Service Allocation for Composite Web Services Based on Quality Attributes

Shahram Esmaeilsabzali and Kate Larson
School of Computer Science, University of Waterloo
Waterloo, Ontario, Canada N2L 3G1
{sesmaeil,klarson}@cs.uwaterloo.ca

Abstract

Web services are software artifacts that can be accessed over the Internet. They can be seen as pay-per-view functionalities that are exposed by some service providers. If there are multiple providers for a Web service, then both the quality and price of the service become key factors when determining which provider to choose. We consider the problem of service allocation for multiple correlated Web services that can be serviced by potentially different Web service providers. We model this problem in a game-theoretic setting and design a reverse auction where optimal service allocation for the service requester is guaranteed. We also present an optimal strategy for the service provider when choosing its quality of service.

1 Introduction

Web services are an emerging technology for distributed computing. They are based on a uniform set of standards and promise a straightforward approach for developing systems that spread across multiple organizations. While Business to Business (B2B) systems are traditionally based on ad hoc sets of technologies, Web services promise an easy and standard approach for organizations to cooperate. A Web service has an interface and interested service requesters can invoke its functionalities over the Internet. Service providers can provide and charge for a certain service for different service requesters. Similar to other kinds of services, a Web service can be specified by its quality of service attributes. Such quality attributes are non-functional requirements of that Web service. In presence of multiple Web service providers it is usually desirable to choose the provider that promises better quality of service.

In this paper we provide a formalism that allows us to reason about the choice of optimal service provider. Unlike other works, where the price of the service is considered to be a quality attribute, we distinguish between price and other quality of service attributes. The distinguishing difference, between price and other quality of service attributes, is that often different quality attributes are not correlated, while price is usually related to other quality attributes. We find this separation of concerns very helpful, as it allows us to view the problem in a game-theoretic/mechanism design setting.

By considering the problem of quality-driven service allocation in a game-theoretic setting a number of interesting problems arise. The game can be viewed from either the service providers' and/or the service requesters' points of view. Each party would like to maximize its utility and seeks an optimal strategy. While the service requester considers the game as a single-round auction, the service provider views the game as a single-round auction that can potentially happen multiple times; this is since each service provider can potentially provide service for multiple service requesters. In this paper, we mainly consider this game from the service provider's point of view. In particular, we focus on the optimal strategy for determining the quality of service from the service provider's point of view.

Another interesting situation is when a certain service requester requires multiple different Web services within a budget constraint. This is a typical real-world problem in the context of quality-driven service allocation. In the presence of multiple service providers, the auctioneer is required to allocate Web services in such a way that maximizes the quality gained by the service requester, without violating the budget constraint. It turns out that this problem is NP-complete. However, by constraining the inputs, we present a *pseudo-polynomial* algorithm for the problem.

The rest of the paper is organized as following: in Section 2 we provide the necessary background materials that allow us to define the problem precisely. We also briefly talk about related works. In Section 3 we present our formalism for modeling the problem. In Section 4 we present an optimal strategy for the service provider in choosing its quality of service attributes. In Section 5 we present the reverse auction for service allocation, and in Section 6 we describe the winner determination algorithm for the auction. Finally, in Section 7, we present our conclusions and lay out

our plans for future works.

2 Background and Related Work

In this section we provide an overview of the background materials needed for this paper. We first quickly describe what is meant by a Web service and outline some of the problems that need to be addressed when using Web services. Our proposed solution for service allocation relies on a game-theoretic approach, so we give an overview of the key game-theoretic concepts in this paper. Also, we briefly discuss some related works and show how our work differs.

2.1 Web Services

"A Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format."[8] As such, Web services provide the grounds for a software development paradigm where functionalities can be discovered and invoked over the Internet, regardless of their network locations and implementation technologies/platforms. It is possible to coordinate multiple Web services to carry out a certain task. Such *composite Web services* are composed of multiple *atomic Web services* that are glued together via some workflow patterns [11]. Some simple workflow patterns are `If-Then-Else` and `While`, with the same semantics as their equivalents in programming languages.

Given a complex task, we want to find the atomic Web services that can carry out that task by collaborating among themselves. The problem of finding proper atomic Web services that can be composed together to satisfy a certain task can be categorized into two main sub-problems:

1. Searching for Web services based on some *functional requirements*. For example, a user might want to find an "English-German" dictionary Web service. If there is not any atomic English-German dictionary, then it is possible to use an "English-French" and a "French-German" dictionary in a pipelined manner to create a composite English-German dictionary [16].

2. Searching for Web services based on some *non-functional requirements*. For example, a user might want to find an English-German dictionary that is available 99.9% of the time.

There has been considerable research in search and composition of Web services based on their functional attributes [13, 3, 2, 16]. On the other hand, search for Web services based on their non-functional attributes is still an open area of research.

Atomic Web services that provide the same functionality, through similar interfaces, can subscribe to similar *service communities*[1] [2]. Interested parties in a functionality can then easily find their desired community and choose one of the Web services that are subscribed to that community. While all Web services under a specific community have the same functional properties, they do not necessarily have similar non-functional attributes. Some typical non-functional attributes for Web services are their execution time, and availability.

2.2 Game Theory Background

In this paper we propose using an auction mechanism to guide the negotiation process between service providers and requesters. However, before we describe the auction model, we first provide an overview of key game-theoretic concepts that are used in this paper. A more thorough presentation of game theory and auctions can be found in many game theory and economics texts [12].

We assume that there is a set of agents denoted by N (with $|N| = n$). Each agent $i \in N$ is defined by a *type*, θ_i, which represents the private information and preferences of the agent. The key concept in game theory is the *strategy*. A strategy for agent i, denoted by $s_i(\theta_i)$ is a contingency plan that specifies the action an agent should take at every point in the game when it has to take an action. For example, in an auction, the strategy of an agent would dictate what bid the agent should submit, given its type and the actions taken by other agents. A strategy profile $s = (s_1, \ldots, s_n)$ is a vector specifying one strategy for each agent in the game (we use the notation s_i instead of $s_i(\theta_i)$ when it is clear from the context). A strategy profile determines how the game is played and thus which outcome $o(s) \in \mathcal{O}$ will occur. For example, in an auction an outcome would state which agent was the winner and how much money must be exchanged. Each agent will have a preference over outcomes and will try to choose a strategy so that its preferred outcomes occurs. These preferences are expressed in terms of *utility functions* where $u_i(o(s), \theta_i)$ is some real number and if agent i prefers outcome o_1 to outcome o_2 then $u_i(o_1, \theta_i) > u_i(o_2, \theta_i)$. The goal of each agent is to maximize its utility.

Game theory is interested in finding equilibria. An equilibrium is a strategy profile which satisfies certain properties. The most well known equilibrium concept is the *Nash equilibrium*. A strategy profile is a Nash equilibrium if no agent has incentive to change its strategy given that the other agents do not change. While this concept is fundamental in

[1] While we find the idea of service community particularly useful in this paper, we do not rely on its underlying technologies. Any other mechanism, such as `UDDI tModel`'s [1], that can categorize Web services based on their interface and functional specifications, fits into our solution.

game theory, it does have several weaknesses. First, there may be multiple Nash equilibria in a game and so agents may be uncertain as to which equilibrium to play. Second, it assumes that all agents have perfect information about every other agent in the game. These weaknesses limit the practicality of the Nash equilibrium for many applications. A stronger equilibrium concept is the *dominant strategy equilibrium*. A strategy is a dominant strategy for an agent if it is the agent's best strategy (i.e. utility maximizing strategy) no matter what strategies the other agents follow. A dominant strategy equilibrium is a strategy profile $s^* = (s_1^*, \ldots, s_n^*)$ where for all i, s_i^* is a dominant strategy. Games with dominant strategy equilibria are easy for agents to play since it is obvious what their optimal strategy is and they do not need to worry about what the other agents are doing. While many games do not have dominant strategy equilibria, in many practical applications it is possible to carefully *design* games in order to guarantee that dominant strategies exist. In particular, auctions can often be carefully designed so that the optimal bidding strategies of the agents are such that agents are best off truthfully telling their true preferences to the auctioneer. An example of such an auction is the famous Vickrey auction [17].

2.3 Auctions and Web Services

A Web service requester may require some non-functional criteria for its desired Web service. Looking for an English-German dictionary that is available 99.9% of times, costs 1c per translation, and performs its task in 0.1 sec, is an example of a request with some non-functional criteria. However, such non-functional criteria may be met by more than one Web service. This situation resembles a *single-item*, *single-round* auction. In particular, this is a *reverse* or *procurement* auction where there is one item to be bought (the service) and there are multiple sellers of the service (providers). Each seller will submit a bid which states how much it wants to be paid for providing the service. This is done without knowing what the other bids are. The auctioneer chooses the winning seller (provider) among the bidders by choosing the one who submitted the lowest bid.

In the application studied in this paper we are not only interested in the price of services, but are also interested in the quality of the service. Our auction thus resembles a *multidimensional* [5] or *multi-attribute* [6] auction. In these auctions the buyer is not only interested in getting the service at the best price, but also cares about the quality of the service. The buyer has a *utility function* that depends on both the price and the set of quality attributes. The winning bidder in the auction is the one which submits a bid that maximizes the buyers utility, instead of simply providing the lowest price.

A Web service provider, belonging to a service community, can provide service for more than one service requester; this is similar to many other digital goods. Web Services, and similar digital goods, could face the so-called *unknown market* phenomenon where we do not have any solid knowledge about the distribution of requests. *Competitive auctions* aim at designing efficient auctions that are capable of dealing with such markets [7]. However, we find the assumption of "unknown market" unrealistic. Unlike, the market for some digital goods (for example, songs in mp3 format), for Web services it is reasonable to assume that the trend and distribution of requests are known. A service community can represent a number of Web service providers and, as such, is able to collect statistical information about the requests. In this paper, we adhere to such an assumption.

2.4 Related Work

Few approaches have been proposed for quality-driven service allocation; all of which basically solve an optimization problem by using different methods. To the best of our knowledge, no Web service allocation approach considers the problem in a game-theoretic and economics-aware setting. Furthermore, none of the approaches consider the problem of policy selection (strategy selection) for service providers and/or service requesters.

Zeng et al. [19] propose an approach that relies on linear programming for maximizing the desired non-functional qualities that a service requester is interested in. Their approach deals with the case where there is one service requester and a set of service providers. They assume a pre-determined quality and price for Web service providers, which does not capture the essential economic dynamics of the problem.

Chen et al. realize the necessity of a "broker" for mediating between the service providers and the requester [9]. The broker is responsible for choosing the optimal service provider for a service requester, based on the information that it collects about the quality of service of different service providers. The approach is proposed for multimedia Web services and considers only atomic Web services. In work by Yu and Lin, this framework is extended to model service selection for composite Web services [18]. The optimization problem is modelled and solved as a variation of knapsack problem.

While in the two papers just discussed the role of a broker is realized, the broker is mainly an entity for verifying that service providers comply with their declared quality levels [9, 18]. We believe that such quality of service monitoring can be achieved more efficiently by an independent entity, such as the *certifier* role proposed by Ran [14]. The broker has only a limited computation and space ca-

pacity, and furthermore can not possibly know about all different service providers' profiles. "Certifiers", on the other hand, can be integrated into UDDI infrastructure and provide seamless way of quality of service monitoring.

3 Modeling Service Requesters and Service Providers

In this section we propose an approach for modeling service requesters, service providers, their quality attributes and prices. Our approach is inspired by models in Che [5] and David et al [6]. While Che proposes a model for capturing the quality as well as the price of items, David et al extend that model to deal with items specified with multiple quality attributes.

We choose to consider two quality attributes for Web services, namely, *reliability* and *availability*. Reliability represents the level at which the service provider is committed to provide the service within the range of promised response time. Availability represents the level at which the service provider is committed to be accessible for service requesters. It is possible to extend this model to specify more than two quality attributes.

In this paper we are not concerned with how such quality attributes are measured. For our purpose, it suffices that they are each represented by a real number, which is greater than or equal to zero. Also, we assume that service providers announce their quality attributes correctly. In presence of certifiers [14], it is possible to verify the announced quality attributes.

In the following subsections, we provide models for formally specifying service requesters and service providers.

3.1 Service Requester

A service requester is interested in service i that is specified by its quality attributes q_{iR} and q_{iA}. Quality attributes, q_{iR} and q_{iA} represent the level of reliability and availability of service i respectively.

A contract, $\langle q_{iR}, q_{iA}, p_i, N_i, f_i(t)\rangle$, is proposed by a service provider and is based on an initial request by the service requester. It specifies the details of a service contract between a service provider and a service requester. It states that the service provider would provide service i according to the contract, and the service requester would pay p_i for every invocation of service i. We assume that the duration of contract is infinite. In the following we specify each item of the contract in more details.

- q_{iR}, q_{iA} are zero or positive real numbers that specify the minimum level of qualities for reliability and availability that the service provider is committed to provide.
- p_i is the price that the service requester should pay *each* time service i is requested and is fulfilled by the service provider.
- N_i is the maximum number of service requests that the service requester can possibly request during one day.
- $f_i(t)$, where $0 \leq t \leq 24$, is a continuous function representing the maximum number of active i services that the service requester can ask or have active at any given point of time. Obviously, $f_i(t) < N_i$ should hold for all $0 \leq t \leq 24$. The service provider can reject a service request if the rate of service requests exceeds the quota specified by $f_i(t)$. We also require that the total number of daily requests should not exceed N_i.

The maximum response time for contract i, d_i, could then be computed as following:

$$d_i = \int_0^{24} f_i(t)\, dt / N_i$$

A service provider can win a contract for service i only if it is always able to provide the service in maximum response time d_i.

Each proposed contract provides a certain amount of utility for the service requester. This utility is computed by the service requester's *utility function*, $U(q_{iR}, q_{iA}, p_i)$. A utility function relies on an *evaluation function*, $V(q_{iR}, q_{iA})$, which specifies the value the service requester gains from being provided service i at quality levels q_{iR} and q_{iA}. The utility function can then be defined as following:

$$U(q_{iR}, q_{iA}, p_i) = V(q_{iR}, q_{iA}) - p_i$$

The service requester prefers a service provider that provides a higher utility. We assume that $V(q_{iR}, q_{iA})$ is publicly available to the service providers and require the following properties to hold for it:

- $V'_{q_{iR}} > 0$
- $V'_{q_{iA}} > 0$
- $V''_{q_{iR}} < 0$
- $V''_{q_{iA}} < 0$
- $\lim_{q_{iR}\to 0} V'_{q_{iR}}(q_{iR}, q_{iA}) = \infty$
- $\lim_{q_{iA}\to 0} V'_{q_{iA}}(q_{iR}, q_{iA}) = \infty$
- $\lim_{q_{iR}\to \infty} V'_{q_{iR}}(q_{iR}, q_{iA}) = 0$
- $\lim_{q_{iA}\to \infty} V'_{q_{iA}}(q_{iR}, q_{iA}) = 0$

Intuitively, the above properties state that the evaluation function is increasing on quality attributes; which is a logical and desirable requirement. Furthermore, in this paper we are not considering strategies for service requesters and thus assume that the evaluation function, that is shared with service providers, is the real internal utility function. In other words, we assume that the service requester is truthful about its declared utility function.

An example of an evaluation function which satisfies our requirements is the logarithmic function. We believe that the service requester is interested in service quality to a certain extent; once a *reasonable* level of quality is provided, any extra quality would not make the service requester significantly happier. This can be captured by natural logarithm function. While the analysis in this paper assumes only that the utility function satisfies the properties outlined above, we use the natural logarithm function to illustrate. That is, we assume that

$$U(q_{iR}, q_{iA}, p_i) = w_{iR}.\ln(q_{iR}+1) + w_{iA}.\ln(q_{iA}+1) - p_i$$

where weights are specified for each of the desired quality attributes by w_{iR} and w_{iA}. Notice that when the value for a quality is zero, the service requester is not gaining any utility from that quality attribute.

3.2 Service Providers

A service provider j may win contract $\langle q_{iR}, q_{iA}, p_i, N_i, f_i(t)\rangle$, if it promises to provide service i based on the contract. Naturally, service providers would have to afford a greater cost when providing higher quality services. We define the cost function of service provider j as $c_j(q_{iR}, q_{iA}, \theta_j)$, where θ_j is the *cost parameter*. We assume that θ_j is not known to the service requesters but instead is identically and independently distributed over the range $[\underline{\theta}, \overline{\theta}]$. We require $c_j(q_{iR}, q_{iA}, \theta_j)$ to be increasing in q_{iR}, q_{iA}, and θ_j. This means that the cost of service increases as its quality and cost parameters increase.

The profit of providing service i by service provider j, is then defined as following:

$$\pi_j(q_{iR}, q_{iA}, p_i) = p_i - c_j(q_{iR}, q_{iA}, \theta_j)$$

In this paper we assume that there is an exponential relation between the cost of the service and its desired quality attribute. That is we assume that

$$c_j(q_{iR}, q_{iA}, \theta_j) = \theta_j.(a_j.e^{q_{iR}} + b_j.e^{q_{iA}})$$

where a_j, b_j are empirical constants, specific for each service provider. Our general analysis does not rely on an exponential cost function. However, studies have shown that as the size, complexity, and quality attributes of systems increase, exponential efforts are required [4]. Thus, we believe that our cost function is well suited to model service oriented software.

4 Service Providers Strategy

Each service provider j can provide service i for multiple service requesters. As such, there are a number of issues to be addressed.

- What level of quality should the service provider commit to provide? It is important to notice that once the service provider commits to a certain level of quality, it can not decrease that level. This leads to a serious decision that should be made by the service provider in the very early stage of accepting or rejecting the *first* contract.
- Which service requests should j accept? It could be the case that by choosing a certain service request, j would not be able to accept some more profitable requests.
- What is the optimal price for a contract? This obviously relies on the service provider's cost function, service requester's evaluation function, and also other service providers' strategies.

In this paper, we do not address the problem of optimal pricing for the service provider. Instead, we focus on how we can find an equilibrium for the quality such that it is the most beneficial quality for the service provider. To reason about the quality, we need to know how important a certain quality is for service requesters. In single item, multi-attribute auctions it is often assumed that the seller knows the exact weights for quality attributes from the buyer's perspective [5, 6]. In the context of our work, this approach is not helpful; there exists more than one service requester being serviced by a single service provider. We believe it is reasonable to assume that the service provider knows the distribution of the weights that service requesters have for different quality attributes. Furthermore, we believe that it is natural for such distributions to be normal distributions. The majority of service requesters are interested in a *reasonable* level of quality (*mean* value of the distribution), and only a small number of potential service requesters are interested in extreme weights. We believe that the parameters for such normal distributions can be acquired from the service communities. Since service communities can have access to a large number of service transaction logs, provided by different service providers. We consider the distribution of weights based on each single service transaction, rather than based on each contract. This is helpful because each contract can potentially represent a large number of service

requests and it is desirable that larger contracts contribute more to the values of parameters in the distribution. In this way, each contract with N_i number of services will contribute N_i times to the normal distribution.

We consider the following normal distributions for w_{iR}, and w_{iA}:

$$\begin{aligned} N_{i1}(w_{iR};\mu_1,\sigma_1) &= \frac{1}{\sigma_1\sqrt{2\pi}}e^{-(w_{iR}-\mu_1)^2/(2\sigma_1^2)} \\ N_{i2}(w_{iA};\mu_2,\sigma_2) &= \frac{1}{\sigma_2\sqrt{2\pi}}e^{-(w_{iA}-\mu_2)^2/(2\sigma_2^2)} \end{aligned} \quad (1)$$

Using the above distributions, we propose an optimal strategy for service providers to choose their level of quality attributes. In fact, there exists an equilibrium for choosing the level of quality attributes that is independent from the price of the service. This may seem counter-intuitive. However, we can generate a contract with the same quality as the optimal quality which yields the same evaluation value. This strategy, however, requires that service providers' profit function and service requester's utility function are defined such that the two quality attributes are not correlated. This is crucial since it allows us to analyze each quality independently from the other one. Observe that our proposed utility and profit functions comply with that criteria.

We use the distributions for the quality weights (equation 1) to define the optimal strategy for choosing quality attributes for service providers. In presence of the distributions for weights of qualities, we choose a strategy that makes the service provider better off in general, as opposed to a strategy that only makes the service provider better off in a single contract. In the following proposition, we introduce our strategy and show that it is not beneficial for service providers to deviate from it.

Proposition 1 *To maximize the chance of winning for service provider j, in procurement for Web service i, there is a dominant strategy for j to choose its quality attributes as following:*

$$opt\_q_{iR} = \texttt{argmax}_{\texttt{q}_{\texttt{iR}}} \int_0^\infty (N_{i1}(w_{iR};\mu_1,\sigma_1).w_{iR}. \ln(q_{iR}+1) - \theta_j.a_j.e^{q_{iR}})dw_{iR} \quad (2)$$

$$opt\_q_{iA} = \texttt{argmax}_{\texttt{q}_{\texttt{iA}}} \int_0^\infty (N_{i2}(w_{iA};\mu_2,\sigma_2).w_{iA}. \ln(q_{iA}+1) - \theta_j.b_j.e^{q_{iA}})dw_{iA} \quad (3)$$

Proof. First, observe that by independently maximizing quality attributes, as shown in equations 2 and 3, we also maximize the value of following:

$$\int_0^\infty \int_0^\infty ((N_{i1}.w_{iR}.\ln(q_{iR}+1) + N_{i2}.w_{iA}.\ln(q_{iA}+1) - c_j(q_{iR},q_{iA},\theta_j))dw_{iR})\, dw_{iA} \quad (4)$$

This is true since each term in (4) is either specified in q_{iR} or q_{iA}, but not both. Therefore, by independently maximizing the above terms with respect to q_{iR}, and q_{iA}, we also maximize the whole term. Also, observe that the integration in equation 4 represents the summation of different $(V(\cdot) - c_j(\cdot))$ values based on distributions for w_{iR} and w_{iA}. By maximizing $(V(\cdot) - c_j(\cdot))$, in fact, we try to maximize the evaluation by the service requester and minimize the cost of the service provider. In this way, by choosing quality values that maximize integration 4, we also make the service provider more competitive in winning different auctions.

Now, by contradiction, we show that there does not exist any other contract that would provide a better chance for the service provider to win the procurement. Suppose that there exists a *better* strategy, than choosing opt_q_{iR} and opt_q_{iA}, and it makes the service provider better off in the competition. Let that strategy have quality attributes q_{iR}, q_{iA}, and price p_i. On the other hand, consider another strategy with quality attributes q'_{iR} and q'_{iA}, equal to our proposed optimal quality levels, and price p'_i. We define this strategy such that it has the same utility value as the *better* strategy:

$$\begin{aligned} &q'_{iR} = opt\_q_{iR} \\ &q'_{iA} = opt\_q_{iA} \\ &p'_i = p_i + V(opt\_q_{iR}, opt\_q_{iA}) - V(q_{iR}, q_{iA}) \end{aligned}$$

As shown in the following, $U(q_{iR},q_{iA},p_i) = U(q'_{iR},q'_{iA},p'_i)$; meaning that the *better* strategy can not be evaluated, by the service requester, higher than the optimal proposed strategy.

$$\begin{aligned} U(q'_{iR},q'_{iA},p'_i) &= V(q'_{iR},q'_{iA}) - p'_i \\ &= V(opt\_q_{iR},opt\_q_{iA}) - [p_i + \\ &\quad V(opt\_q_{iR},opt\_q_{iA}) - V(q_{iR},q_{iA})] \\ &= V(q_{iR},q_{iA}) - p_i \\ &= U(q_{iR},q_{iA},p_i) \end{aligned}$$

For specific w_{iR} and w_{iA}, it is not hard to see that p'_i will be positive and provides a positive profit, if q_{iR}, q_{iA}, and p_i have a positive profit. Considering all possible quality weights, however, we may not always have a positive p'_i. Instead, by following the optimal quality values we have the highest expected chance of winning a contract. In this way, we have shown that by following the above strategy, in choosing values for quality attributes, we can provide a competitive contract and this concludes our proof. □

We have shown that service providers have a dominant strategy which determines how they should choose their quality attributes. This strategy also maximizes the expected profit for the service providers.

5 Service Allocation: A Reverse Auction

In this section we look at the problem of how to decide which service providers to use if there are multiple providers offering different types of services with different qualities. We propose using an auction mechanism to help in the allocation. In the previous section we showed that the service providers have dominant strategies when it comes to setting their quality attributes. This simplifies the bidding problem for the providers and allows us to focus on the details of the auction mechanism itself.

Consider the problem of service allocation for a composite Web service. A composite Web service needs n simple Web services to accomplish its task and is willing to pay maximum P for obtaining those services. The value P is the service requester's budget. We assume that P is unknown to the service providers.

Assuming that there exists appropriate service communities for each of those Web services; for each service i, where $1 \leq i \leq n$, there could exist $k_i > 0$ potential service providers. For each desired service i, the service requester specifies N_i, $f_i(t)$, and $V(q_{iR}, q_{iA})$. On the other hand, each potential service provider j would propose a contract $\langle q_{iR}, q_{iA}, p_i, N_i, f_i(t)\rangle$; stating that it is willing to provide service i with quality attributes q_{iR} and q_{iA}, at price p_i.

Considering k_i potential service providers for service i, we define the following notations:

$v_i[l] = V(q_{iR}, q_{iA})$ where q_{iR}, q_{iA} are qualities proposed by service provider l for providing service i, where $(1 \leq i \leq n)$, and $(1 \leq l \leq k_i)$

$p_i[l] = p_i$, where p_i is the price proposed by l for providing service i

We assume that there exists an independent non-self-interested auctioneer that is authorized, on behalf of service requesters and service communities, to hold our proposed reverse auction. The auctioneer keeps P, the budget, secret and makes the rest of information publicly available for all service providers. Notice that service requests can always be specified by an `OR-of-XOR` bid clause [15].

$$
\begin{aligned}
[\ &((v_1[1], p_1[1])\ \texttt{XOR}\ \cdots\ \texttt{XOR}\ (v_1[k_1], p_1[k_1]))\ \texttt{OR}\\
&((v_2[1], p_2[1])\ \texttt{XOR}\ \cdots\ \texttt{XOR}\ (v_2[k_2], p_2[k_2]))\ \texttt{OR}\\
&\cdots\ \texttt{OR}\\
&((v_n[1], p_n[1])\ \texttt{XOR}\ \cdots\ \texttt{XOR}\ (v_n[k_n], p_n[k_n]))\ ]
\end{aligned}
$$

Intuitively, the above request states that the service requester is interested in choosing exactly one service provider for each of its desired Web services. The challenge is then how to choose n service providers such that the service requester obtains the highest utility and the cost does not violate budget P. In the next section, we present an algorithm for this problem.

6 Winner Determination Algorithm

In this section, we first show that the problem of service allocation, as described in the previous section, is NP-complete. We then propose a *pseudo-polynomial* algorithm that does the allocation appropriately.

The input parameters to the winner determination algorithm are:

1. P: which is the total budget of the service provider.
2. $((v_1[1 \ldots k_1], p_1[1 \ldots k_1]), (v_2[1 \ldots k_2], p_2[1 \ldots k_2]), \cdots, (v_n[1 \ldots k_n], p_n[1 \ldots k_n]))$: that specifies different service providers' offers for different required services.

This problem can be shown to be NP-complete.

Proposition 2 *Winner determination problem is NP-complete.*

Proof. It is trivial to show that winner determination is NP. We show that knapsack problem can be reduced to the winner determination problem in polynomial time. The reduction is quite straightforward. Considering an instance of knapsack problem $\langle w[1 \cdots m], v[1 \cdots m], c\rangle$, where c is the maximum possible weight, we can create an instance of winner determination problem, in polynomial time, as shown in Algorithm 1. The idea of our reduction is to simply define m pairs of services where the first service in the pair is a dummy service and the second one is an item in the knapsack problem. By introducing dummy services we make the winner determination solve the knapsack problem. Recall that in knapsack problem, we are interested in maximizing the value without violating the weight constraint. To maximize the value, it is possible either to choose or not to choose a certain item. Our reduction models "not choosing" by dummy values, the first element of the pair, and "choosing" by the second element of the pair.

Algorithm 1 KnapsackToWinners($w[1 \ldots m], v[1 \ldots m], c$)

$P = c$
for $i = 1$ to m **do**
  $v_i[1] = 0, p_i[1] = 0$
  $v_i[2] = v[i]$
  $p_i[2] = w[i]$
end for
return($\langle (v_1[1 \cdots 2], p_1[1 \cdots 2]), \cdots, (v_m[1 \cdots 2], p_m[1 \cdots 2]), P\rangle$)

□

Kelly has shown that the winner determination in the context of service allocation problem is deeply related to the general knapsack problem [10]. While he mainly considers

the problem in a combinatorial auction setting, we introduce a different instance of the service allocation problem, in a non-combinatorial setting. Our winner determination problem supports Kelly's observations about the inherent connection between the two problems.

6.1 Algorithm

In this subsection we propose a pseudo-polynomial time algorithm for winner determination problem. By considering P, the service requester's budget, to be a constant integer value and assuming that the prices are all integer values, we can achieve an efficient algorithm. These assumptions are not really restrictive since in real applications, prices are integers or we can easily scale them to integer values. Furthermore, it is feasible to determine a reasonable ceiling for the maximum possible budget. In the following algorithm, we introduce a dynamic programming algorithm that determines the winners of the procurement efficiently.

Consider an instance of winner determination problem, as defined in the beginning of this section. Let $M[i, b]$ be the maximum quality obtainable by choosing first i services within the budget limit b. We would then like to compute $M[n, P]$, i.e. the optimal allocation for all services within budget P. There is a recursive definition for computing the M values, as following:

$$M[i, b] = \texttt{Max}(M[i-1, b-p_i[j]]+v_i[j]) \text{ where } 1 \leq j \leq k_i$$

Index i is between zero and n. Value zero for i is a helper value and allows us to appropriately define our dynamic programming algorithm. We start populating the M values in a bottom-up fashion; starting with the situation where only one service allocation is desired with budget limit of zero. We increase the budget up to P and then continue with optimal allocation for the first two services; computation continues until we reach $M[n, P]$, that is the optimal allocation for the first n services within budget P. Algorithm 2 shows this process precisely.

Notice that we need to assure that the previously computed M values are non-zero, otherwise it would mean that there does not exist any allocation for that particular M item. The exception, however, is when we are computing $M[1, j]$ values, i.e. optimal allocation for the first service. In that case, obviously, zero values are correct.

Note that it is trivial to keep track of the optimal allocations by introducing an extra array.

Clearly, the time complexity of the algorithm is $n \times P \times \sum_{i=1}^{n} k_i$. As long as, P can be pre-determined and is not too large, we expect this algorithm to be efficient.

Algorithm 2 ServiceProviderAllocation($(v_1[1 \dots k_1], p_1[1 \dots k_1])$, $(v_2[1 \dots k_2], p_2[1 \dots k_2]), \cdots,$ $(v_n[1 \dots k_n], p_n[1 \dots k_n])$, P)

```
for i = 0 to n do
  for j = 0 to P do
    M[i, j] = 0
  end for
end for
for i = 1 to n do
  for j = 0 to P do
    for l = 1 to k_i do
      if (M[i, j] < M[i − 1, j − p_i[l]] + v_i[l]) and
      ((i == 1) or (M[i − 1, j − p_i[l]] ≠ 0)) then
        M[i, j] = M[i − 1, j − p_i[l]] + v_i[l]
      end if
    end for
  end for
end for
return M[n, P]
```

7 Conclusions and Future Works

In this paper, we formally studied the problem of Web services allocation, based on their quality attributes, in a game-theoretic and economics-aware setting. We showed that it is beneficial for the service provider, in long term, to have a certain level of quality. Our proposed dominant strategy for quality attributes is independent from the price that the service provider proposes. Our strategy relies on the distribution information of the service requesters' interests (weights) for different quality attributes. We consider two quality attributes, namely, reliability and availability; however, it is easy to extend our model to support more quality attributes.

Composite Web services rely on functionalities of multiple simple Web services and as such appropriate services should be allocated to them. We consider the service allocation of composite Web services when quality attributes are important and there is a budget constraint. In the presence of multiple service providers, this problem is NP-complete. We present a reverse auction for service allocation that optimally allocates the services for the service requester. We refine the problem's inputs and present a pseudo-polynomial algorithm that can efficiently determine the winners of the auction.

Our future works will be focused on the following directions:

- Considering the best strategy for the service requester when revealing its preferences, i.e. utility function, in this paper, we assumed that the service requester is truthful. This may be too restrictive and is not nec-

essarily the most beneficial policy for the service requester. We are interested in investigating the possible strategies for the service requesters.

- Defining a strategy for the service provider in determining its price. One of the interesting issues is that this strategy must consider $f_i(t)$ as one of its important parameters. While most service requesters could be interested in services during a specific period of day (e.g. morning to noon) some service requesters may seek the same service at odd times of the day (e.g. after midnight). The proposed strategy should have a mechanism to punish the former and reward the latter service requesters.

- We assumed that once the quality of service for a certain provider is set, it can not be changed. However, it might be possible to *increase* that level of quality. The service provider may realize, through some learning mechanisms, that increasing the quality of service, at some point, is more beneficial. We would like to study the mechanisms that enable us to realize when and how we should increase the quality of service to gain more profit.

- With a composite Web service, the value for a certain quality attribute for that composite Web service relies on its constituent atomic Web services. If we have an atomic Web service with a low quality, it could undermine other potentially high quality Web services, and thus the quality of composite Web service itself. As such, the service allocation could be enhanced to allocate the services consistently. In other words, quality attributes for different atomic Web services would better be in a reasonable range, otherwise, some precious quality of services along with service requester's money could be used inefficiently.

Acknowledgments

The first author would like to thank Ehsan Chiniforooshan for helpful discussions about algorithmic aspects of this work.

References

[1] T. Bellwood and et al. UDDI Version 3.0.1 UDDI Spec Technical Committee Specification. UDDI Committee Specification, Oct. 2003.

[2] B. Benatallah, M. Dumas, Q. Z. Sheng, and A. H. H. Ngu. Declarative Composition and Peer-to-Peer Provisioning of Dynamic Web Services. In *18th IEEE International Conference on Data Engineering (ICDE)*, 2002.

[3] D. Berardi, D. Calvanese, G. De Giacomo, M. Lenzerini, and M. Mecella. Automatic Composition of e-Services that Export their Behavior. In *Proceedings of the 1st International Conference on Service Oriented Computing (ICSOC 2003)*, pages 43–58, 2003.

[4] B. W. Boehm. *Software Engineering Economics*. Prentice Hall, 1981.

[5] Y.-K. Che. Design competition through multidimensional auctions. *RAND Journal of Economics*, 24(4):668–680, Winter 1993.

[6] E. David, R. Azoulay-Schwartz, and S. Kraus. Protocols and strategies for automated multi-attribute auctions. In *Proceedings of the 1st International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 77–85, 2002.

[7] A. Goldberg, J. Hartline, A. Karlin, and A. Wright. Competitive auctions and digital goods. In *Proceedings of 12th Annual Symposium of Discrete Algorithms (SODA 2001)*, 2001.

[8] H. Haas and A. Brown. *Web Services Glossary*, 2004.

[9] T. Y. Hongan Chen and K.-J. Lin. QCWS: an implementation of QoS-capable multimedia Web services. In *Proceedings of IEEE 5th Symposium on Multimedia Software Engineering*, pages 38–45, 2003.

[10] T. Kelly. Generalized knapsack solvers for multi-unit combinatorial auctions: Analysis and application to computational resource allocation. Technical Report HPL-2004-21, Hewlett Packard Laboratories, 2004.

[11] D. Martin, M. Burstein, J. Hobbs, O. Lassila, D. McDermott, S. McIlraith, S. Narayanan, M. Paolucci, B. Parsia, T. Payne, E. Sirin, N. Srinivasan, and K. Sycara. OWL-S: Semantic Markup for Web Services. W3C Member Submission, Nov 2004.

[12] A. Mas-Colell, M. Whinston, and J. Green. *Microeconomic Theory*. Oxford University Press, 1995.

[13] S. Narayanan and S. A. McIlraith. Simulation, verification and automated composition of web services. In *Proceedings of the 11th International Conference on World Wide Web*, pages 77–88, 2002.

[14] S. Ran. A model for web services discovery with QoS. *SIGecom Exch.*, 4(1):1–10, 2003.

[15] T. Sandholm. Algorithm for optimal winner determination in combinatorial auctions. *Artificial Intelligence*, 135(1-2):1–54, 2002.

[16] E. Sirin, J. Hendler, and B. Parsia. Semi-automatic composition of web services using semantic descriptions. In *Web Services: Modeling, Architecture and Infrastructure workshop in ICEIS 2003*, Angers, France, April 2003.

[17] W. Vickrey. Counterspeculation, auctions, and competitive sealed tenders. *Journal of Finance*, 16:8–37, 1961.

[18] T. Yu and K.-J. Lin. Service selection algorithms for Web services with end-to-end QoS constraints. In *Proceedings of IEEE International Conference on e-Commerce Technology*, pages 129–136, 2004.

[19] L. Zeng, B. Benatallah, M. Dumas, J. Kalagnanam, and Q. Z. Sheng. Quality driven web services composition. In *Proceedings of the 12th International Conference on World Wide Web*, pages 411–421, 2003.