A Multiple Criteria Service Composition Selection Algorithm Supporting Time-sensitive Rules

Lei Xu †, Lei Shi †, Runxin Wang † and Brendan Jennings †
* Department of Computer Science, Umeå University, Sweden, E-mail: lxu@cs.umu.se
† Telecommunications Software & Systems Group, Waterford Institute of Technology, Ireland
E-mail: {lxu, lshi, rwang, bjennings}@tssg.org

Abstract—Constructing composite services by using of services offered by third parties is an attractive and inexpensive way for service brokers and aggregators to enhance differentiation from their competitors. When multiple services provide the same or similar functionalities, selecting those that satisfy users’ non-functional requirements is crucial. In many cases, non-functional properties of services are heavily dependent on the activity of the network delivering those services whilst the network activity follows certain time-sensitive rules. We present a service selection algorithm that takes into account time-sensitive variations of non-functional propensities of services to identify a service combination offering the highest quality within a specified time interval.

I. INTRODUCTION

With the proliferation of services and the increasing diversity of service providers, indirect service provision grows quickly in which service brokers and service aggregators create novel compositions by combining services offered by third parties and provide them to customers. In such a business environment, lots of services may offer the desired functionalities but with different non-functional attributes. Therefore, there is a clear incentive for those service brokers and service aggregators to identify service combinations that satisfy service users from those offering the same or similar functionalities. Non-functional aspects of services, such as Quality of Service (QoS), are key factors for such selections.

When we only consider single services or a single non-functional criterion, selecting the service or service combination that meets users’ requirements is relatively simple. When a given task needs to be finished by a combination of multiple services—a composite service, the situation is much more complicated. The service composer will need to have knowledge on how multiple non-functional selection criteria affect each other, and whether they are equally important or not. Furthermore, in many cases, non-functional aspects of services are heavily dependent on the activity of the network that delivers those services. It has been recognised for more than thirty years that network activity follows time sensitive rules [1]. Thus, to select a service combination that satisfies users’ non-functional requirements, the service composer will need to have knowledge of the times at which customers are most likely to invoke the composite service.

Selecting the optimal combination of services from a service population in which multiple services provide the same functionality but differ in their non-functional properties has received significant attention. Much effect has been made to investigate different approaches and algorithms that aggregate multiple selection criteria. However, these solutions do not consider how those non-functional attributes vary with the times at which services are invoked. In this paper, we address the problem by presenting a novel service selection that identifies a composite service with highest quality for access within a defined time interval.

This paper extends our previous work [2] by applying Dynamic Programming to significantly improve the time complexity. Moreover, our previous service selection algorithm dealt only with a single non-functional aspect: service cost. In this paper we generalise the approach to deal with multiple non-functional aspects, where numerical values are used to indicate relative performance of individual services in terms of individual aspects, such as cost, reliability and availability.

The paper is structured as follows. In §II we briefly review the literature on service selection for composite services. In §III-A we define our service selection algorithm which considers multiple non-functional aspects in the presence of time-sensitive problems. §III-B specifies our service selection algorithm, illustrates its operation via an example use case, and analyses its computational complexity. Finally, §IV summarises the contribution of the paper and outlines avenues for future work on this and related topics.

II. RELATED WORK

The increasing availability of services that offer the same or similar functionalities with different non-functional attributes increases the need for sophisticated service selection process to meet user requirements. Service selection can be based on users’ evaluation and preference, such as [3] and [4]. Service selection can also be achieved based on improving of service lookup and discovery standards, such as Universal Description Discovery and Integration (UDDI). The examples of this category are [5] and [6]. Furthermore, some researchers have investigated the usage of Semantic Web technologies to find the matching services, such as [7] and [8]. The mainstream in exiting work of service selection is based on security, Quality-of-Service (QoS), and cost. Much effect has been made in investigating QoS-based optimization approaches. Examples for QoS based service selections include [9], [10], and [11].

Service selection approaches discussed above provide solutions to identify suitable composite services for service com-
positions. However, none of them consider how non-functional aspects vary with times at which services are invoked and thus they do not meet the requirement of our work. We need a piece of work deployed in our charging framework as depicted in [12] to provide suggestions for the service composer – the component that constructs composite services and offers them to customers. These suggestions identify composite services that deliver the highest quality within the time interval of interest based on the non-functional selection criteria requested by the service composer.

### III. SERVICE SELECTION

To select services comprising a composite service whose non-functional attributes vary in different time intervals, a service composer will need to have knowledge of the times at which customers are most likely to invoke that composite service. We assume non-functional attributes of services are depicted by time-sensitive rules associated with them. We use a score system to describe the nature and requirements of those attributes that will be considered in our selection. Those scores of non-functional issues need to be normalised that is to allow for a uniform measurement of service quality independent of units and put all the values in a range from 0 to 1. Non-functional selection criteria could be classified as two categories: those the higher the scores are, the higher the quality delivered and those the higher the scores are, the lower the quality delivered. They follow different ways to normalise, as depicted in section III-A below.

After normalisation, we could calculate quality scores of services comprising that composite service. This can be done by combine time sensitive rules of those non-functional issues to identify sub-intervals within the interval of interest during which scores of multiple criteria will not change. We assume a service with the lower normalised quality score delivers better quality. We assign weights that are decided by the service composer to scores of different non-functional attributes. To merge multiple selection criteria, we sum the weighted scores to get the quality score of each sub-interval. If assuming there is an equal probability of service invocation by users across the time interval of interest, we can calculate the mean expected quality score of selection criteria using a simple weighted computation very expensive as the number of service types combinations and then performs a comprehensive pairwise quality score of selection criteria using a simple weighted time interval of interest, we can calculate the mean expected to get the quality score of each sub-interval. If assuming there are an equal probability of service invocation by users across the time interval of interest, we can calculate the mean expected quality score of selection criteria using a simple weighted.

A better solution is use of Dynamic Programming which addresses complex problems by breaking them down into simpler steps. We could split the pairwise comparison of each service composition’s non-function criteria score into simpler sub-problems in a recursive manner. In each of those sub-problems, we calculate that quality score of service pairs which consists of two services of different service types and then find out the optimal one with the lowest quality score (the highest quality). Once we find out the optimal service pair, we can store the solution and service types of the two services comprising the solution. Further analysis will not performed on any service pair which is comprised by services of the two service types. It is a recursive process and it will continue until all optimal solutions have been found out. This approach takes much less time than the straightforward way because once we get a optimal pair, service types and their services that need to be further checked are reduced and it is not required to compare each service composition with all others time and time again. Meanwhile, it is computationally cheaper to calculate the quality score of a service pair than a full service combination if it consists of more than two services.

#### A. Problem Statement

We formally state the service selection problem as follows. Let $CS$ denote the target composed service, which consists of $N$ service types. Let $y_n \in \{y_1, y_2, \ldots, y_N\}$ denote an arbitrary service type required to construct $CS$. Let $Y$ denote set of service types which will be used to construct $CS$. Let $\bar{Y}_n$ denote all service types included in $Y$ except for $y_n$. Let $|Y|$ denote the number of service types included in $Y$. Let $s_i \in \{s_1(y_n), s_2(y_n), \ldots, s_k(y_n)\}$ denote one of the $X(y_n)$ available services of type $y_n$. Let $PS(s_i(y_n), s_j(y_n))$ denote a pair of services. Let $t_e$ denote the start of the interval of interest indicated by the service composer whilst $t_e$ denote the end of that interval. Let $t_k$, where $t_k < t_s \leq t_e$, denote the end of one of the sub-intervals of $(t_s, t_e)$ – the sub-interval during which the scores of multiple criteria of a given service pair do not vary. Let $K(s_i(y_n), s_j(y_n))$ denote the number of such intervals for the service pair $PS(s_i(y_n), s_j(y_n))$. Let $k$, where $1 \leq k \leq K(s_i(y_n), s_j(y_n))$, denote an arbitrary one of the above sub-intervals and let $t_0 = t_s$, $t_k = t_e$.

Let $CR_k \in \{CR_1, CR_2, \ldots, CR_K\}$ denote one of the non-functional service selection criteria, where $CR$ is total number of criteria. To evaluate each service, we normalise its score of every selection criterion. Let $\hat{CR}_k(s_i(y_n))$ and $\hat{CR}_k(s_i(y_n))$ denote score of criterion $CR_k$ of the service $s_i(y_n)$ before and after the normalization respectively. To normalise $CR_k(s_i(y_n))$, we need to find out the maximum value and minimum value of $CR_k$ of all services of $Y$, denoted as $\hat{CR}_k(Y)_{\text{max}}$ and $\hat{CR}_k(Y)_{\text{min}}$ respectively. If $CR_k$ is a criterion of which the higher the value is, the higher the quality delivers, it can be normalised as:

$$\hat{v}_k(s_i(y_n)) = \begin{cases} \left( \frac{\hat{CR}_k(Y)_{\text{max}} - \hat{CR}_k(s_i(y_n))}{\hat{CR}_k(Y)_{\text{max}} - \hat{CR}_k(Y)_{\text{min}}} \right) & \hat{CR}_k(Y)_{\text{min}} \neq \hat{CR}_k(Y)_{\text{max}} \\ 1 & \hat{CR}_k(Y)_{\text{min}} = \hat{CR}_k(Y)_{\text{max}} \end{cases}$$

(1)

Otherwise $CR_k$ can be normalised as:

$$\hat{v}_k(s_i(y_n)) = \begin{cases} \left( \frac{\hat{CR}_k(s_i(y_n)) - \hat{CR}_k(Y)_{\text{min}}}{\hat{CR}_k(Y)_{\text{max}} - \hat{CR}_k(Y)_{\text{min}}} \right) & \hat{CR}_k(Y)_{\text{min}} \neq \hat{CR}_k(Y)_{\text{max}} \\ 1 & \hat{CR}_k(Y)_{\text{min}} = \hat{CR}_k(Y)_{\text{max}} \end{cases}$$

(2)

Let $\hat{v}(s_i(y_n), s_j(y_n))$ denote the sum of weighted scores of
we apply equation 3 to calculate $\hat{V}(s_i(y_x), s_j(y_y))$ when invoked within the sub-interval $k$ as:

$$\hat{V}(s_i(y_x), s_j(y_y))_k = \sum_{\alpha=1}^{CR} w_{cr} v_{cr}$$

where $CR$ is the number of criteria for each service, $w_k \in [0, 1]$ is the weight assigned by the service composer to the criterion $cr$ such that $\sum_{\alpha=1}^{CR} w_{cr} = 1$ and $v_{cr}$ is the sum of the scores (after normalization) of criterion $cr$ of the both services $s_i(y_x)$ and $s_j(y_y)$ within the sub-interval, where $v_{cr} = \hat{v}_k(s_i(y_x)) + \hat{v}_k(s_j(y_y))$. Let $CS^*$ denote the combination of services with the lowest sum of the expected mean score of different selection criteria. We assume the probability of invocation of $PS(s_i(y_x), s_j(y_y))$ during interval $(t_x, t_y)$ is equal. Based on above, we can calculate the mean score (quality score) of multiple selection criteria of $PS(s_i(y_x), s_j(y_y))$, denote $\hat{V}(s_i(y_x), s_j(y_y))$ as:

$$\hat{V}(s_i(y_x), s_j(y_y)) = \sum_{k=1}^{K(s_i(y_x), s_j(y_y))} \frac{t_k-t_{k-1}}{t_e-t_s} \cdot \hat{V}(s_i(y_x), s_j(y_y))_k$$

Thus, the service pair with the lowest quality score of selection criteria which consist of service type $y_n$ and a service of other service types $\overline{y}_n$ can be stated as:

$$min \hat{V}(X(y_n), X'(\overline{y}_n)) = \min_{PS(X(y_n), X'(\overline{y}_n))} \hat{V}(X(y_n), X'(\overline{y}_n))$$

B. Selection Algorithm

To calculate the quality score of a service pair, the algorithm analyses the time-sensitive rules of each selection criterion of two services to identify sub-intervals within the interval of interest within which the score of each of those criteria will not change. We assume each rule of selection criteria of services contains the necessary data including: the start time of the sub-interval, $t_{x-1}$, the end time of the sub-interval, $t_x$, and the score of each selection criterion for that sub-interval. For each sub-interval of a given service pair $PS(s_i(y_x), s_j(y_y))$, we apply equation 3 to calculate $\hat{V}(s_i(y_x), s_j(y_y))$ the sum of weighted values of selection criteria. Then we apply equation 4 to calculate $\hat{V}(s_i(y_x), s_j(y_y))$ the mean expected value of selection criteria – quality score of $PS(s_i(y_x), s_j(y_y))$.

This selection algorithm analyses services of every service types. If there is only one service type included in $Y$, the algorithm finds out the service of that type with the lowest quality score across the interval of interest and then inserts it to the $CS^*$. This is outline from line 4 to 5 of the algorithm 1. Otherwise, the algorithm checks services of different service types on a pairwise basis. Based on quality score of each service pair, the algorithm tries to find out a service pair $PS(s_a(y_b), s_c(y_d))$ of which $\hat{V}(s_a(y_b), s_c(y_d))$, $\hat{V}(s_c(y_d), s_a(y_b))$ are equal and they are the minimum value of quality scores of those service pairs comprised of a service of $y_b$ and another service of $\overline{y}_b$ and those service pairs comprised of a service of $y_d$ and another service of $\overline{y}_d$ respectively. Afterward, both services $s_a(y_b)$ and $s_c(y_d)$ are added to $CS^*$. Their service types $y_b$ and $y_d$ are removed from $Y$ whose services will not be analysed in further operations. This is outlined from line 7 to 12 of algorithm 1. An example is presented in figure 1 which illustrate how the algorithm can find out the composite service comprising three different services with the lowest quality score. The cost and availability of services are taken account in the selection.

1) Complexity Analysis: Let $rs$ represent the mean number of rules of each selection criterion contained in services of each service type. Let $c$ represent the number of selection criteria. As before, let $N$ represent the number of service types required to construct the target composed service $CS$. Let $M$ represent the mean number of services available for the $N$ required service types.

To generate sets of sub-intervals for a given service pair, the selection algorithm compares rules of two services, given $2rs \cdot c - 1$ comparisons. To find out a service pair with the lowest quality score, we examine $M^2(N-1)$ service pairs. To further evaluate this service pair, we compare it with $\frac{M(N-1)^2}{2}$ other service pairs under the worst circumstances. Therefore, if $N$ is an odd number, the computational complexity of the algorithm is

$$O\left(\frac{(N-1)(2rs \cdot c - 1)}{2} \left( M^2(N-1) + \frac{M(N-1)^2}{2} \right) \right)$$

If $N$ is an even number, the complexity of the selection algorithm is

$$O\left(\frac{N(2rs \cdot c - 1)}{2} \left( M^2(N-1) + \frac{M(N-1)^2}{2} \right) \right)$$

If we only consider the leading term, since the lower-order terms and constant factors are less significant than the rate of growth in determining computational efficiency caused by the leading term, the computational complexity of the algorithm is $O(\frac{(2rs \cdot c - 1)N^3}{2})$ when $N \gg M$ or $O(\frac{(2rs \cdot c - 1)M^2}{2})$ when $M \gg N$. We would expect that in most cases those non-

---

**Algorithm 1 Service Selection Algorithm.**

1. begin
2. Let $Y = \{y_1, y_2, ..., y_N\}$
3. while $Y = \{y_1, y_2, ..., y_N\}$ do
4. if $|Y| = 1$ then
5. Add $s_k(y_1)$ to $CS^*$, where $y_1 \in Y, V(s_k(y_1))$ is min
6. else
7. if $\hat{V}(s_i(y_m), s_j(y_n)) = \hat{V}(s_i(y_n), s_j(y_m))$ then
8. Set $s_a(y_b) = s_j(y_n), s_b(y_d) = s_i(y_m)$
9. while $\hat{V}(s_a(y_b), s_c(y_d)) > \hat{V}(s_i(y_m), s_j(y_n))$ do
10. Set $s_c(y_d) = s_a(y_b), s_a(y_b) = X(y_b)$
11. Add $s_a(y_b), s_c(y_d)$ to $CS^*$
12. Remove $y_b, y_d$ from $Y$
13. end
functional aspects considered in the selection will be less than ten in most cases. Meanwhile, the number of services and service types making up a composed service will be no more than ten and one hundred respectively at the very most. Thus, we believe the computational complexity of the algorithm is acceptable.

IV. SUMMARY AND FUTURE WORK

This paper proposed a service selection algorithm which serves to identify a service combination that is expected to deliver the highest quality when invoked within a specified time interval. It uses a score system to describe the nature and requirements of different non-functional properties that are considered in selections. It normalises scores of those non-functional selection criteria and combining time sensitive rules to identify sub-intervals within the interval of interest within which the normalised scores of rules of a service pair will not change. Weights are assigned to the score of each selection criterion of a service within each sub-interval. We sum weighted scores of selection criteria of a service pair during a sub-interval to aggregate multiple non-functional properties. The mean expected quality score of that service pair is then calculated using a simple weighted average, and optimal service pairs with the lowest quality score are selected and recorded in a recursive manner until the service combination delivering the highest quality is identified.

We have presented our service selection algorithm theoretically in the paper. Future work will initially concentrate on experimental analysis of the scalability of the algorithm. Meanwhile, we will improve the algorithm to make it more close to practice. Currently, we assume that there is an equal probability of service invocation by users across the interval of interest. In practices, the service composer will have more in-depth knowledge of behaviour of its target market. We will investigate more realistic service invocation model, such as the model of network activity patterns presented by [1]. Additionally, in many cases, there may be more than one service offering the same or similar functionalities in a composite service. These services may be invoked in parallel or sequentially until one of them produce the desired outcome. In that scenario, more than one service combination would be required to be identified for our charging framework. We will investigate how to select multiple suitable combinations in that circumstance.

REFERENCES