2012 IEEE Ninth International Conference on Services Computing

A Quantitative and Qualitative Approach for NFP-aware Web Service Composition

Hongbing Wang*, Peisheng Ma*, Xuan Zhou†
*School of Computer Science and Engineering, Southeast University, Nanjing, China
{hbw,psm}@seu.edu.cn
†DEKE Lab, Renmin University of China, Beijing, China
xuan.zhou.mail@gmail.com

***Abstract*—Web service composition is a standard approach to create value-added services from existing ones. As the Web services on the Internet grows, there are more and more services providing identical functionalities while differing in their non-functional properties (NFPs). However, most of the existing techniques for NFP-aware service composition consider either only quantitative NFPs or only qualitative NFPs. In this paper, we present a service composition model considering both quantitative and qualitative NFPs. We propose two algorithms for conducting service composition. One combines global optimization with local selection into one mechanism. The other is a genetic algorithm based solution. We have conducted extensive experiments to evaluate the effectiveness of our proposals.**

***Keywords*-NFP-aware; service composition; NFPs; quantitative and qualitative; TCP-nets;**

I. Introduction

As the number of Web services increases, SOA (service-oriented architecture) is becoming a standard way for software development on the Web. Using the existing SOA technologies, such as SOAP, WSDL and UDDI, service providers can deploy and publish their Web services on the Internet, and service consumers can identify or invoke the services. However, with existing technologies, it still relies on human efforts to select appropriate services from the service repositories to meet a user's requirements. With the rapid increase of Web services, it is natural that many services provide similar or identical functionalities, while differing in their NFPs (non-functional properties) or QoS (quality of service), such as response time, price, availability, throughput, platform, security, SLA level and so on. $Service\ Class$ is usually used to denote the collection of services with the same functionality. We need a system that can automatically select the services from a service class that can satisfy a user's requirements on NFPs.

Service providers tend to make their services generic in terms of functionality, so as to meet the requirements of more potential customers. However, the requirements of customers are becoming increasingly complex. To meet a user's demands, a composite service usually needs to be created by combining the functionalities of multiple services. The component services can be atomic services or composite services on their own. Service composition can be conducted using a workflow based language (e.g. BPEL[1]). There have been extensive research efforts focusing on how to perform service composition easily and efficiently.

Traditional NFP-aware web service composition approaches consider only the quantitative non-functional properties, as in [2], [3], [4], [5]. Most quantitative approaches are computationally efficient, while they have several limitations. First, some NFPs cannot be represented quantitatively, such as the locations of services (e.g. $China$ and USA). Second, user preferences over NFPs can be conditional. For example, a data storage service can have two properties, i.e., $Platform$ and $Location$. A user in $China$ may prefer the service located in USA, given that the $Platform$ is a $database$. He may prefer the service located in $China$, given that the $platform$ is a $FileSystem$. As we can see, pure quantitative solutions cannot handle all the types of NFPs. To bridge the gaps, some [6] have proposed to model users' preferences over NFPs in a qualitative way, so that we can select optimal web services qualitatively. However, the prior proposals have limitations too. First, there could be too many result services, when only the qualitative solution is utilized. Second, qualitative and quantitative NFPs may both be present, and need to be handled simultaneously. In short, using a qualitative model or a quantitative model alone cannot capture users' requirements on NFS completely. One should expect different types of presentation about users' requirements, even within the same decision-making process. In this paper, we propose a new preference model considering both quantitative and qualitative non-functional properties, and a series of solutions to NFP-aware service composition based on the new model.

The main contributions of this paper are summarized as follows: (1) We propose to incorporate both qualitative and quantitative NFPs into the preference model for NFP-aware service composition. (2) We propose two solutions for NFP-aware service composition based on the new preference model. (3) We verify the effectiveness of our approaches theoretically and experimentally.

The remainder of this paper is organized as follows. Section II gives an illustrative example to demonstrate the qualitative and quantitative NFPs and their relationship. Section III gives some background on CP-net and TCP-net. Section IV proposes a new preference model and defines the qualitative pattern and quantitative value of composition

978-0-7695-4753-4/12 $26.00 © 2012 IEEE
DOI 10.1109/SCC.2012.16

service. Section V defines the problem. Sections VI and VII presents the solutions and the empirical study respectively. Section VIII reviews some related work. Finally, Section IX provides a conclusion.

II. Problem Illustration

In this section, we illustrate what quantitative property and qualitative property are. Then we establish the relationship between these two types of properties.

As shown in Table I, there are three service classes ($Service\ Class\ 1$, $Service\ Class\ 2$, $Service\ Class\ 3$), each containing nine Web services. The services of the same class provide the same function but differ in their NFPs values. Suppose there are 7 NFPs properties, which are $response\ time$, $price$, $reliability$, $availability$, $platform$, $location$ and $provider$.

We reuse the definitions of quantitative properties and qualitative properties in [7]. A NFP is a qualitative property *iff* it refers to some world object instead of a numerical value. On the contrary, a NFP is a quantitative property *iff* its value is a numerical value or can be transformed to a numerical value.

It is not trivial to transform a qualitative property to a quantitative one directly, because we cannot determine which value should be assigned to each qualitative value. In [8], the authors proposed a preference model called UCP-nets, which can assign a utility value for each composition of all qualitative properties. However, when using UCP-nets to express preferences, the users have to provide a lot of information, which may not be feasible in practice [9].

In contrast, it is easier to transform a quantitative property to a qualitative one. For example, the domain of the response time can be divided into several intervals. Then a unique name or a qualitative value can assigned to each interval. For instance, $Fast$ is assigned to ($0s$, $0.5s$], $Medium$ is assigned to ($0.5s$, $1.0s$] and $Slow$ is assigned to ($1.0s$, $+\infty$). Although the approach is straight forward, we should note that the difference between the values of the same interval may be great. For example, while the difference between $0.49999s$ and $0.50001s$ is small, they belong to different intervals. While the difference between $0.50001s$ and $0.99999s$ is relatively greater, they belong to the same interval. Such kind of transformation would decrease the reliability of the preference captured by the original quantitative model.

On one hand, it is not appropriate to transform one type of preferences to the other. On the other hand, there are both quantitative and qualitative properties present in real-world Web services. What to do with NFP-aware web service composition? This is the question this paper tries to address.

III. Introduction to CP-net and TCP-net

Conditional Preference Network (CP-net) was first introduced by Boutilier et al. in [10]. It was designed to represent users' preferences on qualitative attributes, also known as categorical attributes. In order to enhance the expressive power of CP-net, Brafman et al. in [11] proposed TCP-net by introducing relative importance. Fig. 1 shows an illustrative example of CP-net and TCP-net over the qualitative NFPs of the web services in Table. I. More detailed information can be found in [10], [11].

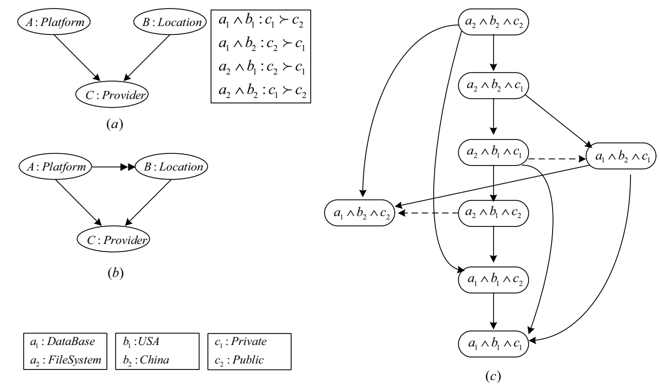

Figure 1. Examples: (a) CP-net (b) TCP-net (c) Induced Preference Graph

A network N is satisfied by some preference ranking $\succ$ iff $\succ$ satisfies each of the conditional preferences expressed in the CPTs of N under the *ceteris paribus* interpretation. The authors of [10] showed that every acyclic CP-net is satisfiable. When there are some cycles in N which are identified as conflicts, they can be removed by using the approach described in [6].

Defnition 1 (Dominance Testing and Ordering Testing)**.** *Given a CP-net N and two instances e_1 and e_2, the dominance testing aims to determine if one of the instances dominate the other, it must reach one of the following three conclusions – (1) $N \models e_1 \succ e_2$ or (2) $N \models e_2 \succ e_1$ or (3) $N \not\models e_1 \succ e_2 \wedge N \not\models e_2 \succ e_1$ (also known as indifference). While the ordering testing aims to reach one of the following conclusions – (1) $N \not\models e_1 \succ e_2$ or (2) $N \not\models e_2 \succ e_1$ or (3) both.*$\square$

In general, the dominance testing for CP-net (and thus TCP-net) is known to be NP-hard according to [10], [11], while the ordering testing can be answered in time linear in the number of variables. The ordering testing is actually good enough for our purpose.

IV. The Model

In this section, a new preference model considering both quantitative and qualitative NFPs is presented. First, we presented the definition of NFP model for component service. Then, a set of rules are defined, according to which the values of composite service can be calculated.

In the NFP-aware service composition architecture, which is proposed in [12], the services in the Service Repository

Table I
AN EXAMPLE OF SERVICES WITH NFPS

Class	ID	ResponseTime	Throughput	Reliability	Availability	Platform	Location	Provider
Service Class 1	$service_{11}$	$1s$	7.1	97%	95%	DB	USA	Private
	$service_{12}$	$2s$	16.0	98%	96%	FS	USA	Public
	$service_{13}$	$1s$	15.3	99%	97%	DB	USA	Public
Service Class 2	$service_{21}$	$4s$	13.5	97%	97%	FS	China	Public
	$service_{22}$	$1s$	10.9	97%	95%	FS	China	Private
	$service_{23}$	$5s$	12.2	98%	97%	DB	China	Public
Service Class 3	$service_{31}$	$1s$	13.3	99%	96%	FS	USA	Private
	$service_{32}$	$3s$	15.2	98%	96%	DB	China	Private
	$service_{33}$	$2s$	14.5	99%	95%	DB	USA	Private

are organized as service classes. The execution processes in the Process Repository are also represented as a series of service classes and their relationships. When the user request arrives, the system will generate an execution plan as an abstract workflow. Only when each of the service classes is instantiated with a concrete service, the abstract workflow can be transformed to an Executable Business Process. The process of transformation is done by a Service Selection Manager. As the functionalities provided by the services of the same class are identical, only the NFPs are used as the selection criteria.

The NFPs are composed of several properties, among which there are some quantitative properties and some qualitative ones. And these properties can be regarded as a property vector, which is formally defined as:

Defnition 2 (NFP MODEL)**.** *The **NFP Vector** of a service is defined as $(V_1, V_2...V_m, V'_1, V'_2...V'_n)$, where m, n are both positive integers, $V_i, i \in [1, m]$ are quantitative variables, $V'_j, j \in [1, n]$ are qualitative variables. And $(v_1, v_2...v_m)$ is called the **Quantitative Value** of the service, $(v'_1, v'_2...v'_n)$ is called the **Qualitative Pattern** of the service, where $v_i, i \in [1, m]$ and $v'_j, j \in [1, n]$ is the value of the corresponding variable according to the service.*$\square$

It is easy to evaluate Quantitative Value and Qualitative Pattern of an atomic service by assigning each NFP value to the corresponding variable in the NFP Vector. However, the question becomes more complex when it comes to a composite service.

On the one hand, the NFP values of a composite service are tightly connected with the structure of the generated execution plan and the type of the NFP variable. The typical structures are *Sequence, Switch, Flow*, and *Loop*. Here, we focus on the sequential composition model, as the others may be reduced or transformed to the sequential model [2], [4]. More detailed information about unfolding cyclic structure can be found in [4].

On the other hand, quantitative and qualitative NFPs use different rules, which are described as below.

A. Quantitative Value of Composite Service

Most of the traditional works about NFP-aware web service composition only consider quantitative properties

Table II
EXAMPLE OF QUANTITATIVE AGGREGATION FUNCTIONS

NFP Vars	Time(T)	Price(P)	Availability(A)	Reliability(R)
Sequential	$\sum_{i=1}^{m} T_i$	$\sum_{i=1}^{m} P_i$	$\prod_{i=1}^{m} A_i$	$\prod_{i=1}^{m} R_i$

and they have adopted the similar aggregation functions to compute the NFP values of composite service [4], [2], [5]. For the quantitative properties in our NFP model, the same aggregation functions are used, and they are described in Table II.

B. Qualitative Pattern of Composite Service

Every service has a qualitative pattern according to the Definition 2. If CP-nets are used to model the preferences on qualitative properties, the qualitative pattern is also known an outcome of the CP-net.

Defnition 3 (Qualitative Pattern of Composite Service)**.** *Given a composite service CS, which is composed of a set of component services. The Qualitative Patterns of these component services compose a set $\{pat_1, pat_2 \ldots pat_n\}$, which is sorted as $\{pat'_1, pat'_2 \ldots pat'_n\}$ based on the results of the Ordering Test (i.e., $N \nvDash pat'_{i+1} \succ pat'_i$ for all is). Then, pat'_n is called the Qualitative Pattern of CS. In other words, the Qualitative Pattern of CS is the worst one of its components' Qualitative Patterns.*$\square$

As NFPs are qualitative, they can not be calculated by arithmetic methods. Fortunately, the Ordering Test of TCP-net can be used to order the Qualitative Patterns, although the result is not necessarily unique. The worst pattern represent the lower bound of the quality of the entire composition. For example, suppose a composite service consists of two component services, whose Qualitative Patterns are $pat_{11}(DB, USA, Private)$, $pat_{21}(FS, China, Public)$ respectively. pat_{11} is preferred by the user, according to the induced preference graph in Fig.1. However, the resulting composite service may not be able to provide the the same quality as pat_{11}, since pat_{21} is also one of its components. It is safer to use pat_{21} to represent its qualitative NFPs.

V. PROBLEM STATEMENT

The traditional NFP-aware service composition is a constraint optimization problem which aims to finding the composition that maximizes the overall utility value while satisfying all the global quantitative NFP constraints [4], [2], [5].

The global quantitative NFP constraints are represented as a set of constraint statements, and the *overall utility value* of each composed result is computed according to users' quantitative preferences, which are expressed as a vector of weights. For each available quantitative NFP attribute, there is a weight in the vector, and the sum of all the weights should be one. Similarly, users may also have qualitative constraints and preferences. The qualitative preferences are modeled as a CP-net or TCP-net, and the qualitative constraints are expressed as a set of qualitative rules. For example, a user may require that the *Location* should be in *USA*. Another user may require that the *Location* should be in *China*.

When both quantitative and qualitative NFPs are considered, the best composition is that maximizes the overall utility value and has the best Qualitative Pattern while satisfying all the global quantitative and qualitative NFP constraints. Formally:

Defnition 4 (Best Composite Service)**.** *Given an abstract execution plan* $CS_{abstract} = \{WS_1, WS_2 \dots WS_n\}$*. After assigned a concrete service for each service class,* $CS_{abstract}$ *generate a composite service* $CS = \{ws_1, ws_2 \dots ws_n\}$*.* CS *is called the best composite service iff (1)* CS *implement user's functional requirements; (2)* CS *satisfy all the quantitative constraints imposed by the user; (3) each component service of* CS *satisfy all the qualitative constraints of the user; (4) there is no other instance of* $CS_{abstract}$ *that satisfy (1), (2), (3) and dominate* CS *in both overall utility value and Qualitative Pattern.*□

If the overall utility value and the Qualitative Pattern are denoted as Dim_1 and Dim_2, the relationships between the possible compositions may be incomparable which are summarized as two cases. Case 1: because no one can dominate the other in both Dim_1 and Dim_2. Case 2: because they are incomparable in Dim_2 alone. In case 2, they are incomparable regardless of the relationship in Dim_1. Fortunately, TCP-net instead of CP-net can be used to reduce the probability that this case occurs.

The composed result can fall in three cases:

1) If there is no composite service that can satisfy the constraints provided by the user, then an exception will be raised and the system will propose the user to relax these constraints.
2) If only one result is the best composite service, then the composite service is returned to the end user directly.
3) If there are more than one optimal composite services according to the dominance relationship, then the system select a result randomly for the user.

VI. Algorithms for NFP-aware Service Composition

To generate the best composite service defined above, two algorithms are presented in this section. The first algorithm is extended from the [13], combining global optimization with local selection, The second algorithm is based on a Genetic Algorithm.

A. Combining Global Optimization With Local Selection

In this approach, the NFP-aware service composition problem is performed by dividing it into two sub-problems, each of which can be solved efficiently.

1) Decomposition of Global NFP Constraints: The first sub-problem is: the system decomposes all the global quantitative and qualitative NFP constraints into local constraints. For qualitative constraints, because global constraints are already local constraints (each component service of a composite service should satisfy all the global qualitative constraints), they can be used for local selection in each service class directly. For quantitative constraints, the method proposed in [13] can be used. At first, the domain of each NFP attribute is divided as a set of discrete *quality levels*, which will be used as local quantitative constraints in the local selection step. Then, mixed integer program (MIP) techniques are used to find the best decomposition, which mapping global constraints into a set of quality levels. Note that, due to the fact that the number of variables in such a MIP model is much smaller than that in [4] (which model the NFP-aware service composition as a MIP problem), the MIP model in our scenario is much simpler in computation.

2) Local Selection: The second sub-problem is: to select a concrete component service for each service class involved in the composite service according to the local qualitative and quantitative constraints. Upon the receipt of the local constraints and the users' preferences, some candidates that cannot satisfy the qualitative constraints are eliminated. Then, local quantitative constraints are used to eliminate the candidates that violate these constraints. After that, a list of qualified services is identified, and their utility values over quantitative NFPs and Qualitative Patterns are computed. In order to return the *best* candidate, the services should be sorted according to our selection criteria and the first one will be returned. They can be sorted according to their utility values first, and if there are two or more candidates have the same utility value they are sorted according to their Qualitative Patterns. Alternatively, Qualitative Patterns are used first and then if there are some services have the same Qualitative Pattern they are sorted according to their utility values. This is similar to the approach in [7].

3) Limitations: There are some limitations for the above algorithm, regardless of either utility values or Qualitative Patterns are used first. As illustrated in Fig. 2, the *overall utility value* and the *Qualitative Pattern* are denoted as two dimensions respectively. The skyline set of the candidates

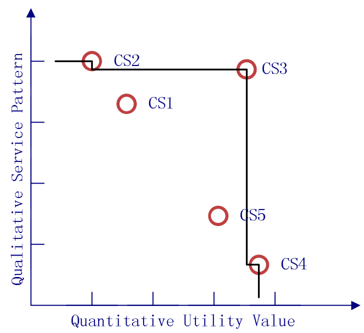

Figure 2. Bad Compromise Example

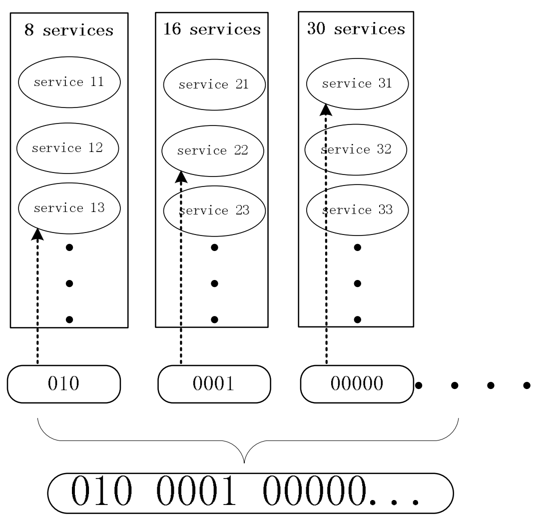

Figure 3. Genome of composite service

consist of CS_2, CS_3 and CS_4. If they are sorted according to the utility value first, CS_4 is the best. If the Qualitative Pattern is used first, CS_2 is the best. However, both CS_4 and CS_2 are bad tradeoffs, because each has the most preferred value in one dimension and the least preferred value in another dimension. Whereas, users usually prefer services with a good compromise (has moderate values in both dimensions) [14], such as CS_3 in this example.

B. Genetic Algorithm

In this section, a Genetic Algorithm based solution which considers both quantitative and qualitative NFPs simultaneously is presented to generate the results with a good compromise. This algorithm is executed after qualitative constraints are applied .

• *Genome Encoding:* Each individual composite service in the population should be encoded, in order to enable the genetic algorithm. Although many representations have been developed for genetic algorithm, the binary case is both the simplest and the most general one [15], [16]. As shown in Fig. 3, there is a genome (represented as a bit string) for each composite service. And the genome is consisted of a set of substrings, each of which is called a chromosome and corresponds to a service class. The decimal value of each chromosome binary string represents the index (start from 0) to the array of concrete services of the service class. As the size of each class may differ, the length of each chromosome may also be different. For example, if there is N candidate services in a class, the number of bits should be n such that $2^n \geq N$ and $2^{n-1} < N$, where $n \geq 1$. When $2^n > N$, the decimal value of arbitrary n bits may exceeds the range $[0, N-1]$. Therefore, the population initialization and the application of crossover and mutation operators should guarantee that the resulted chromosome is meaningful.

• *Fitness Function:* The fitness function defines the criterion for evaluating each composite service in the population and the criterion for probabilistically selecting them for inclusion in the next generation population or to apply crossover operator. To generate the best composite service, the fitness function needs to assign better value to that has higher utility value over quantitative NFP preferences and the better Qualitative Pattern. In addition, the fitness function should assign the worse value to that does not satisfy the quantitative constraints. In this paper, it is assumed that the smaller value of fitness function is better than a bigger one.

Note that the temporary composite service in one generation that does satisfy the quantitative constraints should not be discarded directly, because these services may generate better qualified results eventually by crossover or mutation operator and thus decrease the diversification level. The fitness function is defined as below.

$f_1(CS) = p * \frac{1}{U(CS)}$, where $U(CS)$ is the utility value of composite service *CS*. $f_2(CS) = (1-p) * \frac{1}{O(CS)}$, where $O(CS)$ is the reversed index number (from rear to front) of the Qualitative Pattern of *CS* in $\succ$, which is a ranking of the current population that satisfies TCP-net. $f_3(CS) = q{*}V(CS)$, where $V(CS)$ is some violation value of *CS* with respect to quantitative constraints. For example, supposed the constraint set is $\{response\ time < 0.3s, availability > 98\%\}$, and the actual $response\ time$ of *CS* is $1s$ and the $availability$ of *CS* is 90%, then, $V(CS) = (1-0.3)*10 + (98\% - 90\%)*100 = 15$. And finally, the fitness function is defined as:

$$f(CS) = f_1(CS) + f_2(CS) + f_3(CS) \qquad (1)$$

In the function, p provides a method for balancing quantitative and qualitative NFPs, and q is used to weight the contribution of constraint violation to the overall fitness value. The values of p and q are designated by the process designer.

• *Selection:* The probability, according to which each individual of the population is selected for inclusion in the next generation population or applying crossover operator, is determined by the following equation:

$$Pr(CS_i) = \frac{f(CS_i)^{-1}}{\sum_{j=1}^{n}(f(CS_j))^{-1}} \tag{2}$$

By setting appropriate values for p and q, we can make sure that $0 < f_1(CS), f_2(CS) < 1$ and $f_3(CS) >> 1$, and thus the probability of a qualified composite service to be selected is much greater than that of an unqualified one.

- *Crossover Operator:* The crossover operator produces two new offspring from two parent strings by exchanging some bits. For the sake of diversification, uniform crossover operator is used in this paper. However, as discussed above, the bit strings resulted from crossover may exceed the allowable intervals with respect to each service class. When it happens, the highest bit of the corresponding substring is inverted, i.e. changes 1 to 0.
- *Mutation Operator:* The mutation operator produces small random changes to the bit string by choosing a single bit at random, then changing its value. Similarly, the mutation operator should guarantee that the resulted bit string is still meaningful. When the result is incorrect, the mutation operating is recovered and a new bit is selected to be used. If the recovered bit is the highest one of a chromosome, a new bit is selected from the same chromosome except the highest bit at random; otherwise, the highest bit of the same chromosome is selected.
- *Stop Criteria:* The genetic algorithm runs iteratively, so there should be some stop criteria for GAs to determine when to stop. The criteria may involve the max number of runs of the iteration, or the best utility values and best Qualitative Patterns of the current population meet some conditions (for example, remain unchanged for a number of generations). In our solution, it is determined by two parameters N_1, N_2. N_1 is the max times of iterations, if there is no composition that satisfy quantitative constraints after N_1, the system should raise an exception indicating that the user should relax some constraints. If there are some qualified compositions in the current population, the algorithm will run $min\{N_2, N_1 - N_{used}\}$ times more to refine the result.

VII. Experimental Study

In this section, we present an experimental evaluation for the above algorithms. The aim of this evaluation is to compare the time efficiency of these two algorithms and to observe the compromise quality of their results, i.e. whether it produce good or bad compromise. We have conducted two sets of experiments by using real world and synthetically generated datasets respectively. In each set of experiments we vary the number of service classes and the number of candidate services for each service class.

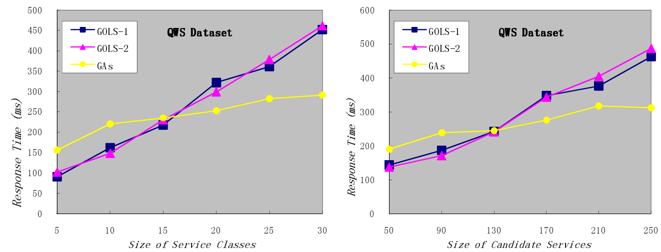

Figure 4. Experimental Result on QWS Dataset

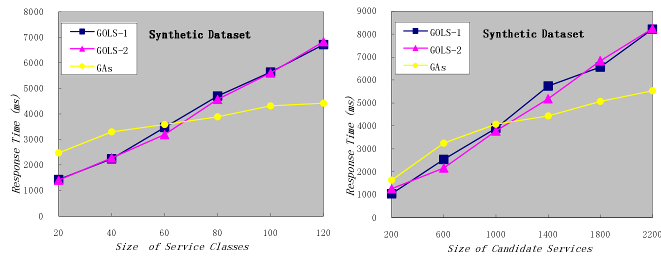

Figure 5. Experimental Result on Synthetic Dataset

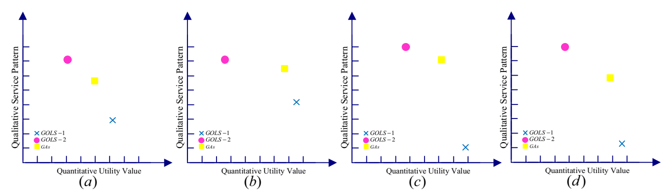

Figure 6. Experimental Results of Compromise Quality: (*a*) QWS, 30 classes, 80 candidates (*b*) QWS, 10 classes, 250 candidates (*c*) Synthetic, 120 classes, 300 candidates (*d*) Synthetic, 20 classes, 2200 candidates

A. Experimental Setup

The NFP-aware service composition system was implemented using Java 1.6, on a personal computer with a CPU of 2.27GHz and a RAM of 2GB, and the operating system is Windows XP.

To simulate user qualitative preferences, we generated random TCP-net. In addition, user quantitative preferences and user constraints are also generated randomly.

Datasets. For real dataset, the publicly available dataset QWS[1] is used. QWS comprises measurements of nine QoS attributes for 2,507 real-world web services, The detailed information about this dataset can be found in [17]. However, as those attributes are all quantitative, We add three qualitative NFPs (i.e. *Platform*, *Location*, *Provider*) to each service and assign each a random value. Each qualitative NFP value is one of the two alternatives, as described in Fig. 1. For synthetic dataset, we generate the experimental datasets at random, where both the quantitative and qualitative NFP values are randomly generated.

[1]http://www.uoguelph.ca/~qmahmoud/qws/index.html/

Algorithms Implementation. We use the open source LpSolve system *lpsolve*[2] version 5.5 in combining global optimization with local selection approach (*GOLS-1*: quantitative first, *GOLS-2*: qualitative first.) Our Genetic Algorithm (*GAs*) was based on an open source Java library[3], and we have incorporated those variations which make true the chromosome is always meaningful into it. We fix N_1 to 100, N_2 to 20 for stop criteria.

Each set of experiment was performed repeatedly for 30 times, and the average performance was calculated.

B. Experiments on QWS Dataset

In the first experiment, we intended to evaluate the effect of the size of service classes. Because there are 2,507 services in QWS and we varied the number of service classes from 5 to 30 and fix the number of candidate services for each class to 80. In the second experiment, we wanted to evaluate the effect of the size of candidate services (the same size is assumed for each service class). We fixed the number of service classes to 10 and varied the size of candidate services from 50 to 250. The experimental result is presented in Fig. 4.

C. Experiments on Synthetic Dataset

In order to evaluate our approaches more extensively, we used a synthetic dataset also. In this set of experiments, we generated 42,000 services. In the first experiment, we varied the number of service classes from 20 to 120 and fixed the number of services for each class to 300. In the second experiment, we fixed the number of service classes to 20 and varied the size of candidate services from 200 to 2200. The experimental result is presented in Fig. 5.

D. The Compromise Quality

In order to compare the compromise quality of these two algorithms, we denote the overall utility value and the Qualitative Pattern as Dim_1 and Dim_2 respectively. As Dim_2 is qualitative, in these experiments we use the index number in a ranking $\succ$, that satisfy the generated TCP-net, to compare the values of Dim_2 as results. There may be some incomparable outcomes in the TCP-net, so that the ranking $\succ$ may not be unique. However, in this experiment we can enforce the program to generate a TCP-net as complete as possible, and in practice we may utilize the preferences of historical users to amend the active user's preference, as done in [6]. For this study, we conducted four experiments and the experimental result is presented in Fig. 6, where the bigger utility value and the bigger qualitative index number, the better the results.

[2]http://lpsolve.sourceforge.net/

[3]http://sourceforge.net/projects/java-galib

E. Remarks

Obviously, the time efficiency of both GOLS and GAs are dependent on the parameters. The former depends on the number of quality levels of each NFP, and the latter depends on N_1, N_2. When both the size of service classes and the size of candidates of each class are large, the GOLS approach is preferred in terms of time efficiency. This is because the local selections of each class can be performed in parallel.

As can be seen in Fig. 6, the compromise quality of GAs are better than that of GOLS-1 and GOLS-2 in most cases. This is because the GAs can incorporate both qualitative and quantitative NFPs simultaneously.

VIII. Related Work

More and more research work started to focus on the NFP-aware web service composition and most of them consider quantitative NFPs only. In [12], the authors presented a broker-based framework, where there are four managers (Service Information Manager, Composition Manager, Selection Manager and Adaptation Manager) and two repositories (Services Repository and Process Repository). In [3], [4], the authors proposed a NFP-aware middleware for web services composition and two approaches to address the problem. They applied local optimization based on task-level selection and global optimization, which is based on global allocation of tasks to services using integer programming. However, the former cannot consider global tradeoffs between quality dimensions and cannot consider global constraints, and the latter overcomes these shortcomings at the price of higher computational cost. In [13], the authors presented a more efficient approach for combining global optimization with local selection by decomposing global constraints to local levels. In [5], the authors tried to solve this problem by using Genetic Algorithm. Skyline operator [2] has also been used to reduce the search space.

In recent years, many researchers have identified qualitative NFP preferences as the criteria for web service selection and NFP-aware composition. In [6], we have presented an approach of service selection that can handle incomplete and inconsistent qualitative user preferences, which are modeled as CP-net. We utilized the preferences of historical users to predicate and amend the active user's preference. In [18], the authors proposed two algorithms for qualitative preferences based service composition, i.e. Naive-Composer and Greedy-Composer. The former assumed that there is a Classic-Composer already and used the CP-net to select the optimal one from the result of Classic-Composer, while the latter incorporated the greedy strategy into the Classic-Composer. However, neither the naive nor the greedy approach considers the qualitative constraints.

To the best of our knowledge, most of the existing techniques for NFP-aware web service composition consider either quantitative NFPs or qualitative NFPs alone. In [7], we presented an approach for web service selection with quantitative and qualitative NFPs. However, we tried to select the optimal service from only one service class and we haven't considered the qualitative constraints there. This paper addresses the problem in a broader context.

IX. Conclusion

In this paper, we present a new NFP model considering both quantitative and qualitative NFPs simultaneously. We provide solutions to the service composition problem under the new NFP model. We present two algorithms, in which one combines global optimization with local selection and the other is based on genetic algorithms. As demonstrated by the experiments, our approaches are highly efficient and our genetic algorithm can achieve a good tradeoff between quantitative and qualitative NFPs.

Acknowledgment

This work is partially supported by JSNSF of China (No.BK2010417).

References

[1] OASIS. (2007, Apr.) Web services business process execution language. [Online]. Available: http://docs.oasis-open.org/wsbpel/2.0/wsbpel-v2.0.pdf

[2] M. Alrifai, D. Skoutas, and T. Risse, "Selecting skyline services for qos-based web service composition," in *Proceedings of the 19th international conference on World wide web*, ser. WWW '10. New York, NY, USA: ACM, 2010, pp. 11–20.

[3] L. Zeng, B. Benatallah, M. Dumas, J. Kalagnanam, and Q. Z. Sheng, "Quality-driven web services composition," in *12th International Conference on the World Wide Web (WWW)*. Budapest, Hungary: ACM Press, 2003, pp. 411–421.

[4] L. Zeng, B. Benatallah, A. Ngu, M. Dumas, J. Kalagnanam, and H. Chang, "Qos-aware middleware for web services composition," *Software Engineering, IEEE Transactions on*, vol. 30, no. 5, pp. 311 – 327, may 2004.

[5] G. Canfora, M. Di Penta, R. Esposito, and M. L. Villani, "An approach for qos-aware service composition based on genetic algorithms," in *Proceedings of the 2005 conference on Genetic and evolutionary computation*, ser. GECCO '05, 2005, pp. 1069–1075.

[6] H. Wang, S. Shao, X. Zhou, C. Wan, and A. Bouguettaya, "Web service selection with incomplete or inconsistent user preferences," in *ICSOC/ServiceWave*, 2009, pp. 83–98.

[7] H. Wang and W. Liu, "Web service selection with quantitative and qualitative user preferences," in *Web Intelligence and Intelligent Agent Technology (WI-IAT), 2011 IEEE/WIC/ACM International Conference on*, vol. 1, aug. 2011, pp. 404 –411.

[8] C. Boutilier, F. Bacchus, and R. I. Brafman, "Ucp-networks: A directed graphical representation of conditional utilities," in *UAI*, 2001, pp. 56–64.

[9] J. Lang, "Logical preference representation and combinatorial vote," *Ann. Math. Artif. Intell.*, vol. 42, no. 1-3, pp. 37–71, 2004.

[10] C. Boutilier, R. I. Brafman, C. Domshlak, H. H. Hoos, and D. Poole, "Cp-nets: A tool for representing and reasoning with conditional ceteris paribus preference statements," *Artificial Intelligence Research*, vol. 21, pp. 135–191, 2004.

[11] R. I. Brafman, C. Domshlak, and S. E. Shimony, "On graphical modeling of preference and importance," *J. Artif. Intell. Res. (JAIR)*, vol. 25, pp. 389–424, 2006.

[12] T. Yu and K.-J. Lin, "A broker-based framework for qos-aware web service composition," in *e-Technology, e-Commerce and e-Service, 2005. EEE '05. Proceedings. The 2005 IEEE International Conference on*, march-1 april 2005, pp. 22 – 29.

[13] M. Alrifai and T. Risse, "Combining global optimization with local selection for efficient qos-aware service composition," in *Proceedings of the 18th international conference on World wide web*, ser. WWW '09, 2009, pp. 881–890.

[14] K. Benouaret, D. Benslimane, and A. Hadjali, "On the use of fuzzy dominance for computing service skyline based on qos," in *Web Services (ICWS), 2011 IEEE International Conference on*, july 2011, pp. 540 –547.

[15] S. Forrest, "Genetic algorithms: principles of natural selection applied to computation," vol. 261, no. 5123, pp. 872–878, 1993.

[16] T. M. Mitchell, *Machine learning*, ser. McGraw Hill series in computer science. McGraw-Hill, 1997.

[17] E. Al-Masri and Q. H. Mahmoud, "Investigating web services on the world wide web," in *Proceedings of the 17th international conference on World Wide Web*, ser. WWW '08, 2008, pp. 795–804.

[18] G. Santhanam, S. Basu, and V. Honavar, "On utilizing qualitative preferences in web service composition: A cp-net based approach," in *Services - Part I, 2008. IEEE Congress on*, july 2008, pp. 538 –544.