6

A Taxonomy of QoS Management and Service Selection Methodologies for Cloud Computing

Amir Vahid Dastjerdi

The University of Melbourne, Australia

Rajkumar Buyya

The University of Melbourne, Australia

CONTENTS

A problem that has become one of the recent critical issues in service computing is automating web services selection. Web service selection has been applied in different computing paradigms namely as SOA, and Grid. A number of approaches with a variety of architectures and algorithms have been proposed to resolve the problem. The aim of this chapter is to create a comprehensive taxonomy for web service selection solution, and apply this taxonomy to categorize selection approaches. In addition, the survey and taxonomy results are presented in a way to identify gaps in this area of research. Furthermore, it indicates how web service selection approaches in SOA and Grid can share their contributions to tackle selection problems in Cloud.

6.1 Introduction

A web service [14] is a piece of software interface that can be invoked over the Internet and can be roughly viewed as a next-generation successor of the Common Object Request Broker Architecture (CORBA) [15] or the Remote Procedure Call (RPC) [763] technique. The main benefits of web services are interoperability, ease of use, reusability and ubiquitous computing. Currently, an increasing number of companies and organizations implement their applications over the Internet. Thus, the ability to select and integrate inter-organizational and heterogeneous services on the Web efficiently and effectively at runtime is an important step toward the development of Web service applications. Recently, researches study how to specify (in a formal and expressive enough language), compose (automatically), discover, select and ensure the correctness of web services [689]. The typical architecture of web services includes three roles, namely service requestor, service broker and service provider. Once the requestor sends a service request to the broker, a matched service provider should be sent back a by the broker from a published service metadata repository. When the broker finds a set of services which all satisfy the functional requirements of a service request, then the crucial issue is how quality of service (QoS) requirements should be used to select the most proper service.

Web service selection has been applied in different computing paradigms namely as SOA, and Grid. That is because a resource can be represented by web services and web service selection approaches can help assigning each request in the queue to the most proper resource in a fair manner. In addition, recently Cloud Computing has emerged as a promising paradigm, in which the platform and even the infrastructure are represented as service, therefore Cloud selection plays a significant role and will attract lots of attention. Considering the popularity of Cloud computing, it is highly possible to conduct redundant researches in this area. There are many works that in the context of SOA and Grid web service selections, can share their contribu-

tions to tackle selection problems in the Cloud. Consequently, the aim of this taxonomy is:

- To identify state-of-the-art challenges in web service selection and extract a general model for service selection.
- To classify works on how they are modeling QoS attributes and how they approach selection including investigation of their advantages and disadvantages.
- To investigate what are new challenges in web service selection special in the context of Cloud computing, and what can be inherited works in the context of Grid and SOA.

6.2 General Model of Web Service Selection

Based on a survey of web service selection literatures, a general model for service selection has been extracted. The model consists of two parts. The first part describes steps that have to be considered for QoS management, and the second part is deals with the process of web service selection based on provided QoS preferences and service descriptions.

6.2.1 QoS Management

As depicted in Figure 6.1, several steps have to be taken into account for QoS management. In general, a QoS management approach for QoS-aware Web service selection can include following phases:

1. **Identifying Roles**: Provider and client (requestor) are the two basic roles in all service selection problems. The objective of selection is determined based on the defined roles in the problem. Selection solutions usually try to maximize provider, requestor, or both profits. In some problems other roles such as monitoring party or other third parties can be identified.
2. **QoS Modeling**: QoS can be described in service requestor's preferences to express their expectations more. It also can be included in a web service advertisement when there are different WS providers that present diverse versions of services to answer varying requirements of their customers. The requestor's requirements and the service offerings have both functional and complex non-functional aspects, which need to be expressed and then for evaluation purposes of service and request, matched against each other [785]. Moreover,

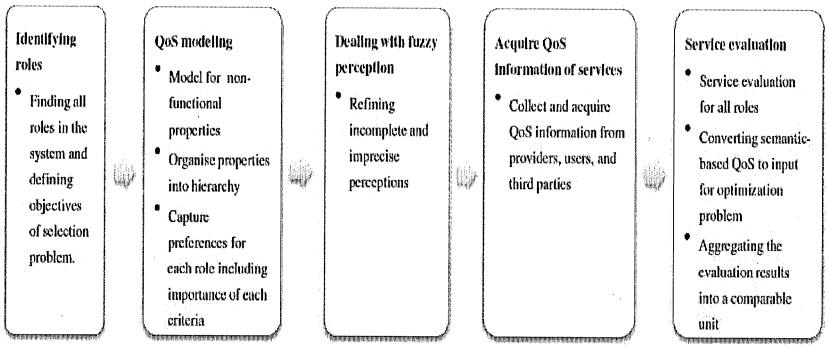

FIGURE 6.1
QoS Management Process.

the model should give the user the ability to express which QoS criteria are more important for them and a way to define the relations between QoS criteria. For example, one user may prefer a service with better response time compared to a service with a lower price and the other may like the cheaper service better. In addition to the model, it increases the transparency if we place QoS properties into hierarchical structure. For example, in the hierarchy throughput and quality, properties are performance aspects, while security and privacy are both safety aspects. Usually work uses ontology to create the hierarchy [713, 715].

3. **Taking Care of Customers' Fuzzy Perceptions**: Dealing with fuzzy perceptions of customers is another important task in managing QoS preferences of requestors. That is because brokers usually face user's perceptions which are incomplete and imprecise [745] (due to complexity and mistrusted information).
4. **Collecting QoS Information**: In this phase, a proper approach to collect and acquire QoS information is needed. Even though some works [707] believe service consumers are responsible for development of QoS information, others [450, 737, 747] assume that service providers are supposed to offer QoS information along with their service descriptions.
5. **Aggregating the Evaluation Results into a Comparable Unit**: This phase can include the transferring of semantically described QoS to a proper input for an optimization problem [295]. It is essential to aggregate QoS criteria and sub-criteria scores to gain a final score for the service. In this step, a suitable aggregation method needs to be selected [785].

6.2.2 Process of Web Service Selection

After web services QoS values have been obtained, they are used as an input for a selection approach to find the most preferable web services. Most of researches suggest the following selection phases:

- Formulating and modeling the problem is done in this first phase which includes finding constraints and objectives for the selection problem. For example, in a work done by D. Tsesmetzis et al. [715] the bandwidth has been considered as a constraint and the objective is to minimize the cost.
- Next, the selection problem is tackled by a proper optimization or decision making technique which suits the modeled problem.

6.3 Taxonomy

6.3.1 Taxonomy of Web Service Selection Context

Web service selection has been investigated in different computing paradigms and most importantly for Grid, and SOA. This section shows how characteristics of the problem in each realm differ from the others significantly. Major objectives of works in each context are summarized and illustrated in Figure 6.2.

6.3.1.1 Grid Computing

Grid Computing aims to enable resource sharing and coordinated problem solving in dynamic, multi-institutional virtual organizations [279, 280]. The goal of such a paradigm is to enable federated resource sharing in dynamic and distributed environments. Therefore, QoS management and selection work in this context is mainly focused on load balancing [149] and fair distribution of resources among service requests. Some efforts [91,737] present the problem of selection as a challenge of finding proper services to form the composition which can maximize objective function.

Load Balancing Xiaoling et al. [747] mentioned that because web services are highly dynamic and distributed, it is likely that loads on service providers are not distributed symmetrically. Consequently, the work approaches the problem by applying the queuing theory. Their idea is using queuing theory to assign requests to service providers that have the least load. Hence, the system obtains all providers which can serve the request and choose the one which has the minimal expected waiting time. The Expected Waiting Time (EWT) is calculated based on the formula below:

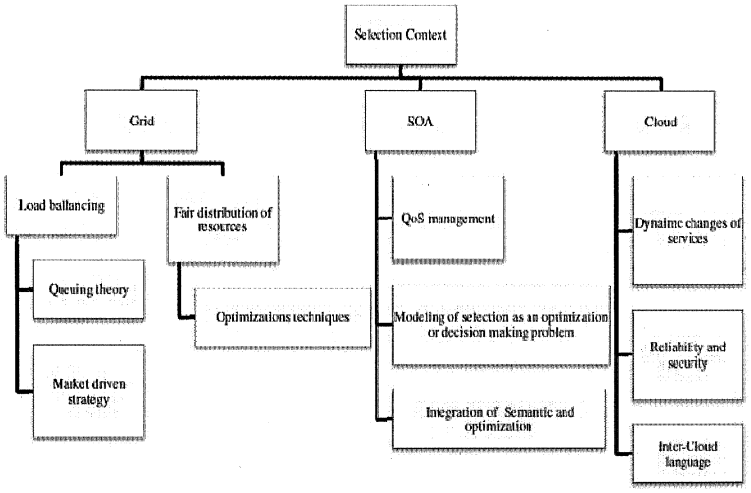

FIGURE 6.2
Selection Contexts.

$$EWT = E(w) = \frac{1}{\mu}\left(\frac{\rho}{1-\rho}\right) = \frac{\lambda}{\mu(\mu-\lambda)} \tag{6.1}$$

where λ is the arrival rate, which specifies the number of requests the provider served in the time unit. This can be obtained by:

$$\lambda = \frac{NUM}{elapstime} \tag{6.2}$$

In the equation elapstime is calculated from the point when the web service provider is started, and Num is number of requests have been satisfied during elapstime. Performance evaluation for the approach shows the selection strategy reduces total waiting time in general and works better than random selection when the provider's capacity is limited and low. Most service selection strategies only take care of requestor's interests, which brings heavy loads to best service providers, however Wancheng et al. [737] uses the price-demand rule in the commodity-market to improve load balancing. The paper discusses that using best-effort strategy to serve requesters leads to an unbalanced system because of greedy competition for the best services. So in this approach, service requesters and providers are considered as buyers and sellers in a commodity-market to adjust their supply and demand [149]. In addition, the selection strategy offers "enough" QoS as a

substitute for "best" QoS. The evaluation model for web service selection introduces the relationship between price of attributes and value of attributes instead of just simply using an additive weighting method for aggregation of QoS properties. It models the service selection as a 0–1 multi- dimensional knapsack problem. Furthermore, the enhancement to the service selection model was made by the use of the web service's price to adjust their supply and demand. The demands of web services are determined by their invocation rate. Therefore, it increases the price when the invocation rate is high and vice versa to obtain a balanced system. The work has not considered requestor's profit in the web service selection model which can be considered as its shortcoming.

Fair Distribution of Resources Tapashree Guha et al. [321] proposed an algorithm, which enables optimal selection on the Grid while considering fair distribution of resources among requestors in a condition that several clients should be persuaded concurrently. First, the paper analyzed traditional matchmaking algorithms [484]. Those algorithms take the first request and then map it to the service with the highest score. And then they select the second best for the second request in the list. The major drawback of this sort of algorithm is that requests placed later in the queue are mapped to services with poorer match scores. One way to tackle this problem is applying a constraint-satisfaction-based matchmaking algorithm (CS-MM). The algorithm effectiveness appears when several requests are assigned to a single service which can only serve one request at a time. Therefore, the algorithm selects the service for a request, which has the lower, second highest match score. Consequently, the algorithm can solve traditional matchmaking problems in a fair manner. However, to achieve better results when we are processing a large number of concurrent requests for particular services the Multiple Objective based Particle Swarm Optimization Algorithm using Crowding Distance (MOPSO-CD) was adopted. The algorithm is swarm based artificial intelligence algorithm [198, 354, 568] and was inspired from the nature of bird flocking. The experimental results in the paper show although the MOPSO-CD taking considerably more time to be executed; however, it can produce a far more accurate solution than CS-MM.

With the aim of fair distribution of Grid resources (services) among requests, Ludwig et al. [485] presented a service selection approach for Grid computing based on a Genetic Algorithm (GA) and compares its performance with traditional matchmaking methods. First, the paper proposes a scenario in which many service requests coming into the system have to be served at the same time. There are five quality of service criteria considered as: execution duration, price, reputation, reliability, and availability. Next, it compares matchmaking and genetic algorithm approaches. The matchmaking algorithm uses a weighting approach for selection and selects the service with highest score for each request. The main weakness of this method is that requests later in the list are very likely to map to services having worse

match scores. To tackle this problem the paper applies a NSGA-II genetic algorithm to find the solution for mapping the request to services in a fair manner. The genetic algorithm has been tested for several populations and the average matching score has been compared with matchmaking algorithms. Results show that if proper population size is chosen for NSGA-II approach, it can deliver better performance compared to traditional matchmaking algorithms. However, the work has solved the fairness problem by introducing a new problem of finding a proper population size for the GA.

6.3.1.2 Service Oriented Architecture

In this paradigm the main concern is QoS management (including QoS description languages for services and user preferences), definition of QoS ontology and optimization of multi-criteria web service selection. For this purpose the semantic web service has been applied by many works [295, 709] in SOA to increase expressiveness, flexibility, and accuracy by applying ontology for representing QoS properties. Semantically described QoS information then has to be converted to comparable units to be proper inputs for optimization-based solutions [295] for selection. In the following we show how challenges mentioned have been addressed.

Quality of Service Description Toma et al. [708] discuss various approaches and their advantages and disadvantages toward modeling QoS. Initially, the work starts with discussing three main approaches to processing QoS in SOA namely as: combined broker, separate QoS-broker, and direct negotiation. The combined broker approach [642] is an extended version of UDDI in which a broker extended to be able to process QoS information. In the second method [502, 740] the devoted broker is responsible for processing QoS. And the third one requires provider and requestor to agree on service level agreement [482, 712]. Subsequently, the paper talks about current supports of QoS in WSMO with the help of non-functional properties. WSMO proposes a number of non-functional properties for each element in a service description. The description of QoS is based on the Dublin Core Metadata Initiative [753]. Finally, the article offers three techniques to extend WSMO/WSML support for QoS. The first method specifies each QoS property through the use of relations in WSML. Therefore, there is no need to add separate vocabulary for QoS. The second technique is to define a new concept for QoS. And through the appropriate property, we can include relationship between non-functional properties. The third approach is to model them like capabilities in WSML. The main disadvantage of this approach is the need for extending WSML syntax. Among the three, the third approach seems to be the most suitable one, as it can provide the full support of QoS. However, this approach has to be developed from scratch to deliver clear syntax and proper ontology for reasoning.

Integration of Semantic and Optimization In addition, with the emer-

gence of semantic web services, some of the works in this context concentrate on filling the gap between semantic-based solutions and optimization-based solutions [295]. Furthermore, some efforts [91,478,737] describe the selection problem as a part of service composition problem.

J. Garcia et al. [295, 296] present a framework to transfer the user preferences which are in the form of semantic web services into an optimization problem which aims to select the best web service. The paper argues that, semantic web services define ontology which helps web services to be discovered and composed automatically. As they are mostly using description logic for those purposes, they are infertile in dealing with QoS based selection. The reason is QoS based selection approaches lead to optimization problems which cannot be solved by applying description logic. Authors build their selection on their previous efforts [294] which apply semantically described utility function to define user requirements. They use the work to offer a selection approach that transforms user preferences into a optimization problem. That optimization problem can be tackled using various techniques, like constraint programming or dynamic programming. The selection approach consists of the following phases. In the first phase, QoS values are retrieved from the semantic description of web services. Proceeding to the next stage, the pervious phase results are linked to the user preferences which express utility functions for related QoS criteria. In the last stage, an XSL transformation has been applied to the mentioned utility function to acquire the specification of desired optimization problem. The optimization problem chosen for this work is the Constraint Satisfaction Optimization Problem (CSOP) [621]. Moreover, any other optimization techniques can be supported by designing a proper XSL style sheet. And then web services are ranked and selected using CSOP.

6.3.1.3 Cloud Computing

Rajkumar Buyya and colleagues' definition of the Cloud highlighted main aspects of the Cloud, namely as dynamic scalability, and SLA negotiability. According to their definition [152] "A Cloud is a type of parallel and distributed system consisting of a collection of interconnected and virtualized computers that are dynamically provisioned and presented as one or more unified computing resource(s) based on service-level agreements established through negotiation between the service provider and consumers." Therefore, in a Cloud environment, users' preferences are changing dynamically [281], therefore selection strategy has to match them dynamically to proper services. At the same time services in the Cloud also change dynamically, for example based on the market circumstances the price of services may jump up as demand for the service is growing.
Even though there are a few works [337,793] which presented service selection in Cloud context, however, to the best of our knowledge they are mostly in the early stage and not considered Cloud specific characteristics and QoS di-

mensions. In the context of a Cloud, selection objective is not fair distribution of resources between requestors. That is because in a Cloud resources can be considered as infinite [514]. Instead, the Cloud environment has emphasized QoS dimensions such as reliability, cost (including data transfer cost), and security [281]. Cloud customers are typically concerned about the reliability of Cloud providers' operations. The reason is by migrating to a Cloud, they move their information and services out of their direct control [399].
In the Cloud environment, as SLA plays an important role, reliability can be measured based on SLA. Consequently, SLA has to contain a set of goals (for example certain network and system performance), which determines acceptable performance of services. By monitoring of SLA we can find out to which extent the provider has deviated from promised SLA for each criteria and then calculate reliability. Therefore, monitoring services in the Cloud are crucial for determining QoS criteria such as reliability.
Moreover, in a heterogeneous environment such as a Cloud, it is difficult to enforce syntax and semantics of QoS description of service. Therefore, the first step towards QoS modeling in a Cloud would be building an inter-Cloud language. Furthermore, service selection in a Cloud has to be done based on requestor context and with the consideration of all roles (such as user, service creator, and provider) and their objectives.
The rest of the chapter aims to survey various strategies that have been taken for QoS modeling and service selection in detail, and shows which one can be applied to tackle Cloud computing issues.

6.3.2 Web Service QoS Modeling Taxonomy

Different approaches have been offered in literature to respond to requirements explained in each phase in Section 1.2.1. In fact, selected works can be classified based on how they tackle those issues. For example, currently there are three ways to define QoS attributes for web services namely, extended Universal Discovery, Description and Integration (UDDI), and semantic web services. However, UDDI and Web Services Description Language(WSDL) [442] cannot support modeling of QoS properties of web services; therefore, works [122,598] such as UDDIe [642] and web service level agreement [483] were proposed to enrich web services with QoS properties. In following subsections more classifications for QoS management is given and their summary illustrated in Figure 6.3.

6.3.2.1 User Preferences and QoS Criteria Relation and Tendency Modeling

When service requestors express their expectations from services, they identify functional and non-functional (QoS) characteristics of required services. In addition, they have to identify which of the QoS criteria are more important than the others. A simple way to do this is to ask requestors to give weight to

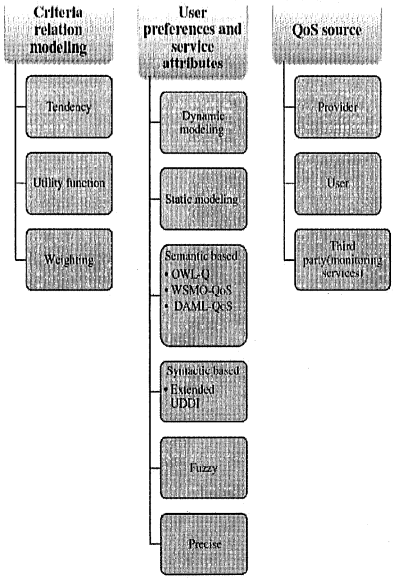

FIGURE 6.3
Web Service QoS Modeling Taxonomy.

each criterion. Using weights to achieve a decision matrix is one of the primary ways of modeling importance of criteria in user preferences. This approach has been applied by many works [264, 498, 746] as it is simple and computationally efficient. However, the major drawback concerning this approach is the complexity of finding proper weight coefficients in real world application. Furthermore, requestors' QoS preferences in terms of tendencies [746] have to be considered. For example, it has to be defined whether a parameter value is more desirable for a particular user when it is smaller or greater. There can be some other tendencies assigned to Qos properties like "exact" and "close," as presented by Vuong Xuan Tan [713].

Moreover, works in this area have focused on modeling relations between QoS criteria. For example, Tsesmetzis et al. [715] discussed the importance of QoS consideration for service providers and requestors which can help providers attract more customers and make them able to increase their infrastructure utilization. Moreover, the work creates a QoS vocabulary ontology in OWL [642] with the maximum height of two, to reduce the complexity. The QoS vocabulary covers a wide range of non-functional properties from performance to security and reliability. It also applies a standard generic model [564] for

defining association between QoS attributes and the approach for measuring them.

A comprehensive work on the relation modeling is done by Hong Qing [784]. In the work a mechanism for ranking of web services using logic scoring preferences (LSP) [244] and ordered weighted averaging (OWA) [162,265] is offered. The work has found out that current ranking algorithms are ignoring relations between individual criteria and the simple arithmetic metric they used is incapable of representing logic relations such as simultaneity and replaceability. The work claims those drawbacks can be addressed by adopting LSP and OWA. The work makes use of LSP, which was originally developed for solving hardware selection, and considers the relation between criteria of selection such as reparability, simultaneity, and mandatory-ness. However, since the work is based on LSP, it cannot deal with selection problems with many QoS criteria.

Utility function recently has been used [296,445,450,621] and it is said [296] to be the most appropriate way to model user preferences expressively. Therefore, work in a Cloud also can adopt it for Service selection. Utility function is a normalized function and shows which values of QoS criteria are more preferable. For the selection of the best alternative, all of the utility functions — each of them belongs to a QoS attribute — have to be aggregated to compute the global utility value. J. Garcia et al. [296] offer a new approach of ranking based on the description of user preferences in the form of a utility function. The work presents novel hybrid architecture for service ranking integrated in the Web Service Modeling Ontology (WSMO) [224,615]. Therefore, user requirements and preferences are modeled using the Web Service Modeling Language (WSML), adding support to utility functions. When we deal with several non-functional properties, each utility function has to be associated with a relative weight. Consequently, to solve a multi-criteria ranking problem, the user preference value calculated as weighted composition of the associated utility function values. User preferences definitions are inserted as part of a goal in the form of WSML. Figure 6.4 shows a sample of a utility function extracted from their work. As depicted by the Figure 6.4 highest utility value is reflected by that function if the price is lower than 60 and it is falling down when the price increases.

In addition to those methods, Tran et al. [713] adopted (Analytic Hierarchy Process) AHP [626] for QoS criteria relation modeling and service selection. AHP consists of three main phases: problem breakdown, comparative evaluation, and priority composition. In the first phase, each problem is broken done to three elements — overall goal, its criteria and sub-criteria. Next, pairwise comparisons for all criteria and sub-criteria will be done to obtain their relative importance for decision making and subsequently all solutions will be ranked locally applying those sub-criteria. In final stage, all relative local ranks of solutions will be combined to obtain the overall rank for the solution. Since the method is using pairwise ranking its performance decreases when the selection problem consists of large number of criteria. Moreover, it increases flexibility

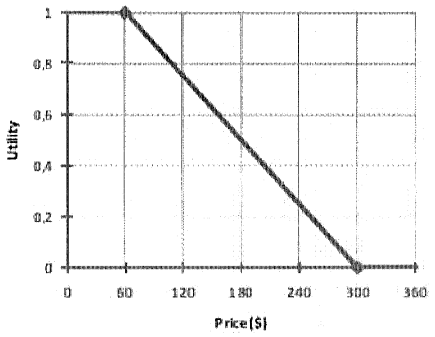

FIGURE 6.4
Price Utility Function [296]

of the ranking algorithm by allowing two options of mandatory and optional QoS constraints. It offers more options (exact and close) for tendency characteristics of QoS criteria in addition to negative and positive options offered by other works [296,709]. The exact option shows that the property value should be equal to the defined value by request. And the close option says that closer value to the requested value is more desirable than the others.
Furthermore, Wang et al. [745] presents a novel resolution process for determining the linguistic weight of QoS criteria. First, it creates the framework for evaluation QoS criteria, which summarizes QoS requirements of web services from decision makers' opinions and market surveys. Next, it determines the importance of criteria by aggregating all groups of participants' opinions. It can be said that the approach is a complement to the works [222,686] which helps to select web services in market places based on QoS.

6.3.2.2 QoS Source: Provider, User, or Third Party

The majority of works made an assumption that QoS information has been supplied by providers along with services description. However, some believe [745] that not all provides are willing to provide the related QoS information for comparison, or that they are likely to advertise their services exaggeratedly. And that's why we have to consider consumers' feedback on their experience of using web services to determine QoS values. In addition, there are values that cannot be determined by users or providers such as network related QoS information, reliability, and trust, which is usually evaluated by a third party (monitoring services). To collect QoS information Zhenyu et al. [478] built an approach based on distributed agents. The main contribution of the work is building a procedure for processing the quality of web service from multiple locations. The article presents a distributed approach to acquire the QoS data from users in different locations by the help of agents scattered in the network.

6.3.2.3 Context-Aware

There are cases when QoS information of web services vary based on the requestors' context. The context-aware QoS information let service providers or third parties offer QoS information for web service based on the service requestor context. Hong Qing et al. [784] adopts an in-context project [12,13] for providing dynamic context information. This information includes location, budget of users, and availability of services. Therefore, the web service selection is based on reasoning based on the context data. In addition, a selection approach offered by Lamparter et al. [450] is context sensitive as it adopts utility function policies to model context dependent user preferences. For example, there might be a case where web service which is selected as the best service in the list is not available in a particular location; therefore selecting it for the requestor in that context is not acceptable.
In Cloud commuting service selection, context information also plays a key role. The first is context of users and providers can affect performance, client location information can be used by selection as the basis for determining which data center location is closest and thus can provide service with less latency and perhaps higher throughput. There are some restrictions applied by law for deploying on Clouds in particulate Geographic locations. For example, according to DPA, Clouds located outside the European Union are unlawful. As Papakos et al. [565] discussed, another important QoS information (for service selection in Cloud) to be considered is the user's device. As they have mentioned, requirements of a client with a mobile device can change because of changes in the context of the device. These status changes encompass hardware resources, environmental variables, or user preferences. Binding to a service offering different quality of service levels from the ones required may lead to excess consumption of mobile resources such as battery life, as well as unnecessarily high provision costs. Therefore, the paper proposes VOLARE middleware that dynamically selects Cloud service requests.

6.3.2.4 Dynamic Versus Static Modeling of User Preferences and Web Service QoS Attributes

The value of QoS property for a web service can remain constant and therefore identified once, or it can be updated regularly. On the other hand, user preferences also can be specified once or can change during interaction with the system. For example, a service response time can rely heavily on network traffic. Therefore, it can have a short response time at a moment in a day and then can increase dynamically to a certain level when it is not any longer available.
Lampar [450] argued the shortcomings of current works like the WS-agreement [88] in modeling of users' dynamic preferences on service configuration. And then it shows how those could be tackled by a mechanism which efficiently declares non-functional attributes such as price. In order to succeed in that, the service configuration is modeled using utility function policies [426,451]

which are presented in the form of OWL ontology. Another achievement of this work is finding out that performance crucially depends on the way web service offers and requests are modeled. Consequently, if service selection is runtime, one way to increase the performance would be limiting the expressiveness of biding language.
In context of the Cloud, Andrzej et al. [793] have taken the advantage of state-full web services to deal with dynamic changes of Cloud services. They proposed a higher layer of abstraction which provides selection based on QoS criteria values that describe dynamically the state and characteristics of Cloud Services (cluster as a service). In addition, they have implemented their proposed solution, and through several tests, it was proved that the proposed technology is feasible.

6.3.2.5 Semantic-based Versus Syntactic-based

There are two ways to describe entities of a web service, namely as semantic-based or syntactic-based (extended UDDI). The syntactic-based approach uses numerical and key word values for web service QoS properties. Although an extended version of UDDI can encompass the QoS information of web services, it is not machine understandable, hence not suitable for automatic selection. The automation selection enables service provider and requestor to be decoupled, which means they don't have to be aware of each other before execution phase. In addition, different service providers and users can apply a variety of models for describing QoS attributes, and then it is essential to acquire a solution to understand different QoS representations. That solution which covers mentioned drawbacks is semantic web service which can increase expressiveness, flexibility, and accuracy by applying ontology for representing QoS properties. A number of QoS ontologies have been proposed for web service which mainly developed based on three semantic languages, namely as OWL-Q [445, 629], WSMO-QoS [503, 709, 746], and DAML-QoS [798].
OWL-Q is built on OWL-S [498] language and has the strength of not only modeling and measuring static QoS attributes but also dynamic properties. Nonetheless, it doesn't offer a way of defining QoS criteria importance and whether they are mandatory or optional. WSMO-QoS is yet another promising approach for QoS modeling of web services. It offers an upper level ontology which can describe each attribute of a web service in detail. Moreover, a new category of properties with the name of non-functional has been added to Goal and semantic service description. The new category provides attributes for describing QoS type, metric, dynamicity level, tendency, and importance. Tran et al. [713] offers comprehensive OWL-based QoS ontology (include diverse data type) and an AHP-based ranking algorithm to dynamically rank services at different levels of provider and consumer requirements.
Finally, DAML-QoS developed as a complement to DAML-S to be capable of specifying QoS ontology. More specifically, the ontology consists of three layers — QoS profile, QoS property, and QoS metric. The QoS-profile layer is

developed mainly to provide advertisement, and request ontology for providers and requestors. The QoS property deals with name, and domain, and range of QoS attributes can be defined in DAML-QoS. And finally, the QoS metric layer is offered for measuring QoS metrics and computing their values.
In Cloud computing environment semantic of services and data can be a first step towards building an inter-Cloud language. As we have discussed in our previous work [220] in a heterogeneous environment such as Cloud, it is difficult to enforce syntax and semantics of service description (including QoS description) and users' requirements. Therefore, applying symmetric attribute-based matching between requirements and request is impossible. To tackle that we proposed an advertisement approach for Cloud providers by modeling Cloud services into WSMO. Then we offered ontology-based matchmaking to help users deploy their application on the most proper IaaS providers based on their QoS preferences when both sides are not using the same notation to describe their services and requirements.

6.3.2.6 Fuzzy Versus Precise Preferences

As Wang et al. [745] believe, the complex nature of QoS made the service consumers' perception fuzzy. Therefore, it proposes new selection algorithms based on MaX-Min-Max composition of Intuitionistic Fuzzy Set (IFS) under the vague information. The work expects fuzzy perception of service consumers and providers. Therefore, it suggests a fuzzy multi-criteria decision making solution with following aspects:

- Capable of handling imprecise preferences of service consumers.
- A clear weighting strategy for QoS criteria.
- A QoS aware service ranking ability.

The article uses the QoS criteria presented by the W3C working group in 2003. In addition, it mentions the fact that not all providers are willing to provide the related the QoS information for comparison. And that is why we have to consider works that use consumers' feedback on their experience of using web services. As mentioned earlier, the web service selection approach in this work is based on IFS. IFS was introduced in 1989 by Atanassov [101] and it can be considered as a generalization of the concept of fuzzy set which is effective in dealing with vagueness, in addition, this type of selection problem can be modeled by Fuzzy multi-objective and multi-level Optimization [801].

6.3.2.7 Identifying Roles in the Problem

It is important to determine selection algorithm goal. Is it working to maximize provider profit, or takeing care of requestors' interests or both. There are some works [713] which are flexible enough to act for both parties. In details, they give weight to objectives of requestors and providers in the utility function.

For example, by giving more weight to providers in the utility function they are working in favor of providers. Nevertheless, before this step all the roles in the problem and their objectives have to be clearly determined and then a decision has to be made on the importance of each role goal. A summary of surveyed approaches in different context and their applicability to Cloud Computing is shown in Table 6.1.

TABLE 6.1
Selection Works in Different Contexts and Their Applicability to Cloud Computing

Criteria	Grid	SOA	Cloud
Semantic based		[295, 708, 709]	Yes
Load balancing		[737, 747]	No
Fair distribution of resources	[321, 485]	[747]	No
maximize user or provider profit	[149, 715]	[450, 737]	No
Integration of semantic and optimization		[295, 715]	Yes
Considering reliability		[383]	Yes
Provider as QoS source	[295, 708, 709]		No
Consumers as QoS source		[450, 707]	No
Third Party as QoS source(monitoring service)		[220]	Yes
Selection in Service Composition contex		[91, 737]	Yes
Utility based function for preferences		[295, 296, 450]	Yes
Context aware		[450, 478, 565, 784]	Yes
Dynamic modeling	[450, 793]		Yes

6.3.3 Taxonomy of Web Service Selection Approach

As it is illustrated by Figure 6.5, selection work mainly has utilized two types of approach for web service selection, namely as optimization and decision making. Decision-making can be defined as process of identifying and choosing alternatives based on values and goals of decision makers. Therefore, the assumption is that, there are many candidates to be chosen and the aim is to select the one that best fits our goal, desires, and constraints. This process can be depicted as Figure 6.6. When there is one single criterion, then selection can be made by identifying the alternatives with the best value of that criterion. However, when there are several criteria as it is depicted in Figure 6.6, we have to know which criteria have higher priorities to users. And this is where the decision making techniques offer approaches such as AHP to help users

assign the comparative importance to those criteria. In the case of multiple criteria and a small number of explicitly given alternatives, and when there is no existing scale of measurement for the criteria, the problem can be solved using decision making approaches such as the Analytical Hierarchy process (AHP) and the multi-attribute utility theory (MAUT).

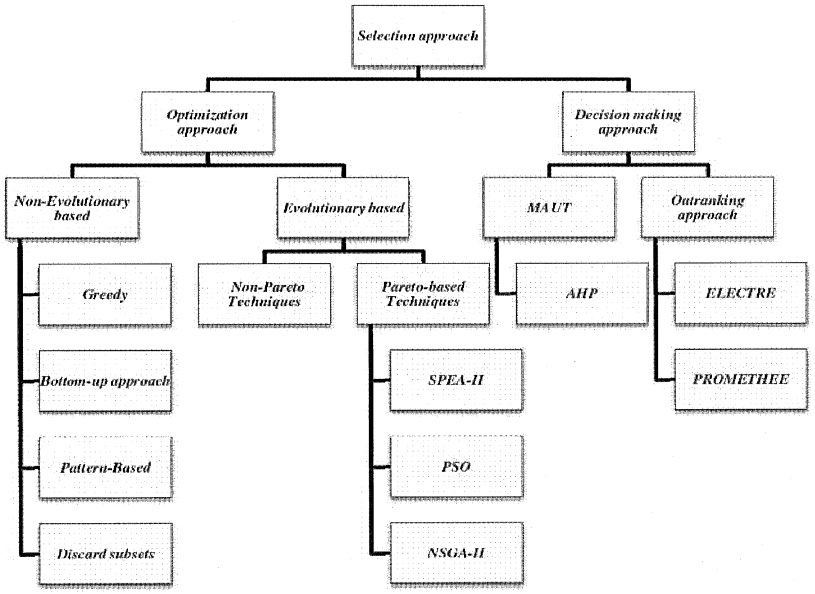

FIGURE 6.5
Web Service Selection Approaches.

However, if given alternatives are many or implicit, multiple criteria optimization techniques can be applied. These techniques can be categorized into evolutionary-based and non-evolutionary based techniques. Evolutionary-based techniques (for multi-objective problem) are based on the Pareto solution, which is an economic concept and applied for the condition when a better value for an attribute can only happen once the value of at least one other attribute gets worse. AI solutions, such as the Non-Dominated Sorting Genetic Algorithm-II (NSGA-II), Strength Pareto Evolutionary Approach 2 (SPEA-2), and particle swarm optimization (PSO) are among the most scientifically used techniques in this area. In the following each of these selection approaches (from decision making to optimization) will be discussed in detail.

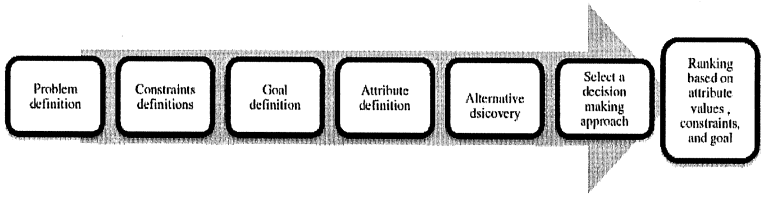

FIGURE 6.6
Process of Decision Making

6.3.3.1 Decision Making Formulation and Analytical Hierarchy Process(AHP)

The problem of multi-criteria decision making(MCDM) is a favorite method for expressing decision making problems which can be considered in the following form: C1,...,Cm as criteria, A1,Ĕ,An as alternatives, W1,Ĕ,Wm which are weights assigned to criteria, a Matrix which its aij element shows the score of alternative Aj against criteria Ci, and Xi,Ĕ, Xn are aggregative score of alternative Ai. MCDM approaches can be classified into two main categories namely, Multi-Attribute Utility Theory (MAUT) and outranking approach. MAUT is benefited from applying utility function which has to be maximized. Moreover, it allows the complete payoff between criteria that can show relative importance of each attribute in Alternatives evaluation. In literature, the Analytical Hierarchy process (AHP) is one of the most applied methods in the MAUT category, therefore in this section we will investigate it more. The AHP method was suggested by Satty in 1998 and is based on a pairwise comparison of criteria to determine their weights in utility function. From other work in this category, a distance-minimizing approach [299] can be named.
The major contribution of AHP is to convert subjective assessments of relative importance to numerical values or weights. The methodology of AHP is based on pairwise comparisons of the criteria by asking the question of "How important is criterion Ci compare to criterion Cj?" The answer to this question determines weights for criteria. Figure 19.7 depicts the process of choosing the best service provider when there are four criteria (cost reliability, security, and trust) to be considered. The answers to questions can be one of the following: After the pairwise comparison as it can be seen from Figure 19.7, relative importance has been given to each criterion. In next step, and similar questions have to be asked to evaluate the performance scores for alternatives on the subjective criteria. And then based on the result of this phase we can rank alternatives and select the one with the highest rank.
The next category in decision making approaches is named Outranking which was introduced by Roy in 1968, and the idea behind it can be simply defined as follows. In this approach all alternatives are compared in a way that alternative Ai can outrank Aj if on the majority of criteria Ai performs as well as Aj,

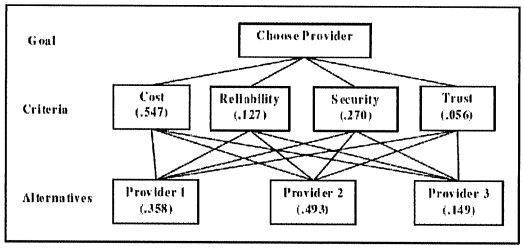

FIGURE 6.7
Choosing a Provider Using AHP.

TABLE 6.2
Major Scale of Pairwise Comparisons

Scores	Response to the question
1	Equal importance or preference.
3	Moderate importance or preference of one over another.
5	Strong or essential importance or preference.
7	Very strong or demonstrated importance or preference.
9	Extreme importance or preference.

at the same time as it can achieve an sufficiently acceptable score in other criteria. ELECTRE [272] and PROMETHEE [138] are among the most famous approaches in this category.

6.3.3.2 Optimization Methods

Optimization methods seek the most suitable service, which is usually the one which maximizes or minimizes one or several criteria, such as cost, deployment time, etc. The problem can be complicated when we consider more than one criterion. Furthermore, there are some constraints in selection problems which can be imposed by service requestors or consumers. By considering constraints in selection, the definition of optimization can be rewritten as "finding of the most suitable services for the clients or providers, which maximizes or minimizes one or several criteria and still adhers to the constraints." For example, assume that the best service is the one with minimum cost, highest availability, and least deployment time when there is a limitation for providers to serve the requestor [715,737] due to bandwidth limitation. Then, the problem of selection can be formulated as the "Selective Multiple Choice Knapsack Problem" (SMCKP) which is kind of NP-hard problem. The works in areas of selection has faced the problem in different ways, some tries to find the optimal solution [582], others aimed at finding the semi-optimal solution by offering suitable heuristics [91,485,512,745]. Dominant approaches in this area

can be classified in two main categories — non-evolutionary and evolutionary optimization methods — which will be investigated in the following.

Non-Evolutionary Optimization Method Four classes of selection approach in this category are: pattern based, selection using discarded subset results, bottom-up selection, and the greedy. This classification is done by Jaeger et al. [383]. The objective is to solve the problem of multi-criteria selection. It compares several algorithms for selecting the best web service candidates. They have considered four QoS categories introduced by Zeng et al. and Menasce [511,792]. The categories are namely: execution time, cost, reputation and availability. After that, an approach for aggregating QoS [385] of individual web services was applied. At the comparison stage the work applied the Simple Additive Weighting (SAW) approach which was extracted from the context of Multiple Criteria Decision making (MCDM) [358]. They have compared algorithms' performances, and the results are reported as follows:

- A greedy selection is not able to consider constraints. Instead it can find the candidate that scored the highest among all the other candidates.
- Bottom-up approach [384] relies on a fact that the selection problem for composition (selection in composition) of web services shows similarities to Resource Constrained Project Scheduling Problem (RCPSP). In RCSP, project is divided into individual tasks, and then each task has to be assigned to an available worker to complete the whole project in a way that meets the constraints, such as time. Bottom-up selection results in the second worse QoS; however. the computation effort is negligible.
- Pattern-Based selection [316] which considers each composition pattern separately and then tries to find the best assignment. This approach offers the best achieved QoS compared to all heuristic approaches. The computational effort of this selection is reasonable, and depends on the composition structure.
- The selection by discarding subsets is a kind of backtracking-based algorithm which uses a search tree consisting of nodes, each representing a possible pair of a candidate and task. It results in the best QoS possible and also meets the constraints.

Evolutionary Multi-objective Optimization Method Evolutionary Multi-objective Optimization methods are based on the principle of natural selection, which is also called the survival of the fittest and originally characterized by Charles Darwin [219]. Since evolutionary approaches have shown desirable potential for solving optimization problems, this class of search strategy has been utilized for multi-objective optimization from mid-1980s. Evolutionary multi-objective optimization is in fact a combination of the evolutionary computation and traditional multiple criteria decision making.

Evolutionary approaches follow two major concepts. The first concept is the competition for reproduction which is called selection. And second one mimics the ability of producing new generation by mutation, which is also called variation.

A considerable number of evolutionary multi-objective optimization (EMOO) techniques have been developed in recent years [197, 719]. In an attempt to discuss the most important approaches proposed, we decided to classify these techniques using the following scheme of Non-Pareto Techniques and Pareto-based Techniques such as NSGA, NPGA, three of the most applied methods in the web selection literature as NSGA, SPEA, and PSO are briefly explained.

NSGA Non-dominated Sorting Genetic Algorithm (NSGA) [675] was proposed by Srinivas and Deb. The algorithm modifies the ranking procedure originally proposed by Goldberg [309] based on several layers of classifications of the individuals. NSGA was highly computational intensive and had several other drawbacks which led to the rise of [227] NSGA-II. In the first step NSGA-II constructs a space of solutions, then performs sorting based on non-domination level, applies a crowded-comparison operator to create a new pool of offspring. As it applies a fast non-dominated sorting approach which has O(MN2) computational complexity, M is the number of objectives and N the population size. The algorithm is capable of outperforming many other genetic optimization algorithms [227]. That is mainly because it applies the crowded-comparison operator.

SPEA The Strength Pareto Evolutionary Algorithm (SPEA) is presented by Zitzler and Thiele [802]. The method is a result of integrating different EMOO techniques. It has the unique character of archiving non-dominated solutions already found in order not to lose certain portions of the current non-dominated front due to random effects. For ranking (calculating the strength) of non-dominated solutions an approach similar to MOGA has been used. In addition, fitness of individuals is calculated based on the strengths of all non-dominated solutions in archive which can dominate it. Moreover, for maintaining diversity a method called the "average linkage method" [523] has been used. The 0/1 knapsack problem [582, 802] is one of the main fields SPEA has been applied to.

PSO The particle swarm optimization (PSO) [424] was built by Kennedy and Beernaert and inspired by the flocking and swarm behavior of birds and insects. Every solution in PSO is represented by a particle. In the first phase a number of particles are generated with random positions and velocities. In the next step, each particle flies through the search space with the velocity constantly updated based on two important factors, first by its best position, and second by the position of the best particle in a problem space which corresponds to the cooperative effect of the particles in optimization searching. Therefore it guides particles toward the global best position found

so far. On the other hand, in order to control the ability of the particles to search and be restricted within the search space boundary, a maximum velocity vector Vmax was introduced. The PSO algorithms are mainly used in scheduling problems and New PSO alternatives are constantly being developed to improve scheduling performance [321].

6.4 Future Directions and Conclusion

This chapter introduced a general model for QoS-aware service selection which consists of QoS management, and service selection. Furthermore, selection works in different context of Cloud, Grid, and SOA have been reviewed in order to find out what could be inherited from works in other paradigms and what has to be done uniquely (considering special characteristics of Cloud Computing) for Cloud Computing. Below, a summary of future directions is given:

- Cloud services have specific characteristics and QoS dimensions which have to be defined, and then approaches for measuring those QoS criteria (reliability, security, and trust) have to be discovered. For example, methods to evaluate reliability and trust of providers from user feedback can be further studied.
- In modeling user preferences, there is opportunity to explore how to capture relative importance of criteria from requestors or other decision makers in selection problems.
- In addition, when selection aim is to find the best Individual services to form a composition, the general utility function which is given for a whole composition can be decomposed to provide a utility function for each individual service in the composition. Techniques for the decomposition of utility functions in this area can be investigated.
- In Cloud computing, dynamic modeling of service status and user demand and preferences is an essential task which can be further enhanced.
- In addition, work can focus on identifying roles and parties in the decision making process to ensure success of selection in maximizing all engaging parties' profits.
- Building an inter-Cloud language using Semantic web service for modeling services and data is another promising field to be considered.
- Moreover, researchers can investigate and identify Cloud users' context attributes (users' device characteristics and location) and study their effects on the performance of Cloud service deployment.