee a reliable execution of the resulting composition. Thus, QoS-aware and transactional-aware should be integrated. However, the problem is generally addressed from the QoS side or from the transactional side separately. The conventional QoS-aware services selection algorithms do not consider the transactional constraints during the composition process [3,4], likewise transactional-aware ones do not consider QoS [5]. So, implementing a composite web service to integrate transaction and QoS becomes an important challenge. The research has been emerging in this regard [7], but can only be used in static service composition, because that algorithm is realized by capturing both aspects in order to evaluate the QoS with various transactional requirements in design-time. So it cannot support dynamic service selection. Other research proposes the selection algorithms by matching transactional properties with the users' QoS requirements [8]. It is also hard to implement service selection while service executing synchronously.

3. Key problems and design principle

With the increasing number of web services on the Internet, the complexity of service composition has increased [9].WSC should investigate in automatic selection where the user is relieved as much as possible from the composition and execution processes to ensure correct and reliable execution as well as optimal QoS. Therefore, in order to achieve it, the following key issues need to be addressed:

(i) QoS model supporting transaction service composition

In order to achieve global optimization, the selection algorithm should be constructed in the services composition framework. So the requirement for the formal model is put forward. It would facilitate the automatic selection of the dynamic composition. The model also needs to be able to evaluate the transactional characteristics and easy to integrate with QoS. The feasible solution is to establish automatic services composition model, which can be applied in real-time dynamic integration of tasks. The model would support compensation mechanism of LRTs. Web services' transactional attributes reflect the parts of QoS, and through establishment of the association between them the standards can be specified and unified automatically.

(ii) Service selection algorithm base on transaction QoS

In the algorithm, it needs a proper mechanism to aggregate the qualitative transaction criteria and existing service evaluation in dynamic environment. The effectiveness of selection in WSC also needs considering.

The feasible solution is to design parallel algorithms [10] for implementing dynamic service selection and take advantage of uncertain search algorithm for achieving composition optimization.

4. WSC choreography model

This section presents the formal description of WSC. In the model of WSC, services are defined to specify the web services implementing transactional interoperability between web services domains and provide a means to compose transactional qualities of service into web services applications. It represents a transaction service as a Finite State Machine (FSM), and regards service as an activity sequence with state. Modeling service as FSM is the most promising automatic composition approach for dynamic WSC [11,12]. Service composition model can be implemented through a combination of automata.

4.1. Transaction service model

In the transaction model, the notion of transaction can be reflected by the service 'state', and 'time' is an important metric in QoS [13]. The correlation between time and state of service should be measured. Timed automata model is an extension of the FSM model which allows modeling real-time systems and formal reasoning. So, the building of the transaction service model is learned from the theory of timed automata.

Definition 1(timed automata, TA [14]). *A timed automata model is a tuple $(Q, \Sigma, C, \delta, s_0, F)$, where Q is a finite, non empty set of states; Σ is an alphabet; C is a finite set of clocks; $\delta : Q \times \Sigma \times \phi(C) \times 2^C \to Q$ is the transition function in which $\phi(C)$ is the set of clock constraints; $s_0 \in Q$ is the initial state; F is a non empty set of final states.*

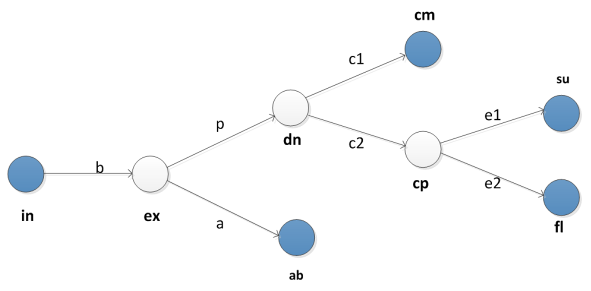

Figure 1 Comparison of service selection efficiency.

Definition 2(transaction service model, TSM). *A TSM is a TA of the form $(Q, I, O, C, \delta, s_0, F)$, where $Q : ST \times OP$ is a finite, non empty set of states in which OP is the set of operations and ST is the set of transaction states; I is a non empty set of input commands, and O is a non empty set of output messages; $I \times \mathrm{Pr} \times O$ is the alphabet of the TA; C is a finite set of clocks; $\delta : Q \times I \times Pr \times O \times \phi(C) \times 2^C \rightarrow Q$ is the transition function, which represents the evolution from a state to another for a given tuple $< \mathrm{inputcommand}, pr, \mathrm{outputmessage}, \lambda, \delta >$, in which the set $\lambda \subseteq C$ gives the clocks to be reset with this transition and δ is a clock consraint over C; $s_0 \in Q$ is the initial state; $F : ST_f \times OP \subseteq Q$ is a non empty set of final states, i.e., states where the client can terminate its interaction with the transaction service. A transaction service can be represented as a TSM.*

In the model, ST is the transaction states set including *initial (in), executing (ex), done (dn), committed (cm), compensated (cp), aborted (ab), succeeded (su)* or *failed(fl)*, which is determined by a set of operations (OP) and service primitives (or primitive operations, *Pr*). *Pr*= {*begin (b), precommit (p), commit (c1), compensate (c2), abort (a), evaluate1 (e1), evaluate1 (e2)*,θ}, where *'evaluate'* can be regarded as the actualization of *'compensate'* and θ is NOP (No Operation). A primitive moves the service from one state to another. The legal final state, *stf*, belongs to one of the states $ST_f = \{in, cm, ab, su, fl\}$. The $ST_{f+} = \{cm, su\}$ represents the set of successful states and $ST_{f-} = \{fl, ab\}$ is that of failed. When it is not executed, the final state is initial state *'in'*.

The sets of states and primitives can be represented by a state transition diagram as Fig. 1. It reflects the two-phrase committing process of transaction. A web service executes along the partial track of the transaction diagram.

4.2. Model of transaction service composition

For the basic process structure, it proposes the model of transaction service composition by single composition operator. Three types of composition are shown in accordance with the operator.

(1)Sequence Composition

Composition Operator 'Sequence' denotes two services can execute successively, which is the output of service1 (se$_1$) is the input of se$_2$. So the new service combined by se$_1$ and se$_2$ is Sequence (se_1, se_2), whose formal model $\boldsymbol{TSM_{S(se_1,se_2)}}$ is defined in the below.

$TSM_{S(se_1,se_2)} = (Q_{S(se_1,se_2)}, I_{S(se_1,se_2)}, O_{S(se_1,se_2)}, C_{S(se_1,se_2)}, \delta_{S(se_1,se_2)}, s_{0S(se_1,se_2)}, F_{S(se_1,se_2)})$, where the set of states $Q_{S(se_1,se_2)} = Q_{se_1} \cup Q_{se_2} - \{s_{0_{se_2}}\}$; the set of input commands $I_{S(se_1,se_2)} = I_{se_1} \cup I_{se_2}$; the set of output messages $O_{S(se_1,se_2)} = O_{se_1} \cup O_{se_2}$; the alphabet $I_{S(se_1,se_2)} \times Pr \times O_{S(se_1,se_2)} = I_{se_1} \times Pr \times O_{se_1} \cup I_{se_2} \times Pr \times O_{se_2}$; the set of clocks $C_{S(se_1,se_2)} = C_{se_1} \cup C_{se_2}$; the transition function $\delta_{S(se_1,se_2)} = \delta_{se_1} \cup \delta_{se_2} \cup \{s_{rse_1} \times \phi_{se_2}(C_{se_2}) \times 2^{C_{se_2}} \times i_{0se_1}/pr/o_{0se_2} \rightarrow s_{1se_2}\} - \{s_{0se_2} \times \phi_{se_2}(C_{se_2}) \times 2^{C_{se_2}} \times i_{0se_2}/pr/o_{0se_2} \rightarrow s_{1se_2}\}, pr \in Pr$; the initial state $s_{0S(se_1,se_2)} = s_{0se_1}$; the set of final states $F_{S(se_1,se_2)} = F_{seS_1} \cup F_{se_2} - \{s_{rse_1}\}$, $\exists r_{se_1} \in F_{se_1} = \{s_{rse_1}, \cdots, s_{tse_1}\}$.

(2) Parallel Composition

Composition Operator 'Parallel' denotes two services execute concurrently. The new service combined by se_1 and se_2 is Parallel (se_1, se_2), which is implemented by setting a new initial state s_{0P} (acting as the synchronization state). Its formal model $\boldsymbol{TSM_{P(se_1,se_2)}}$ is defined in the following.

$TSM_{P(se_1,se_2)} = (Q_{P(se_1,se_2)}, I_{P(se_1,se_2)}, O_{P(se_1,se_2)}, C_{P(se_1,se_2)}, \delta_{P(se_1,se_2)}, s_{0P(se_1,se_2)}, F_{P(se_1,se_2)})$, where the set of states $Q_{P(se_1,se_2)} = Q_{se_1} \cup Q_{se_2} \cup \{s_{0P_{se_2}}\}$; the set of input commands $I_{P(se_1,se_2)} = I_{se_1} \cup I_{se_2} \cup \epsilon$; the set of output messages $O_{P(se_1,se_2)} = O_{se_1} \cup O_{se_2} \cup \epsilon$; the alphabet $I_{P(se_1,se_2)} \times Pr \times O_{P(se_1,se_2)} = I_{se_1} \times Pr \times O_{se_1} \cup I_{se_2} \times Pr \times O_{se_2} \cup \{\epsilon/\theta/\epsilon\}$, where ϵ is null string; the set of clocks $C_{S(se_1,se_2)} = C_{se_1} \cup C_{se_2}$; the transition function $\delta_{P(se_1,se_2)} = \delta_{se_1} \cup \delta_{se_2} \cup \{s_{0P(se_1,se_2)} \times \phi_{se_1}(C_{se_1}) \times 2^{C_{se_1}} \times /\epsilon/\theta/\epsilon \rightarrow s_{0se_1}, s_{0P(se_1,se_2)} \times \phi_{se_2}(C_{se_2}) \times 2^{C_{se_2}} \times \epsilon/\theta/\epsilon \rightarrow s_{0se_2}$; the initial state $S_{0P(se_1,se_2)}$; the set of final states $F_{P(se_1,se_2)} = F_{se_1} \cup F_{se_2}$.

(3) Condition Composition

Composition Operator 'Condition' denotes a service executes only when certain condition (ψ) is satisfied with. So an additional initial state s_{0C} needs be added. The new service is Condition (ψ, se), whose formal model $\boldsymbol{TSM_{C(\psi,se)}}$ is thus defined.

$TSM_{C(\psi,se)} = (Q_{C(\psi,se)}, I_{C(\psi,se)}, O_{C(\psi,se)}, C_{C(\psi,se)}), \delta_{C(\psi,se)}, s_{0C(\psi,se)}, F_{C(\psi,se)}$, where the set of states $Q_{C(\psi,se)} = Q_{se} \cup \{s_{0C(\psi,se)}\}$; the set of input commands $I_{C(\psi,se)} = I_{se} \cup \{\psi\}$; the set of output messages $O_{C(\psi,se)} = O_{se} \cup \{\epsilon\}$; the set of clocks $C_{C(\psi,se)} = C_{se}$; the alphabet $I_{C(\psi,se)} \times Pr \times O_{C(\psi,se)} = I_{se} \times Pr \times O_{se} \cup \{\psi/\theta/\epsilon\}$; the transition function $\delta_{C(\psi,se)} = \delta_{se} \cup \{s_{0C(\psi,se)} \times \phi_{se}(C_{se}) \times 2^{C_{se}} \times \psi/\theta/\epsilon \rightarrow s_{0se}\}$; the initial state $s_{0C(\psi,se)}$, the set of final states $F_{C(\psi,se)} = F_{se}$.

Complex composition of web services can be achieved by these combinations of the three operations once or many more times.

5. Selection algorithm based on TSM and QoS

This section proposes the transaction QoS selection algorithm(**TQSA**).It can apply to dynamic service composition when service selection and execution perform simultaneously. In the algorithm, on one hand, the selection strategy is based on the QOS service composition and has a mechanism of transaction compensating, on the other hand, by recording the state of affairs real-time, it can unify with the quality of service and give a feedback to the subsequent choice of services.

5.1. Service selection algorithm

The algorithm-**TQSA** is a parallel algorithm, which is implemented by QoS service selection algorithm (**QSA**) and transaction compensation selection algorithm (**TSA**) collaboratively.

In **QSA**, we choose the three basic attributes: τ (execution time), r (reliability), c (cost) as the criterion of QoS. For each service provider, the values of the three attributes are their average statistical numbers. For the three composition process structure, the values of QoS are calculated according to the following terms.
Sequence: $\tau_s = \tau_{se_1} + \tau_{se_2}, c_s = c_{se_1} + c_{se_2}, r_s = r_{se_1} + r_{se_2}$.
Parallel: $\tau_p = \max\{\tau_{se_1}, \tau_{se_2}\}, c_p = c_{se_1} + c_{se_2}, r_p = \min\{r_{se_1}, r_{se_2}\}$.
Condition: $\tau_c = \tau_{se_i}, c_c = c_{se_i}, r_c = r_{se_i}$ (when se_i satisfy ψ).

In order to unify the quality of service metrics, each attribute value is normalized. 'm' is the number of the candidate service providers.

$$\tau_{se_i} = \frac{\tau_{se_i}}{\sum_{j=1}^{m} \tau_{se_j}}, c_{se_i} = \frac{c_{se_i}}{\sum_{j=1}^{m} c_{se_j}}, r_{se_i} = \frac{r_{se_i}}{\sum_{j=1}^{m} r_{se_j}}.$$

QoS service selection will search the executing service series of best utility. The series should meet specific global constraints and it is chosen from distributed multi-candidate service providers. Such problem was proved to be NP-complete [15]. To address it, we adopt genetic algorithm. Algorithm 1 shows the algorithm **QSA**, which tries to acquire a schedule that can optimize the utility of service composition. The input includes the composition service structure obtained from **TSM**, used to code the gene. The evolutionary parameters such as mutation probability, crossover probability and population size are also involved. The output is the selected service series.

Line 1 constructs encoding of the gene. The individual is a binary coding denoting the candidate providers, its length $l = \sum_{i=1}^{n} \left(\sum_j \lceil \log_2 \mathrm{NUM}(p_{ij}) \rceil + 1 \right)$. The code ranges from 1 to $2^l - 1$. '0' denotes no provider for the task. 'p_{ij}' denotes the j-th provider for task i.The length of the code can suit for the increasing of candidate providers dynamically.

```
   Input: TSMcs
   Output: SS
1  Code(TSMcs);
2  $g \leftarrow 0$;
3  Initpop Pg;
4  while (not sat endcond) do
5      FitnessEvaluate(Pg);
6      Cross(Pg);
7      Mutate(Pg);
8      Evolve(Pg);
9      $g \leftarrow g + 1$;
10 end
11 SS←DecodeOptimal(Pg);
12 return SS;
```

Algorithm 1: Algorithm QSA

Line 3 chooses the initial population of individuals. For the condition process, if the provider's code of the task satisfying the condition is not equal '0', the other providers' codes in the process will be set to'0'.

From line 4 to line 8 is a loop when the end condition is not satisfied. It breeds new individuals through crossover and mutation operations to give birth to the offspring. 'Evolve' aims to select the best-fit individuals in line 8.

The utility value is evaluated by fitness-evaluate function in Line 5, where $\mathrm{Fitness} = p_1 r / (p_2 \tau + p_3 c)$, $\sum_{i=1}^{3} p_i = 1$, in which the value of 'c' is obtained from providers'τ' is calculated in the process of execution, and 'r' is determined by the transaction state. The latter two can result from the algorithm-**TSA**.

Algorithm 2 shows the algorithm **TSA**, which gives the values of 'τ' and 'r' by monitoring the service execution. It also provides the compensating strategy when certain service fails. The input is the initial selected service series and the final series could be changed while the service composition executing. The output of '*stf*' has a mapping set to the value of reliability(r) according to the Tab. 1.

From line 1 to line 3 is a loop to construct the initial state transition diagram of service composition. Line 7 gives the calculation method of committed service successfully. As for the abort eservice, the composite service needs to re-select services for execution to start the compensation process from line 10 to 14. If the service fails again, there may be compensation repeatedly for the task within a limited time from line 15 to 25. The computation of illegal *stf* is given from line 27 to 36. When the task finishes, the service composition will update.

In Table 1, the top row represents the state of service's execution, and the value of reliability is expressed below. The parameter value is set to $0 < k_i < 1 (i = 1 \cdots 5)$ and $k_j < k_l$ when $j < l$.

```
   Input: $SS(se_1, se_2, \cdots, se_n)$
   Output: $stf$, $\tau$
1  for $i \leftarrow 1$ to $n$ do
2      $pr_i = \theta$, Construct $TSMcs$;
3  end
4  for $i \leftarrow 1$ to $n$ do
5      while $t_i \le t_{imax}$ & $stf_i \notin ST_{f+}$ do
6          switch $(pr_i)$ do
7              case $c1$:
8                  $\tau_i = \phi(c_1) - \phi(b)$, $stf_i$=cm;
9                  Update $TSMse_i$, Update $TSMcs$;
10             case $a$:
11                 $\tau_i = \phi(a) - \phi(b)$, $stf_i$=ab;
12                 Update $TSMse_i$, $P_i < -P_i - SS(se_i)$;
13         endsw
14         SS=QAS(TSMcs),$pr_i = c_2$;
15         while ($stf_i \in ST_{f-}$) do
16             switch $(pr_i)$ do
17                 case $e_1$:
18                     $\tau_i = \phi(e_1) - \phi(c_2)$, $stf_i$=su;
19                     Update $TSMse_i$, Update $TSMcs$;
20                 case $e_2$:
21                     $\tau_i = \phi(e_2) - \phi(c_2), stf_i = fl$;
22                     Update $TSMse_i$,
                       $P_i < -P_i - SS(se_i)$;
23             endsw
24             SS=QAS(TSMcs),$pr_i = c_2$;
25         end
26     end
27     if $stf_i$=in then
28         switch $(pr_i)$ do
29             case $c_2$:
30                 $stf_i$=cp,$\tau_i = \phi(c_2) - \phi(p)$;
31             case $p$:
32                 $stf_i$=dn,$\tau_i = t_{imax}$;
33             case $b$:
34                 $stf_i$=ex,$\tau_i = t_{imax}$;
35         endsw
36     end
37     if $stf \notin ST_{f+}$ then
38         Update $TSMcs$;
39     end
40 end
41 return $stf$, $\tau$;
```

Algorithm 2: Algorithm **TSA**

Table 1 The value of reliability(r)

stf	ab	fl	ex	cp	dn	su	cm
r	0	k_1	k_2	k_3	k_4	k_5	1

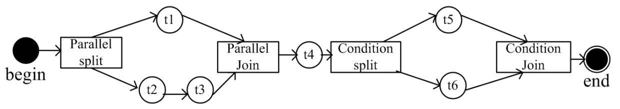

Figure 2 Service composition instance.

5.2. Optimization of the algorithm

In order to enhance the success rate of the composite service, we improve the evolution strategy in the solution of **QSA** to form the algorithm-**I-TQSA**. Algorithm 3 shows the detail of evolution algorithm. If the reliability value of individual is greater than 'k', then accept the individual with a certain probability ρ in line 3, 4.

```
   Input: $P_g$
   Output: $P_{g+1}$
1  $df$ = FitnessEvaluate($p_{gn}$) - FitnessEvaluate($p_{gmin}$);
2  for $j \leftarrow 0$ to $N$-$1$ do
3      if ($df$>0) or (random(0,1)< $\rho$ & $r(p_{gn}) > k$) then
4          Replace $p_{gmin}$ with $p_{gn}$ in $P_{g+1}$;
5      else if $\exp(df/|\mathrm{Fitness}(p_{gn})|) > \mathrm{random}(0,1)$
       then
6          Replace $p_{gmin}$ with $p_{gn}$ in $P_{g+1}$;
7  end
8  return $P_{g+1}$;
```

Algorithm 3: Algorithm **Evolve**

To avoid the genetic algorithm into a local optimum in search process, the algorithm accepts the deterioration of solution in a certain probability in line 5, 6. These above two points show the difference from traditional evolution strategy of choosing the best.

6. Evaluation of the performance

The instance shown in Fig. 2 was adopted to verify the validity of the algorithm. The testing services are built on pastry platform of PeerSim [16], and the web service information of same function can be stored in the same node by its special routing mechanism, which is suitable for large-scale services selection. The running environment is CPU: Intel 2.40 GHZ, RAM: 3.0 GB.

In Fig. 2, the number of tasks $n = 6$, the global constraints: $t_{max} = 10, c_{max} = 20$, the local constraint $t_{imax} = 5$. Each service costs meet the random distribution on $[0, 15]$. The executing states have a uniform distribution over the sets of $[cm, ab, ex, dn, cp, su, fl]$, the clock set $C = [0, 10]$. For each service provider, $\phi(C)$ is the legitimate random mapping from the state to the clock.

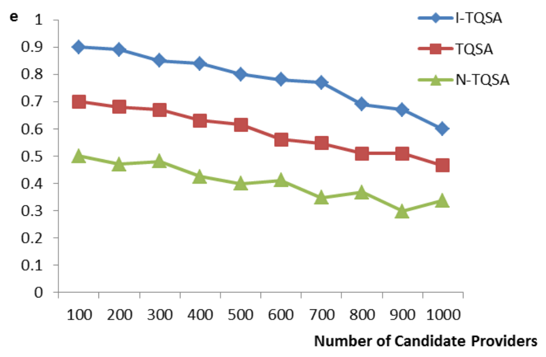

Figure 3 Comparison of service selection efficiency.

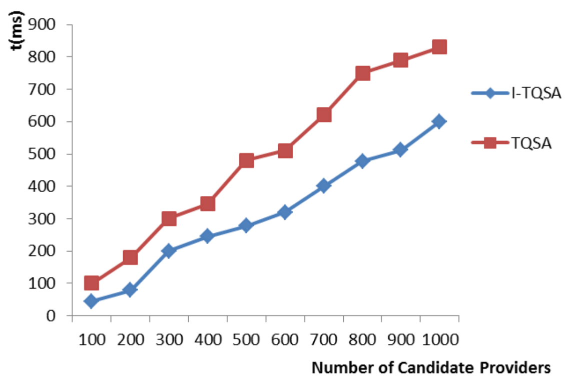

Figure 4 Comparison of service selection time.

The efficiency of the algorithm is measured by 'e', which $e = |s|/|S|$. $|S|$ is the times of service composition, and $|s|$ is the times of successful selection for service composition.

$$s = \begin{cases} 1, & stf \in ST_{f+} \\ 0, & others \end{cases}$$

The improved evolutionary strategy TQSA algorithm (**I-TQSA**), TQSA parallel algorithm (**TQSA**) and basic QSA algorithm (**N-TQSA**) were compared in the following, where $|S| = 500$, the number of candidate providers for each task ranges from 100 to 1000, $k_1 = 0.1, k_2 = 0.2, k_3 = 0.3, k_4 = 0.6, k_5 = 0.9, p_1 = p_2 = p_3 = 1/3, \rho = 0.8, k = 0.6$. The crossover rate of genetic algorithm is 0.8, and the mutation rate is 0.05.

As shown in Fig. 3, using **TQSA** can improve the reliability of the service composition with compensation, and **I-TQSA** can further enhance the effectiveness.

The selection time of using **TQSA** and **I-TQSA** is shown in Fig. 4.

The results demonstrate that the select times are both prolonged with the providers' number increasing. Algorithm **I-TQSA** greatly reduces the computation time while maintains the high reliability.

7. Conclusions

In this paperwe explore the transaction service selection problem for dynamic web service composition. In order to improve the reliability of WSC and reduce the searching time, the transaction QoS selection algorithm (**TQSA**) and its improved algorithm (**I-TQSA**) are proposed, and we mainly make three contributions which are summarized as follows:

1) A transaction service composition model (**TSM**) is proposed to combine the transaction state and QoS. The model provides a proper mechanism to aggregate the qualitative transaction criteria and quantity of service. It also models the behavior of real-time systems over time to evaluate the dynamic web service composition.

2) Base on TSM, a QoS-aware with transaction compensating selection algorithm (**TQSA**) is proposed to implement dynamic service composition by combing **QSA** and **TSA** algorithm. The algorithm **TQSA** greatly improves the validity of the service selection.

3) The evolution strategy is improved to optimize the searching process of GA. Experiment results show the algorithm **I-TQSA** can improve the efficiency of selection and reduce composition time further.

Acknowledgement

This work has been supported by the National Science Foundation of China (Grant No. 61101214, 60972140), Beijing Natural Science Foundation (Grant No. 3102009) and the Fundamental Research Funds for the Central Universities (Grant No. 20100842002, 20110842001). Thanks for the help.

References

[1] B. Al-Shargabi, A. Sabri and A. El Sheikh, Web Service composition survey: state of the art review, Recent Patents on Computer Science, **3**(2), 91-107 (2010).

[2] W. Jiang, T. Wu and S. HuQoS–aware automatic service composition: a graph view, Journal of Computer Science and Technology, **26**(5), 837-853 (2011).

[3] XZhao, B. Song and P. Huang, An improved discrete immune optimization algorithm based on PSO for QoS-driven web service compositionApplied Soft Computing, **12**(8), 2208-2216 (2012).

[4] M. Liu, M. Wang and W. Shen, A quality of service (QoS)-aware execution plan selection approach for a service composition process, Future Generation Computer Systems, **28**(7), 1080-1089 (2012).

[5] K. Rajaram, A. Adiththan and C. Babu, Tx-policy: transactional policies for reliable web service composition, ICWET 2011, 738-743 (2011).

[6] G. Le Gao, S. Urban and J. Ramachandran, A survey of transactional issues for web service composition and recovery, International Journal of Web and Grid Services, **7**(4), 331-356 (2011).

[7] H. Liu, Z. Zheng and W. Zhang, A global graph-based approach for transaction and QoS-aware service composition, KSII Transactions on Internet and Information Systems, **5**(7), 1252-1273 (2011).

[8] C. Lin, R. Sheu and Y. Chang, A relaxable service selection algorithm for QoS-based web service composition, Information and Software Technology, **53**(12), 1370-1381 (2011).

[9] C.Y. Zhang, LX. Shao and WQ. Li, Self-organizing mechanism for cloud services and performance analysis, China Communications, **9**(6), 135-144 (2012).

[10] C.Y. Zhang, K. Huang, X. Cui and YF. Chen, Energy-aware GPU programming at source-code levels, Tsinghua Science and Technology, **17**(3), 278-286 (2012).

[11] I.D. Pietro, F. Pagliarecci and L. Spalazzi, Model checking semantically annotated service, IEEE Transactions on Software Engineering, **38**(3), 592-608 (2011).

[12] J.J. Hu and X. Zhao, A service composition model with characteristic of transaction based on finite state machine, The 2008 International Conference on Computer and Electrical Engineering, ICCEE 2008, 450-454 (2008).

[13] G. Fan, H. Yu, L. Chen and R. Tong, An approach to analyzing time constrained service composition, Journal of Computers, **6**(8), 1723-31 (2011).

[14] R. Alur, Timed automata, computer aided verification, Lecture Notes in Computer Science, 1633, 8-22 (1999).

[15] T. Yu, Y. Zhang and K. Lin, Efficient algorithms for web services selection with end-to-end QoS constraints, ACM Transaction on Web, **1**(1), 16-32 (2007).

[16] http://code.google.com/p/ PeerSim-Pastry.

Hu jingjing received the PhD degree in computer science from Beijing Institute of Technology, Beijing, China. She is currently a lecturer in the school of Software of Beijing Institute of Technology. Her research interests are in the areas of service computing, multi-agent systems, and GPU-based computer tomography.