2014 IEEE 7th International Conference on Service-Oriented Computing and Applications

QoS-aware Service Composition in Mobile Environments

Nguyen Cao Hong Ngoc, Donghui Lin, Takao Nakaguchi, Toru Ishida
Department of Social Informatics, Kyoto University
Yoshida-Honmachi, Sakyo-Ku, Kyoto, 606-8501, Japan
Email: nchngoc@ai.soc.i.kyoto-u.ac.jp, {lindh, nakaguchi, ishida}@i.kyoto-u.ac.jp

***Abstract*—Service composition involves combining many existing services to solve a complex task in the service-oriented environment. Due to the progress in modern smart devices, mobile devices can now act as service providers. The mobility of mobile service providers and the dynamism of the mobile environment are unique features. As a result, there are two issues with service composition in mobile environments: (1) how to use the mobility of devices as an non-functional Quality of Service (QoS) criterion, and (2) how to deal with the dynamism of the mobile environment to realize the optimal service composition. In order to deal with above issues, we proposed a QoS-aware service composition model for service composition in mobile environments considering the feature of mobility in the mobile environment. Moreover, we proposed an adaptive approach called k-neighbor. The main idea of k-neighbor approach is to decompose the composite service into smaller elementary composite services then to set up a solution plan for those elementary composite services based on the available provider set. The evaluation shows that the way of decomposing a composite service adapts to different dynamism of the environment.**

***Keywords*-service composition; quality of service; mobility; mobile environment**

I. Introduction

Service composition is the combination of many existing services to achieve a larger and more complex task. There are many research projects on service composition have been published in recent years. For example, to find exactly the optimal execution plan, Zeng *et al.* [12] use Linear Integer Programming, Gao *et al.* [4] use Dynamic Programming for linear objective function while Aiello *et al.* [1] apply the Backward Breadth First algorithm and Wan *et al.* [9] adopt Divide-and-Conquer based selection in the case of non-linear objective function.

In traditional service composition, an implicit assumption is that services are running on heavyweight enterprise servers and so can support computation-intensive functions. However, modern mobile devices, mainly smart phones, equipped with an array of sensors and powerful computation-power, may break this assumption and context. The services provided by mobile phones will be quite different from conventional computation-intensive services. They could be moving location-based or context-aware services sensing and provide, through their sensors, immediate real world information. There are studies on service composition that examine various aspects such as load balance and stability [7], resource [5] [11], mobility [3] [6] [8] [10]. One of the key differences between mobile and traditional environment is the variation in Quality of Service (QoS) due to the mobility [2].

In the field of service composition in mobile environments, Wang [10] has considered the scenario in which the interval of time that service providers are still available in the current environment is utilized instead of predicting their future locations. For example, mobile devices carried by passengers in a subway form a mobile ad hoc network through Bluetooth or Wi-Fi communication. The available time of the services (such as file download or GPS services) provided by those mobile devices is determined by the time that those passengers spend in the subway. However, the previous research mainly focused on the impact of the mobility of service providers on reliability. We note that in the field of service composition in mobile environments, not only the mobility but also other traditional QoS criteria such as execution cost and reputation should be considered as well. Moreover, because of the mobility of service providers, the dynamism of the environment will depend on the number of available providers, the frequency of new arrivals, and the available duration. The composition approach has to be able to adapt to each level of dynamism of the environment.

In summary, there are two issues with service composition in mobile environments: (1) how to associate the traditional QoS criteria to the mobility of service providers, and (2) how to deal with the dynamism of the mobile environments to realize the optimal service composition. Thus, to deal with these issues, this paper proposes a QoS-aware service composition model by associating traditional QoS criteria to the mobility of mobile providers and an approach that is adaptive to the dynamic environments.

The remainder of the paper is organized as follows: Section II describes the problems. Section III describes the proposed model and our service composition approach. We then report computation results in Section IV. Finally, Section V discusses related work and Section VI concludes the paper.

II. Problem description

In the scenario that we mentioned in Section I, mobility prediction information can be disseminated through any underlying service discovery protocol [10]. Hence, one can

978-1-4799-6833-6/14 $31.00 © 2014 IEEE
DOI 10.1109/SOCA.2014.51

Table I
A TYPICAL EXAMPLE OF SERVICE COMPOSITION IN MOBILE ENVIRONMENT

$\mathbf{p_i}$	$\mathbf{t_{0i}}$	$\mathbf{t_{ai}}$	$\mathbf{t_{bi}}$	$\mathbf{s_i}$	Execution duration	Execution cost	Reputation
p_1	0	10	13	s_2	7	4	0.3
p_1	0	10	13	s_3	1	2	0.7
p_2	0	10	13	s_1	5	3	0.3
p_2	0	10	13	s_2	6	3	0.4
p_3	3	12	16	s_2	2	2	0.5
p_3	3	12	16	s_3	2	2	0.7

know the time interval in which mobiles are always available and the moment that mobiles will be unavailable in the current environment. Note that if a mobile provider has a micro-mobility (comparing to the mobile requester), we consider that the provider is present in the current environment. In particular, providers are unavailable when they are outside the current environment or shutdown due to battery depletion. Besides, the research is limited to the strictly sequential service composition.

Consider a successive service composition problem where a service composition request consists of n different services s_j $(1 \leq j \leq n)$ with a strictly defined invocation sequence $s_1 \rightarrow s_2 \rightarrow \ldots \rightarrow s_n$. Suppose that when the service composition request arrives, there is a set of m available service providers $P = \{p_1, p_2, \ldots, p_m\}$. Each service provider p_i has time of arrival t_{0i} and reports time interval $(t_{ai}, t_{bi}]$. It means that the provider p_i is always available in $[t_{0i}, t_{ai}]$; it may be unavailable at any time in $(t_{ai}, t_{bi}]$; and it will be unavailable in (t_{bi}, ∞). It is also assumed that when p_i is available, all services provided by p_i are available, too.

Consider the example of a composition request with three services as a sequence $s_1 \rightarrow s_2 \rightarrow s_3$ and three service providers $\{p_1, p_2, p_3\}$. p_1 arrives at $t_{01} = 0$, reports time interval (10, 13], and can provide services $\{s_2, s_3\}$. p_2 arrives at $t_{02} = 0$, reports time interval (10, 13], and can provide services $\{s_1, s_2\}$. p_3 arrives at $t_{03} = 3$, reports time interval (12, 16], and can provide services $\{s_2, s_3\}$. It is often the case that execution duration, execution cost and reputation are used to describe service quality. Details of each provider's services are showed in Table I.

An execution solution of the composite service, denoted by $X = ((x(s_j), s_j); 1 \leq j \leq n)$ is a vector indicating the selection of service providers for all atomic services of the composite service. $x(s_j)$ indicates the selected service provider for the service s_j. Traditionally, when the service composition request arrives (current time is zero), the system recognizes that there are two available service provider, p_1 and p_2. Only provider p_2 provides service s_1 and only provider p_1 provides service s_3. Therefore $x(s_1) = p_2$ and $x(s_3) = p_1$. For service s_2, there are two providers p_1 and p_2. However, service s_2 from provider p_2 is superior to service s_2 from provider p_1 in all criteria (such as the execution duration, the execution cost and the reputation). Traditionally, provider p_2 is chosen for service s_2. Hence, the execution solution $X = ((p_2, s_1), (p_2, s_2), (p_1, s_3))$ is chosen. However, when the first service, s_1, finishes (after 5 units of time), provider p_3 has become available. Service s_2 from provider p_3 is now the superior choice. Moreover, if service s_2 is executed on provider p_3, it is executed completely in the safe period $[t_{03}, t_{a3}]$ of the provider p_3. But if service s_2 is executed on provider p_1 (or p_2), there are two units (or one unit) of time that the service s_2 is executed in the unsafe period $(t_{a1}, t_{b1}]$ (or $(t_{a2}, t_{b2}]$) of the provider p_1 (or p_2). Thus, it is better to choose provider p_3 for service s_2. For the service s_3, all criteria of service s_3 from p_1 are better than those of service s_3 from p_3. However, service s_3 has to be executed completely in the unsafe period $(t_{a1}, t_{b1}]$ of provider p_1. Recall that a provider may be unavailable at any time during its unsafe period. Thus, is it better to choose service provider p_1 or service provider p_3 for service s_3? Moreover, consider the environment in which many service providers exist and they will be available for a long time (the dynamism of the environment is low) and the environment in which mobile providers are only available for a short time but new providers arrive frequently, which strategy should we apply, set up the solution plan for all services before execution or select a provider each time an atomic service has to be executed?

As the above example shows, there are two issues with the service composition in mobile environments:

- How to associate the traditional QoS criteria to the mobility of service providers?
- How to deal with the dynamism of the mobile environment to realize the optimal service composition?

In conventional QoS model, all criteria are static. This means that their values are determined before selecting providers for atomic services of the composite service. Although some criteria may change during service execution, such as a change in execution cost or the disappearance of some services, these changes are not considered in this paper. Only the criteria whose values are changed depending on which service providers are selected for prior services are considered. Those criteria are so-called dynamic criteria. In this scenario, the ratio of the time that a service is executed in the safe period $[t_{0i}, t_{ai}]$ of its provider and its execution duration is dynamic. For example, if $X = ((p_2, s_1), (p_3, s_2), (p_3, s_3))$, service s_3 has to be executed within two units of time in $[t_{03}, t_{a3}]$ for a total of two units of duration execution. If $X = ((p_2, s_1), (p_2, s_2), (p_3, s_3))$, service s_3 has to be executed within one units of time in $[t_{03}, t_{a3}]$ for a total of two units of duration execution. The concept of dynamic criteria is used in the service composition model in order to adapt to the mobile environments.

III. Service composition in mobile environments

In previous section, two issues about service composition in mobile environments were described. In order to deal with those issues, this section provides a formal description of the model and our approach.

A. QoS model

For each service, the quality of service s_j provided by mobile provider p_i can be represented as $q(p_i, s_j) = (q_1(p_i, s_j), q_2(p_i, s_j), q_3(p_i, s_j), q_4(p_i, s_j), q_5(p_i, s_j))$ where $q_1(p_i, s_j) = t(p_i, s_j)$, $q_2(p_i, s_j) = c(p_i, s_j)$, $q_3(p_i, s_j) = r(p_i, s_j)$, $q_4(p_i, s_j) = z(p_i, s_j)$, $q_5(p_i, s_j) = a(p_i, s_j)$ correspond to execution duration, execution cost, reputation, successful execution rate and availability. These are the five criteria of an atomic service. The first four criteria are static and their definitions are taken from the model proposed by Zeng *et al.* [12]. The last criterion is dynamic and its definition has been modified to be suitable for mobile environments.

1) QoS criteria for atomic services: With regard to the five criteria of atomic services, we will recall the definitions of the first four criteria which are given in detail in [12] and specify the last criterion, availability.

(i) Execution duration
The execution duration of service s_j from provider p_i, denoted by $t(p_i, s_j)$, measures the period between the moment when a request is sent and the moment when the results are received.

(ii) Execution cost
The execution cost of service s_j from provider p_i, denoted by $c(p_i, s_j)$, is the fee that the service requester has to pay for invoking the service s_j.

(iii) Reputation
The reputation of service s_j from provider p_i, denoted by $r(p_i, s_j)$, is a measure of its trustworthiness. The value of the reputation is defined as the average ranking given to the service by end users.

(iv) Successful execution rate
The successful execution rate of service s_j from provider p_i, denoted by $z(p_i, s_j)$, is the probability that a request is correctly responded to. The value of the successful execution rate is the ratio of the number of times that service s_j has been successfully completed within the maximum expected time frame in the total recent Y invocations.

(v) Availability
Suppose that service providers for the first $j-1$ ($1 \leq j \leq n$) services have been chosen, the availability of service s_j from provider p_i, denoted by $a(p_i, s_j)$, is the possibility that service s_j can be executed completely by provider p_i. To compute the value of availability, let us introduce two definitions:

a) Completion time
Suppose that service providers for the first j ($1 \leq j \leq n$) services have been chosen and provider p_i is chosen for service s_j, we define the completion time of service s_j, denoted by $C(s_j)$, as the time that all services up to and including s_j will be finished. So that $C(s_1) = t(p_i, s_1)$ and $C(s_j) = C(s_{j-1}) + t(p_i, s_j)$ if $j \geq 2$.

b) Safe duration
Suppose that service providers for the first $j-1$($1 \leq j \leq n$) services have been chosen, the safe duration of service s_j from provider p_i, denoted by $e(p_i, s_j)$, is the duration in which service s_j can be executed in a safe period $[t_{0i}; t_{ai}]$ of provider p_i.
In cases of the first service, the following formula is used to compute the value of the safe duration:

$$d(p_i, s_1) = \begin{cases} t(p_i, s_1) & \text{if } t_{ai} \geq t(p_i, s_1) \\ t_{ai} & \text{if } t_{ai} < t(p_i, s_1) \end{cases} \quad (1)$$

In case of other services, the following formula is used:

$$d(p_i, s_j) = \begin{cases} t(p_i, s_i) & \text{if } t_{ai} \geq C(s_{j-1}) + t(p_i, s_j) \\ t_{ai} - C(s_{j-1}) & \text{if } C(s_{j-1}) < t_{ai} < C(s_{j-1}) + t(p_i, s_j) \\ 0 & \text{if } t_{ai} \leq C(s_{j-1}) \end{cases} \quad (2)$$

When selecting provider p_i for service s_j, only providers which are available at the completion time $C(s_{j-1})$ of the previous service should be considered. There are some cases showed in Figure 1.
At $C(s_{j-1})$, provider p_i is in its 100% available period. It is good if service s_j executed by provider p_i can be finished before t_{ai} because service s_j will be surely completed (Case 1 in Figure 1). In a worse situation, service s_j has to be executed in an unsafe period for a short time $(t_{ai}, C(s_{j-1}) + t(p_i, s_j)]$. The service may be finished or unfinished and the risk of incompleteness depends on how long service s_j spends in the unsafe period (Case 2 in Figure 1). We will not consider the case that the execution duration needed by service s_j is bigger than the duration in which provider p_i is available because it cannot be completed.
At $C(s_{j-1})$, provider p_i is in unsafe period $(t_{ai}, t_{bi}]$ so if service s_j is executed by provider p_i, it is completely at risk for its entire execution duration (Case 3 in Figure 1). This is the worst case. We define its availability as zero.
In general, the value of availability is computed using the following formula:

$$a(p_i, s_j) = \frac{d(p_i, s_j)}{t(p_i, s_j)} \quad (3)$$

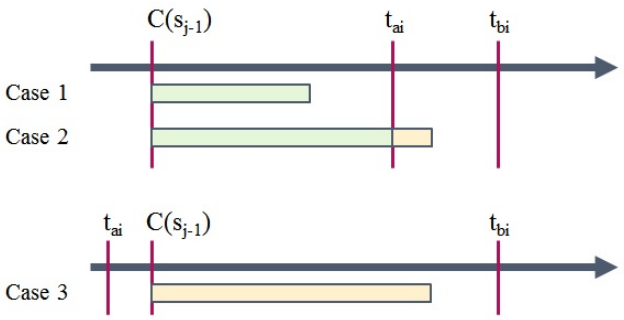

Figure 1. Cases of the availability

A QoS criterion can be classified as a positive or negative criterion. A criterion is positive (negative) if the higher value is, then the higher (the lower) quality is. The execution cost and the execution duration are negative criteria; all the others are positive.

2) Quality criteria for composite services: Given the execution solution $X = ((x(s_1), s_1), (x(s_2), s_2), \ldots, (x(s_n), s_n))$ where $x(s_j)$ indicates the selected service provider for service s_j, each criterion of the composite service is briefly defined below:

(i) Execution duration
The execution duration of the composite service, denoted by $t(X)$, is the sum of those of the atomic services.

(ii) Execution cost
The execution cost of the composite service, denoted by $c(X)$, is the sum of those of the atomic services.

(iii) Reputation
The reputation of the composite service, denoted by $r(X)$, is the average of those of the atomic services.

(iv) Successful execution rate
The successful execution rate of the composite service, denoted by $z(X)$, is the product of those of the atomic services.

(v) Availability
The availability of the composite service, denoted by $a(X)$, is the ratio of its safe duration and the execution duration.

The formula of these criteria are written in Table II.

3) QoS of composite services: In the model proposed by Zeng *et al.* [12], to compute the quality of the composite service, two phases of applying Simple Additive Weighting (SAW) are used: the scaling phase and the weighting phase. The five quality criteria discussed earlier are used. They are denoted by l, which has value from 1 to 5 with 1 = execution duration, 2 = execution cost, 3 = reputation, 4 = successful execution rate, and 5 = availability. In general, $q_l(X)$ denotes the quality criterion l $(1 \leq l \leq 5)$ of the composite service.

Table II
QOS CRITERIA OF THE COMPOSITE SERVICE

QoS criterion	**Formula**
Execution duration	$t(X) = \sum_{j=1}^{n} t(x(s_j), s_j)$
Execution cost	$c(X) = \sum_{j=1}^{n} c(x(s_j), s_j)$
Reputation	$r(X) = \frac{1}{n} \sum_{j=1}^{n} t(x(s_j), s_j)$
Successful execution rate	$z(X) = \prod_{j=1}^{n} t(x(s_j), s_j)$
Availability	$a(X) = \frac{1}{t(X)} \sum_{j=1}^{n} d(x(s_j), s_j)$

(i) Scaling phase
The execution duration and the execution cost are negative, i.e., the higher the value, the lower the quality. We use the following scaling formula:

$$v_l(X) = \begin{cases} \frac{max_l - q_l(X)}{max_l - min_l} & \text{if } max_l \neq min_l \\ 1 & \text{if } max_l = min_l \end{cases} \quad (4)$$

Other criteria such as the reputation, the successful execution rate and the availability are positive, i.e., the higher the value, the better the quality. For positive criteria, values are scaled according to the following formula:

$$v_l(X) = \begin{cases} \frac{q_l(X) - min_l}{max_l - min_l} & \text{if } max_l \neq min_l \\ 1 & \text{if } max_l = min_l \end{cases} \quad (5)$$

where

$$max_l = max(q_l(X))$$

$$min_l = min(q_l(X))$$

For example, to compute the value of max_1, we select the provider with longest execution duration for each service and sum up all these execution durations. In order to compute the value of min_1, we select the provider with the shortest execution duration for each service and sum up all these execution durations.

(ii) Weighting phase
The following formula is used to compute the quality score for the service provided by each service provider:

$$q(X) = \sum_{l=1}^{5} (v_l(X) * w_l) \quad (6)$$

w_l $(1 \leq l \leq 5)$ represents the weight of criterion l. End users express their preferences regarding QoS by providing values for the weights w_l.

B. Approaches

Consider a composite service consisting of the sequential execution of n atomic services. The main idea of the k-neighbor approach is to decompose the composite service into $\lceil n/k \rceil$ $(1 \leq k \leq n)$ smaller elementary composite services. Each elementary composite service includes at most k consecutive atomic services. Instead of giving a solution plan for all n atomic services at once when the invocation request arrives, we only give a solution plan for k consecutive services at a time. Once these k services are executed completely with the given solution, then the solution plan for the next k services will be provided. In the following part of this section, first, the k-neighbor algorithm will be introduced, then three special cases of the algorithm will be discussed in more detail.

Algorithm 1 k-neighbor algorithm

INPUT: Invocation request $s_1 \rightarrow s_2 \rightarrow \ldots \rightarrow s_n$ and k
OUTPUT: Execution solution X

```
1: P: set of current available providers
2: X': set of execution solution
3: q: quality of current k services
4: X ← ((∅, s_1), (∅, s_2), ..., (∅, s_n))
5: j ← 1
6: while j < n do
7:   if (j + k − 1) > n then
8:     j' ← n
9:   else
10:    j' ← (j + k − 1)
11:  end if
12:  P ← get_providers()
13:  X' ← generate_solutions(s_j, s_j', P)
14:  if is_empty(X') then
15:    return X
16:  else
17:    q ← 0
18:    for all X_i ∈ X' do
19:      q(X_i) ← calculate_qos(s_j, s_j', X_i)
20:      if q < q(X_i) then
21:        q ← q(X_i)
22:        Update providers for k services from s_j to s_j'
           in X with X_i
23:      end if
24:    end for
25:    Execute k services from s_j to s_j' with X
26:    j ← j + k
27:  end if
28: end while
29: return X
```

In Algorithm 1, $get\_providers()$ is used to update current available mobile service providers of set P. Thus $generate\_solutions(s_j, s_{j'}, P)$ will list all compositions from service s_j to service s_{j+k-1}. In this procedure, we have to ensure that all services from s_j to s_{j+k-1} can be finished if all currents providers $p_i \in P$ will not leave the current environment before t_{bi}. Each composition of k services (from s_j to s_{j+k-1}) is considered as a temporary composite service and its QoS value is calculated by two phases which are discussed in Section 3. The solution with the biggest QoS value will be chosen. All services from s_j to s_{j+k-1} sequentially executed in sequence before the next loop.

1) Offline approach: The offline approach is a specific case of the k-neighbor approach with $k = n$. This approach is also known as the global optimization [12] because the providers of all services are selected before their execution. If all information about the providers are known, the offline approach will achieve the optimal result because this approach has the global view of all possible compositions. Therefore, in static environments, the offline approach should be chosen to achieve the highest result.

To deliver a solution for n services, it is required to set up the plan only one time. Assume that there are on average m service providers in current environment, it has at most m^n compositions. To identify the optimal composition, the QoS values of all compositions must be calculated. Therefore, the complexity of the offline approach is $O(nm^n)$.

However, in mobile environments, service providers may often arrive and leave. In the offline approach, a solution plan is delivered when the invocation request arrives and the information is not update to reflect the arrival of new providers. The offline approach yields the best solution but its shelf-life is severely limited. Therefore, in dynamic environments, since providers arrive and leave frequently, it is very hard to achieve the highest result. Besides, if the composite service consists of many atomic services and the execution duration of the composite service is long, it is very difficult to select providers for the last few atomic services (for example the service s_{n-1} and the service s_n). This is because the execution duration needed for all services tends to exceed the duration that service providers stay in current environment and new service providers are not utilized.

2) Online approach: The online approach is a specific case of the k-neighbor approach with $k = 1$. The main idea of the online approach is to select a provider p_i for the service s_j each time the service s_j has to be executed. This approach is known as the local optimization [12]. The mobile environment is dynamic. For example in the scenario that we mentioned in the Introduction, people may get on and get off the train at each station. Therefore, the available provider set changes often It is better to update the provider set each time a new atomic service needs to be executed. Due to the frequent update, the online approach can adapt well to changes in the mobile environment but it may lead to the local optimal results. Moreover, in searching for the

best result, the online approach may choose a solution with a long execution duration. As a result, it may fail to choose a provider for the next service if the number of available providers is small.

To deliver a solution for n services, we need to set up the plan n times. If there are, on average, m service providers in current environment, there are at most m ways of choosing a provider for the considering service based on their QoS values. Therefore, it takes $O(m^2)$ to select a provider for each service.

3) Neighbor approach: If the offline approach is a global optimization and the online approach is a local optimization, the neighbor approach $(1 < k < n)$ is a local global-optimization. It updates the available provider set as well as having a further view when selecting a solution for the composite service.

To deliver a solution for n services, it is required to setup the plan $\lceil \frac{n}{k} \rceil$ times with the neighbor approach. Assume that there are on average m service providers in current environment, it has at most m^k ways to choose providers for k considering services. Therefore, it takes $O(km^k)$ to set up a solution plan each time.

IV. Evaluation

In this section, we evaluate the k-neighbor approach. The dynamism of the environment is low if many providers exist and all will remain available for a long time. Indeed, it can be considered as a static environment. The dynamism of the environment is high if mobile providers are only available for short period but new providers arrive frequently. This is considered as an extremely dynamic environment. To observe the effect of k values in different environments, one must not only compare the QoS values of the composite services but also analyze the percentage of successful composition among those ones.

In our evaluation, we fixed the number of atomic services in the composite service by five. Each provider can provide three (random assignment) of the required services. The first four quality criteria (execution duration, execution cost, reputation and successful execution rate) of an atomic service provided by each provider were generate randomly $(1 \leq t, c \leq 10;\ 0 < r, z < 1)$. The availability will be calculated when setting up a solution based on the Equation 3. To specify the dynamism of the environment, the number of arrival providers, the duration of stay in the current environment and the frequency of arrivals are increased at the same time. With this setting, we started with five available providers; each provider is available in the environment from five to nine units of time; and after every five units of time, new five providers arrive. This environment is considered as an extremely dynamic environment. All those parameters will be increased at the same time to observe the QoS value and the successful composition. In our evaluation, because there are five atomic

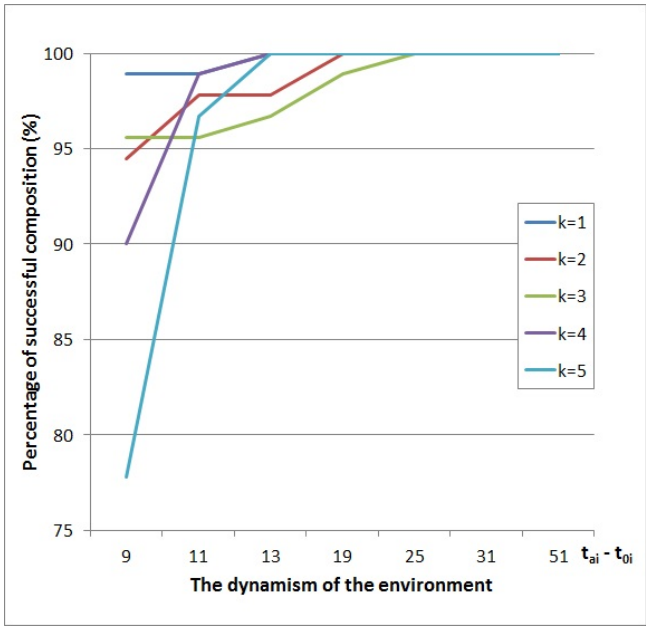

Figure 2. The percentage of successful composition (from left to right, the environment changes from dynamic to static)

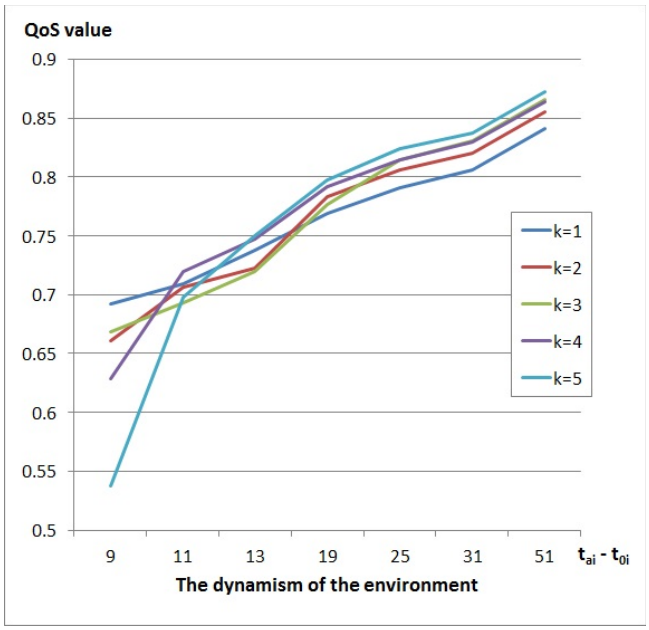

Figure 3. The average QoS value of the composite service (from left to right, the environment changes from dynamic to static)

services and each service will take at most 10 units of time to execute, the environment in which service providers are available longer than 50 units of time can be considered as a static environment. With an invocation request, if there exists a way of choosing providers for the composite service, we consider it as a successful composition.

In Figure 2, in extremely dynamic environment, the online

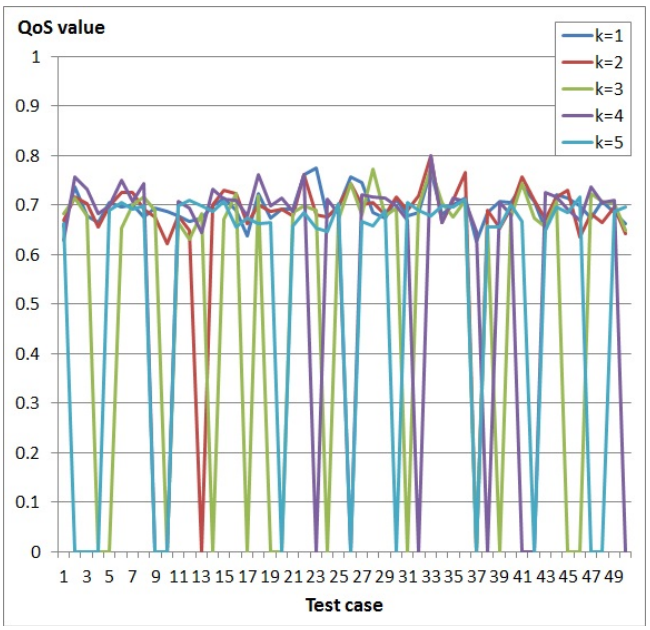

Figure 4. The QoS value of the composite service in specific environment

approach achieved the highest percentage of successful composition; the offline approach secured the lowest percentage of successful composition; the neighbor approach received the values between those two. When the dynamism of the environment increases, the percentage of successful composition of different k values decreases significantly. In static environment, all values of k achieved the same highest percentage of successful composition. Moreover, this figure also shows that traditional approach (set up a solution plan before the execution of the composite service) is not suitable in mobile environments because the percentage of having a composition is very low.

In Figure 3, the general trend is that as we increase the dynamism of the environment, the best average QoS value decreases (because the percentage of successful composition decreases). However, the value k that yields that best average QoS value changes as well. When the dynamism of the environment is low (i.e. in static environments), then the offline approach $(k = n)$ gives the best outcome, whereas when the dynamism of the environment is high, then the online approach $(k = 1)$ is superior.

Figure 4 shows that these two values of k do not always yield the best QoS values for an environment; in particular, one with middle level of the dynamism of the environment.

In summary, our evaluation has shown the trade-off between global optimization and local optimization in selecting providers for a composite service. The output of k-neighbor algorithm changes when the environment changes. In one specific environment, using a different value of k yields a different result. As a result, to achieve the optimal service composition, it is important to adapt to the dynamism of the environment. For example, in Figure 3, there is an environment in which the k-neighbor approach with $k = 4$ achieved the highest average QoS value. How to choose the right value of k to achieve the greatest benefit is related to many parameters such as the number of available providers, the number of atomic services in the invocation request, the average execution duration of each atomic service.

V. Related work

Service composition has been the subject of a lot of research but most studies focused on wired-environments, where the service providers are static and well-known. For example, to find exactly the optimal execution plan, Zeng *et al.* [12] use Linear Integer Programming, Gao *et al.* [4] use Dynamic Programming for linear objective function while Aiello *et al.* [1] apply the Backward Breadth First algorithm and Wan *et al.* [9] adopt Divide-and-Conquer based selection in the case of non-linear objective function. Although our model is similar to the model proposed by Zeng *et al.* [12], our model targets a combination of QoS model and mobility. As a result, we have modified the definition of the availability sense it is treated as a dynamic criterion in our model. We also propose an approach that handles this dynamic criterion.

In the field of wireless ad hoc environments, McDonald [6] predicted the path availability as a function of time while Su, Lee, and Gerla [8] predicted the future location based on the speed, the direction, the radio propagation and time. However, none of those are about service composition. In 2011, Wang [10] proposed a service composition model where predictions were made of the time that mobile devices would be in the current environment. Recently, Prete and Capra [3] proposed a network-aware QoS model for the composite service that used historical co-location patterns to predict the time a provider would remain collocated with a client. However, both Wang's model and model proposed by Prete *et al.* do not consider other traditional QoS criteria such as execution cost, reputation or successful execution rate. Moreover, Wang's model [10] assumes that the execution duration of each service is the same on every service provider. This is not reasonable in real life because each service provider has a different hardware configuration; as a result, the execution durations will differ. In summary, our model is an improvement to the model proposed by Wang [10] and different from traditional QoS models since it combines a QoS model with mobility.

VI. Conclusion

With the progress in modern smart devices, mobile devices can become service providers. In mobile environments, non-functional aspects uniquely belonging to mobile devices such as the mobility should be considered in addition to the traditional QoS criteria. In this study, we proposed a novel model suitable for QoS-aware service composition in

mobile environments. We have proposed the concept of a dynamic criterion as well as an effective approach in service composition with a dynamic criterion. More specifically, our contributions are (1) a formal model for QoS-aware service composition with the concept of a dynamic criterion, availability, and (2) the k-neighbor approach which can handle dynamic environments. Specially, we discussed three cases: the online approach, the offline approach and the neighbor approach. The offline approach can be regarded as the global optimization of service compositions while the online approach can be regarded as the local optimization of service composition and the neighbor approach can be seen as intermediate between the two other approaches. Evaluation showed the trade-off between global optimization and local optimization in different environments and the necessity of proposed local global-optimization approach to adapt well to the dynamism of the environment. In this work, we have focused on the strictly sequential service composition but it is also useful to consider more general cases. Besides, it is also necessary to explore the effect of other parameters on the decision of how to choose the value of k in k-neighbor approach. Further consideration will be given to those tasks in future.

ACKNOWLEDGMENT

This research was supported by a Grant-in-Aid for Scientific Research(S) (24220002, 2012-2016) from Japan Society for the Promotion of Science (JSPS).

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions to improve the quality of the paper. They also grateful to Dr. Yohei Murakami for his help.

REFERENCES

[1] M. Aiello, E. El Khoury, A Lazovik, and P. Ratelband. Optimal qos-aware web service composition. In *Commerce and Enterprise Computing, 2009. CEC '09. IEEE Conference on*, pages 491–494, July 2009.

[2] D. Chalmers and M. Sloman. A survey of quality of service in mobile computing environments. *Communications Surveys Tutorials, IEEE*, 2(2):2–10, Second 1999.

[3] Lucia Del Prete and Licia Capra. Mosca: service composition in mobile environments. In *Proceedings of the ACM/IFIP/USENIX Middleware'08 Conference Companion*, pages 87–89. ACM, 2008.

[4] Yan Gao, Jun Na, Bin Zhang, Lei Yang, and Qiang Gong. Optimal web services selection using dynamic programming. In *Computers and Communications, 2006. ISCC '06. Proceedings. 11th IEEE Symposium on*, pages 365–370, June 2006.

[5] Jingwen Jin and K. Nahrstedt. Source-based qos service routing in distributed service networks. In *Communications, 2004 IEEE International Conference on*, volume 4, pages 2036–2041 Vol.4, June 2004.

[6] AB. McDonald and T.F. Znati. A mobility-based framework for adaptive clustering in wireless ad hoc networks. *Selected Areas in Communications, IEEE Journal on*, 17(8):1466–1487, Aug 1999.

[7] Bhaskaran Raman and R.H. Katz. Load balancing and stability issues in algorithms for service composition. In *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, volume 2, pages 1477–1487 vol.2, March 2003.

[8] William Su, Sung-Ju Lee, and Mario Gerla. Mobility prediction in wireless networks. In *MILCOM 2000. 21st Century Military Communications Conference Proceedings*, volume 1, pages 491–495. IEEE, 2000.

[9] Changlin Wan, C. Ullrich, Limin Chen, Rui Huang, Jiewen Luo, and Zhongzhi Shi. On solving qos-aware service selection problem with service composition. In *Grid and Cooperative Computing, 2008. GCC '08. Seventh International Conference on*, pages 467–474, Oct 2008.

[10] Jianping Wang. Exploiting mobility prediction for dependable service composition in wireless mobile ad hoc networks. *Services Computing, IEEE Transactions on*, 4(1):44–55, Jan 2011.

[11] Shuichi Yamaoka, Tao Sun, Morihiko Tamai, Keiichi Yasumoto, Naoki Shibatay, and Minoru Ito. Resource-aware service composition for video multicast to heterogeneous mobile users. In *Proceedings of the first ACM international workshop on Multimedia service composition*, pages 37–46. ACM, 2005.

[12] Liangzhao Zeng, B. Benatallah, A H H Ngu, M. Dumas, J. Kalagnanam, and H. Chang. Qos-aware middleware for web services composition. *Software Engineering, IEEE Transactions on*, 30(5):311–327, May 2004.