Service Recommendation and Selection in Centralized and Decentralized Environments

Mariwan Hassan Ahmed

A Thesis Submitted in Fulfilment of the Requirements for the Degree of

Doctor of Philosophy

University of Derby

School of Engineering and Technology

2016

Contents

List of Figures

List of Tables

List of Publications

Conferences

1- M. Ahmed, L. Liu, J. Hardy and B. Yuan, "An Efficient Algorithm for Partially Matched Web Services Based on Consumer's QoS Requirements," *Utility and Cloud Computing (UCC), 2014 IEEE/ACM 7th International Conference on*, London, 2014, pp. 859-864.

2- M. Ahmed, L. Liu, B. Yuan, M. Trovati and J. Hardy, "Context-Aware Service Discovery and Selection in Decentralized Environments," *Pervasive Intelligence and Computing (PICOM), 2015 IEEE International Conference on*, Liverpool, 2015, pp. 2224-2231. **(Best Paper Award)**

Journals

1- M. Ahmed, L. Liu, J. Hardy and B. Yuan, and Nick Antonopoulos," An Efficient Algorithm for Partially Matched Services in Internet of Services," *Personal and Ubiquitous Computing*, 2016, 20, 3, pp. 283—293.

Abstract

With the increasing use of web services in everyday tasks we are entering an era of Internet of Services (IoS). Service discovery and selection in both centralized and decentralized environments have become a critical issue in the area of web services, in particular when services having similar functionality but different Quality of Service (QoS). As a result, selecting a high quality service that best suits consumer requirements from a large list of functionally equivalent services is a challenging task. In response to increasing numbers of services in the discovery and selection process, there is a corresponding increase of service consumers and a consequent diversity in Quality of Service (QoS) available. Increases in both sides leads to a diversity in the demand and supply of services, which would result in the partial match of the requirements and offers. Furthermore, it is challenging for customers to select suitable services from a large number of services that satisfy consumer functional requirements. Therefore, web service recommendation becomes an attractive solution to provide recommended services to consumers which can satisfy their requirements.

In this thesis, first a service ranking and selection algorithm is proposed by considering multiple QoS requirements and allowing partially matched services to be counted as a candidate for the selection process. With the initial list of available services the approach considers those services with a partial match of consumer requirements and ranks them based on the QoS parameters, this allows the consumer to select suitable service. In addition, providing weight value for QoS parameters might not be an easy and understandable task for consumers, as a result an automatic weight calculation method has been included for consumer requirements by utilizing distance correlation between QoS parameters.

The second aspect of the work in the thesis is the process of QoS based web service recommendation. With an increasing number of web services having similar functionality, it is challenging for service consumers to find out suitable web services that meet their requirements. We propose a personalised service recommendation method using the LDA topic model, which extracts latent interests of consumers and latent topics of services in the form of probability distribution. In addition, the proposed method is able to improve the accuracy of prediction of QoS properties by considering the correlation between neighbouring services and return a list of recommended services that best satisfy consumer requirements.

The third part of the thesis concerns providing service discovery and selection in a decentralized environment. Service discovery approaches are often supported by centralized repositories that could suffer from single point failure, performance bottleneck, and scalability issues in large scale systems. To address these issues, we propose a context-aware service discovery and selection approach in a decentralized peer-to-peer environment. In the approach homophily similarity was used for bootstrapping and distribution of nodes. The discovery process is based on the similarity of nodes and previous interaction and behaviour of the nodes, which will help the discovery process in a dynamic environment. Our approach is not only considering service discovery, but also the selection of suitable web service by taking into account the QoS properties of the web services.

The major contribution of the thesis is providing a comprehensive QoS based service recommendation and selection in centralized and decentralized environments. With the proposed approach consumers will be able to select suitable service based on their requirements. Experimental results on real world service datasets showed that proposed approaches achieved better performance and efficiency in recommendation and selection process.

Declaration

The study outline in the dissertation was carried out in the School of Engineering and Technology at The University of Derby, under supervision of Professor Lu Liu. This is to declare that the work stated in this thesis was done by the author, unless otherwise is indicated, and no part of the thesis has been submitted in a thesis form to any other university or similar institution.

Mariwan Ahmed

University of Derby

Derby, August 2016

Acknowledgement

It is obvious through the help and support of others you can reach any goal in life. Thankfully I am blessed with many helpful and supportive people around me. Firstly, I want to start with my supervisor Dr Lu Liu, one of the most supportive and helpful supervisor I have come across in my academic life, this thesis would never been possible without his help and support. Thank you very much for your enormous advice and patience during my PhD. In addition, I would like to thank all colleagues and staff at the department and university of Derby, James Hardy and Bo Yuan in particular. A special thanks for my beloved wife Shena for her encouragement and support at all times, my mum and dad, and all my brothers and sisters, family and friends.

To my lovely wife and beloved parents

Chapter 1 Introduction

1.1 Overview

Web services are self-contained and self-describing computational Web components designed to support machine-to-machine interaction by programmatic Web method calls [1]. Services are platform independent entities that can be used in atomic form or in composite forms, where services offer integrated functionality by combining with other services. Web services, understood as a means for developing highly complex systems by using loosely coupled services, are one of the main implementations of a service oriented architecture (SOA).

With the rapid development of distributed systems, web services are widely accepted in academic and industry communities. They are understood as a means for developing high complexity systems by using loosely-coupled services. Web service discovery is the process of finding suitable services that match consumer requirements from a list of available web services published in a service registry. From the ever-increasing number of services, it has become a trend that the data oriented web is moving towards a service oriented web [2]. As a result, finding a desired web service is similar to looking for a needle in a haystack [3]. Efficiently and effectively discovering and selecting web services that match consumer requirements becomes a critical issue, especially when there is a pool of functionally equivalent services with different quality of service (QoS).

Consumers are concerned about the QoS parameters in addition to the functional properties of the services. Different consumers prefer different QoS parameters, or

different ways of expressing QoS parameters. This diversity leads to partial or imprecise matching of QoS parameters. Services may not be selected because of the existence of partial matching of some properties even though the services are relevant to the user's query. For example, a potential passenger may have several criteria for a flight service, and an airline operator may have a list of relevant offers. Not having an exact match for all the criteria the user is asking for leads to these offers being excluded from the selection process. For an accurate and efficient service selection, there should be a mechanism to deal with partial matching during service selection. This leads to the necessity for an accurate, efficient and dynamic service selection method using a partial match between consumer requirements and provider offers.

In addition, increases in the number of web services with different QoS makes it challenging for service consumers to decide on and select the best available web service for their requirements. Service recommendation is one of the approaches that help consumers by providing a list of recommended services that match their requirements. By using recommendation systems, and based on previous interactions and behaviours, consumers can be provided with a list of recommended web services. While there are a significant number of works on recommendation systems using collaborative filtering (CF) [4] [5] [6] [7] [8] and content-based filtering [9] [10], most of the approaches ignore the latent interests of services and consumers and correlations with neighbouring web services. To improve the recommendation process, a hybrid approach is required. Such an approach would use topic modelling as an efficient dimension reduction technique to extract the latent interests of providers and consumers. Employing probability distributions and using correlations among neighbouring web services would improve the accuracy of prediction of QoS

properties of web services and return a list of recommended web services to the consumer.

Last but not least, the cloud computing trend, which is a service-oriented application, has resulted in a growing number of web services such as Software as a Service (SaaS), platform as a Service (PaaS), and Infrastructure as a Service (IaaS). The increasing number of web services has led to critical issues like efficiency and scalability. Most of the current approaches focus on centralized architectures in which a single service registry or multiple service registries synchronize together. These approaches lead to bottlenecks, single points of failure, and scalability limitations. In addition, one of the main drawbacks of centralized approaches is having a global knowledge about the system that is not presented in decentralized environments. While availability and demand for services is growing significantly, the inherent disadvantages in centralized environments prevent web services from being applied in large scale service networks. As Service Oriented Computing environments are largely distributed, a decentralized approach appears to be a suitable way to address the issues and achieve scalable, reliable and robust service discovery and selection.

The term *services* as it is used throughout this thesis is a general term for web services in service oriented computing, as well as other forms of services, such as services in the Internet of Things (IoT) [11]. IoT connects billions of devices in an internet-like structure. Each device is encapsulated as a real-world service that provides functionality and information exchanges with other devices. In this thesis, three approaches are used to tackle these challenging research problems. The first approach is directed to web service partial matching by proposing an efficient ranking and

selection algorithm for partially matched services in the Internet of Services (IoS). The next approach addresses service recommendations by proposing hybrid service recommendation using correlation-based and semantic content features. The last approach focuses on web service discovery and selection in a decentralized environment by proposing a context-aware service discovery and selection algorithm in decentralized environments. The detailed motivations, contributions, and organizations of these approaches are presented in section 1.2, section 1.3, and section 1.4 respectively.

1.2 Research Motivations

Services have become a persistent fact in the field of computing and seemingly enter every aspect of life. SOA, Grid, and cloud computing are different computing paradigms based on web services. With an increase in the number of web services, the requirements of consumers increase seamlessly.

Each service can be described by its functional and non-functional properties. The functional properties of a service to the consumer can be represented by its inputs and outputs and by the behaviour of the service; it describes what the service does. However, the non-functional properties of a service show the details of how well a service can provide that functionality. Examples of those properties are response time, availability, throughput, latency, and so on. Commonly, QoS represents non-functional attributes of a service. However, some literature describes QoS as a sub-class of non-functional properties. In the context of this thesis, we use QoS to represent all non-functional attributes.

There is a plethora of web services currently available, and new services continuously emerge. In service discovery, a service consumer has a list of requirements, and service providers publish services and advertise their capabilities. During the discovery process, service capability is matched to consumer requirements, and a list of web services with similar functionality but with different QoS parameters is returned. In this case, the QoS parameter plays an important role in determining which service will be selected and utilised by the consumer. Due to the complexity of QoS metrics, the issue of service selection is not fully addressed. There is an array of consumer QoS criteria that can be used to differentiate between services. Consumers may place different levels of importance on different QoS criteria. The algorithm must meet current needs to have flexibility in a variety of QoS attributes and be scalable to meet future needs without re-engineering. In addition, several factors give rise to a situation where the list of QoS attributes from the consumer requirements and from the provider parameters are not identical. Services that would otherwise satisfy the consumer's requirements might not be considered simply because of imprecise or partially matching QoS properties. In this case, the discovery and selection algorithm should avoid excluding partially matched services. It should include them in the process of discovery and selection.

The growth and popularity of web services make personalized recommendations one of the most important research issues. Web service recommendations can be considered an automatic process of discovering and recommending several web services to the consumer based on the QoS properties of the web services and the behaviour of the consumer. Collaborative filtering is a significant approach used in the process of service recommendation by exploiting consumers' and providers'

interactions with the system. There are two main types of collaborative filtering approach, user-based and item-based [12]. Additionally, a hybrid approach combining both user-based and item-based approaches has also been proposed [7] [8].

Current collaborative filtering approaches make recommendations based on mutual relationships between users and services, but neglect the latent interests of the users and services. The latent interests of the user can be considered a requirement of the user that will avoid the user having to provide explicit requirements, as it is always prone to error. The latent interests of the service reduce the search dimension by describing each service by a list of latent interests (topics); those topics represent the behaviour of the service. Another well-known problem to this approach [13] [8] is that it does not take the relationship among neighbouring services into consideration during the process of service recommendation. The correlation has an impact on improving the accuracy of predicting missing QoS values.

Since the use of web services continually increases in everyday life, we are entering an era of the Internet of Services (IoS). Traditional service discovery approaches are often supported by centralized registries. The centralised approach cannot cover all aspects of practically unlimited consumer requirements because it is not feasible for any service repository or service provider to meet all consumer demands. A centralized approach is also always prone to single point failures and bottlenecks, hugely affecting the availability and reliability of web services. Therefore, a decentralized P2P approach is proposed where each peer node can represent a cloud platform and provide different services. A practically unlimited number of services can be found outside each cloud platform and can be outsourced based on the

consumer's request. The nodes can dynamically join and leave the network, which provides an elastic environment for a limitless number of services. As a result, each node can outsource services from other nodes based on the context and on social behaviour. Consumer demands will be fulfilled in this way, and the issue of single point failure will be solved using different numbers of nodes with each of them acting as a provider and requester of services.

1.3 Research Aim and Objectives

1.3.1 Research Aim

The overall aim of this research is to develop a novel and efficient method for service recommendation and selection based on Quality of Service (QoS). This includes development of a suitable discovery and ranking method in centralized and decentralized environments. It is necessary to divide the aim into more specific objectives. They are listed in section 1.3.2.

1.3.2 Research Objectives

- To review background research and related work in the area of service selection and recommendation in different platforms.
- To develop an efficient method for service recommendation and selection based on QoS parameters. The method should deal with partially matched services and consumer requirements and adequately rank services by aggregating different QoS parameters and consumer requirements.

- The recommendation and selection method should be scalable and capable of dealing with several web services and QoS parameters (i.e. two-dimension scalability).
- The method should be easily expanded to handle extra aspects like adapting to different environments (centralized and decentralized).
- To develop a hybrid service recommendation method that considers service correlations and service semantic content extraction. The method should consider the consumer's preferences and criteria in order to provide the most suitable services.
- The evaluation should show improvements of the proposed methods in comparison with the existing methods in the area of service selection and recommendation.

1.4 Research Contributions

Based on our research aim and objectives, we can list the key contributions of this thesis as follows:

1.4.1 Partially matched service selection

Based on the listed challenges of partial matching, scalability, and consumer preferences, contributions of partially matched service selection can be determined to include:

- **Service selection algorithm:** One of the key purposes of the investigation of the service selection method is to help consumers select the best available service for their requirements. Consumers may not be able to

make the correct decision, as the selection constraints are complicated. In Chapter 3, we study a QoS-based partial service selection approach by evaluating the similarity score between requested and published QoS parameters. We found that traditional service selection algorithms that reflect an exact match between the request and the published QoS parameters cannot represent the matchmaking process well under certain conditions. To observe the tolerance in some QoS parameters that a service could provide, robust partial matching has been proposed. For calculation of the partial match, a comprehensive service request has been proposed to let consumers express their requirements. The QoS value of each dimension for both service requests and published QoS parameters could be described by attributes such as values, datatypes, weight, and the tendency of the preferences. By using six different equations, the normalized value of each QoS parameter is calculated, and based on the aggregation of the values, the rank of the service will be attributed.

- **Automatic weight value calculation:** It is difficult to evaluate different criteria, as each criterion might be expressed in different ways, and the preferences of each consumer vary for each criterion. Collecting weights and preferences of each QoS parameter from consumers may not be an efficient choice, particularly in complex selection processes with tens or hundreds of consumer requirements. Therefore, we provided an automatic weight calculation method based on weight value dependency among QoS parameters. Based on the method, the distance correlation of QoS

distributions will be calculated and used to predict the weight value of each QoS parameter.

- **Scalability of the partially matched selection method:** Increase in the number of web services has impact in the efficiency and accuracy of the selection and recommendation process. With the increase in the number of services, the selection and recommendation system should be scalable and capable to deal with large number of services. In our proposed approach the scalability of the partially matched selection method will be tested. Based on the results, our approach has to be scalable in terms of increasing the number of services in the selection process.

1.4.2 Personalized service recommendation method

Based on the mentioned challenges in service recommendation and selection, the contributions in this section can be summarized as:

- **Hybrid Recommendation Method:** To achieve efficient web service recommendation and selection, we proposed a personalized service recommendation approach in Chapter 4. In this method, a hybrid approach for web service recommendation is proposed that combines correlation-based and content-based recommendation of services. Unlike other conventional service recommendation approaches, our approach considers correlations between web services based on co-invocation by different consumers, and exploits a Latent Dirichlet Allocation (LDA) model that simultaneously considers the similarities of users and content of web services and extracts latent interests of users and services.

- **Correlation based recommendation:** Some of the well-known approaches [8] [10] ignore relationships among Web services in similar neighbouring selections, which has an important impact on QoS prediction accuracy. In this approach, we use a correlation-based top-*k* recommendation approach to improve the process of finding *k* neighbours by using correlations between Web services instead of similarities. The relationships between services can be achieved by using the number of co-invoked web services by different consumer of services. Then a correlation graph will be generated which shows the services and links between the nodes representing the correlation between services. Based on the relationships between services in the correlation graph, a service ranking matrix will be generated to be used in the prediction and recommendation process.
- **Content based recommendation:** Content-based service recommendation is the process of the exploration of the similarities of the content between services. In the recommender system, model ratings of services depend on an underlying set of topics. The model has a three-layer representation: service, topics, and users. Each service is represented as a probability distribution over topics, and each topic is a probability distribution over users. In addition, by using consumer's latent interests, the next possible consumer preferences can be statistically estimated.

1.4.3 Service discovery and selection in decentralized environment

In this section, a service discovery and selection algorithm in a decentralized environment is proposed that will improve the efficiency and performance of the discovery and selection process in comparison to the previous work in this area. Contributions can be summarized as a decentralized discovery and selection method and the simulation of a prototype platform.

- **Decentralized discovery and selection method:** Chapter 5 introduces a service discovery and selection method in a decentralized environment to address the need for acquiring the QoS attributes based on consumer requirements and selecting the required services more accurately. A homophily-based approach is proposed, meaning that the most noticeable attributes of services that determine similarities between nodes in the system are utilized as a method of bootstrapping and creating links between the nodes of the decentralized environment. The discovery process is based on semantic similarity, previous interaction, and the social behaviour of nodes. It will aid the discovery process by creating communities based on the similarities, and by providing a more dynamic environment for joining and leaving nodes. It does not just concern discovering services, it also considers the selection of the best available service by taking into account the QoS properties of the services.

Simulation of prototype platform: A prototype platform is proposed for a self-organized discovery and selection environment using a real world semantic dataset.

To observe the evolution of network topology in the simulations, the process starts with the initialization of nodes, resources, and social profiles. Then the contents of the semantic services are used to evenly distribute them among the self-organized nodes in the system. The experimental results with a real world semantic web service dataset show that this approach achieved better performance and efficiency in both the discovery and selection processes.

1.5 Thesis Organization

The rest of the thesis is organized as follows:

- Chapter two:

  In this chapter, I review some background knowledge and related work in the area of service recommendation and selection in different environments.

- Chapter three:

  In this chapter, we propose an efficient algorithm for partially matched services in Internet of Services. Our approach considers partial matching, scalability, and personalization of QoS preferences. The proposed approach reflects IoT service ranking and selection algorithm by considering multiple QoS requirements and allowing partially matched services to be counted as a candidate for the selection process.

- Chapter four:

  In this chapter, we propose a personalized service recommendation using correlation-based and semantic content features. The work considers accuracy, diversity and cold start problems for the recommendation of web services. We propose a hybrid service recommendation approach using latent topic modelling and use this as a technique for

dimension reduction, which extracts latent interests of services and consumers in a probability distribution form. In addition, correlation between neighbouring web services is considered to enhance the accuracy of prediction of QoS properties of web services, and to return a list of desired recommended web services to the consumer.

- Chapter five:

  In this chapter, we propose context-aware service discovery and selection in decentralized environments. In the approach, homophily similarity was used for creation and distribution of nodes. The discovery process is based on the similarity of nodes, and behaviour, which will help the discovery process in a dynamic environment. Our approach is not only considering service discovery, but also the selection of the best available web service by taking into account the QoS properties of the web services.

- Chapter six:

  In this final chapter we present the overall conclusions of research using the stated approaches and provide some suggestions for future directions for further exploration.

Chapter 2 Research background and literature review

2.1 Background of Web Services

In the service oriented environment, web services are considered one of the most widely used implementations. The World Wide Web Consortium (W3C) defined a web service as follows:

> A web service is a software application identified by a URI, whose interfaces and bindings are capable of being defined, described, and discovered as XML artifacts. A web service supports direct interactions with other software agents using XML-based messages exchanged via Internet-based protocols [14].

Based on this definition, web services can be published, described, and invoked. In addition, web services are understood as a set of operations with input parameters and output values that are used for developing a complex system by using loosely coupled services where components are not dependent on others. Services are platform independent entities that can be in atomic form, a service whose implementation is self-contained and does not invoke any other service, or composite form, a service whose implementation calls other services. Throughout the dissertation, services are not limited to only web services.

Web services, however, perform all the characteristics of a service oriented architecture (SOA) and are considered an implementation of SOA. “SOA is a flexible, standardised model with a deeply rooted concept of encapsulating application logic within services to better support the integration of various applications and the sharing

of data" [15]. With SOA, there are three main entities, the service provider, the service consumer, and the service registry, as shown in Figure 2.1.

Service Provider: This entity provides services to consumers. Each provider can deliver one or more services to consumers. A service provider can be a device, smart phone, business entity, publicly available service repository, government body, academic institution, etc.

Service Consumer: This entity invokes services, and can be a human consumer, another web service, or a web application.

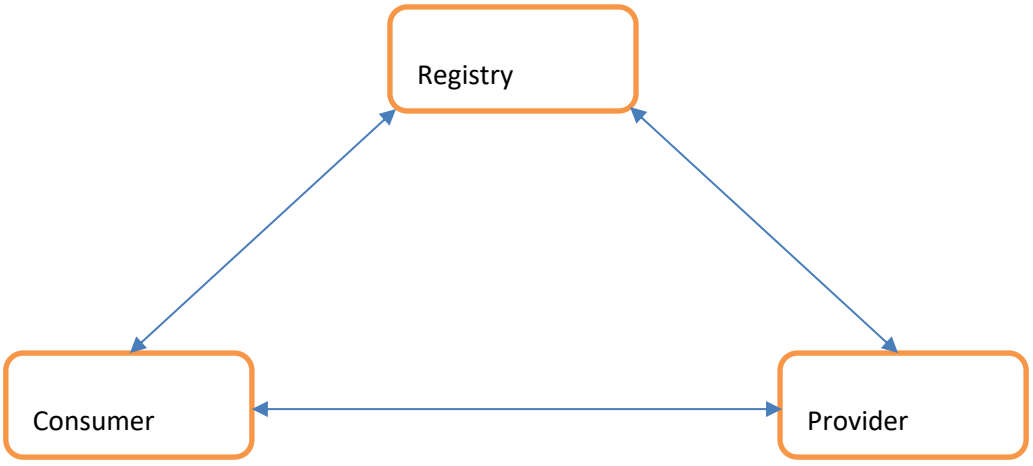


Figure 2.1 Web Service Architecture Model

Service Registry: This entity contains a collection of published web services that can be searched and accessed. The service registry is responsible for the coordination of requests from consumers and providers and is also responsible for defining and encoding the service information. There are two kinds of service registries. One type

is public, which are available through the internet, and private, which are only accessible to a limited number of consumers.

2.2 Background of Service Selection

Web service is a mature concept concerning the delivery of software services over networks that are communicated through web-based standards and protocols [16]. Many techniques for service definition, publication, discovery, and invocation have been introduced by academia and industry, such as XML-based Web Service Description Language (WSDL) and semantic markup language (OWLS, OWL). These techniques have significantly accelerated the process of web service discovery and selection. Service selection is the process of finding a suitable service that matches the consumer requirements. Essentially, the service selection process consists of gathering the requirements of the consumers and the information published by registered service providers, then matching the requirements to the services. The requirements of the consumers usually include a description of the service and the quality of the service, for example a hotel booking consumer looking for hotel in London with four or more stars as customer review , while the published information on the service contains the semantic behaviour (such as inputs, outputs, pre-conditions, and post-conditions) and QoS parameters.

Several service selection approaches and techniques have been proposed, such as Multi-Criteria Decision Making (MCDM) [9], optimization techniques [10] such as dynamic programming [12], integer programming [13], or greedy programming [14], and Analytic Hierarchy Process/Analytic Network Process (AHP/ANP). Based on the literature, three service selection approaches have been identified:

- **Functional service selection approach:** Functional service selection is the process of matching a suitable service to a consumer's requirements based on the functional properties of the service. Normally, semantic descriptions of the services are used for functional service selection. In the semantic web, several approaches have been introduced to describe the semantic behaviour of services by using ontologies, such as WSDL-S, OWL-S, and WSMO. Semantic service matchmaking and selection are considered logic inference tasks [17]. The author in [18] proposed an extended signature-matchmaking to the specification of semantic matchmaking in the semantic web. Sycara et al. [19] proposed a DAML-S based heterogeneous web service discovery.

- **Non-functional service selection approach:** With web service selection, a list of web services might provide similar functionality, but each one may have different QoS parameters. With non-functional based service selection, services will be selected based on the non-functional properties that are characterized by QoS and the context of the service. There is diversity in the QoS parameters based on different consumers. Consumers want to select a service based on QoS preferences. QoS can be gauged by response time, throughput, availability, reliability, cost, etc.; the context of the service can refer to location, time, provider's profile, e-mail address, etc. Zeng et al. [20] [21] proposed two QoS driven methods for web service selection and recommendation. The methods compute the quality ratings from the users of a service and then aggregate them using a simple arithmetic average to derive the QoS without considering the context. In

[22], a tree model has been proposed to represent quality parameters and to explore user QoS preferences. The author also proposed a QoS broker-based architecture to facilitate the work of requesters in selecting preferred web services. In this approach, the tree structure is limited to only four constraint nodes, and only numeric data criteria were considered in the selection. Tian et al. [23] introduced WSQoS for QoS-based web service selection and monitoring, and supported the use of the service broker for the QoS based service selection.

- **Trust based service selection:** The trust (reputation) based service selection uses consumer feedback, or ratings, to identify good service providers. In an environment where service users do not have information about service providers, it is hard for a user to select a service. If the user selects a service without having prior knowledge about the provider, the result might be dissatisfaction during the service use. In such situations, service reputation can help a user to select an optimal service. Reputation is an important factor in the process of service selection; it helps the process to give a more accurate result to the user. In [24], middleware is proposed for QoS-aware service composition. In that approach, service reputation is considered, and an average rating method is used for reputation. It is the simplest form of service reputation, calculating the average of all user ratings and using this as a reference for new service users. Amazon uses the same average rating for customers. This is not considered to be an effective method, as the number of ratings may increase rapidly, and the view of the users towards the service may change over time. EBay employs the same approach as in

[25]. This approach uses the difference between the negative and positive feedbacks of users. Other approaches for service reputation are available. In [26] the reputationNet method was proposed which helped in calculating effective service recommendation. In the work, reputation was calculated based on the standing of the service designer and it is popularity. Zacharia et al. [27] proposed a centralized reputation model that allowed consumers to give feedback on a service and used that feedback as a measure for it's reputation. This kind of approach has been used by online auction websites such as Yahoo. In [28], a novel method for calculating service reputation is proposed. The method has two phases. The first phase uses a dynamic weight formula to calculate reputation that reflects the latest tendency of the service, and the second is removing the negative effects of unfair ratings using an olfactory response formula. The work depended on first impression and mind-set. Most of the time, decisions based on first impressions are correct, but this is by no means guaranteed.

This thesis explores non-function based service selection. Particular focus is given to the QoS aspect of the non-functional properties. For the functional aspect, our approaches assume that the service providers are classified in the same functional groups by semantic presentation with different QoS values. Furthermore, it is considered that the QoS values indicate the objective opinion of the service consumers. The subjective opinion, such as trust value, also needs to be considered for a comprehensive service selection method. The QoS based service selection approaches are introduced in further detail during the following section.

2.3 QoS parameters

As the number of web services with similar functionality rises, the first step is to find the QoS attributes that need to be gathered and analysed in the QoS based service selection process. Classification of QoS parameters can be used as a method to examine the parameters and make the selection process easier. Several approaches for modelling QoS attributes for web services have been proposed by researchers [29] [30]. Dobson at el. [31] proposed an ontology-based classification of QoS attributes called QoSOnt, which identifies two QoS attribute classes, measurable and unmeasurable attributes. Most QoS selection approaches concentrate more on the detail of the ontology rather than on finding an optimal selection mechanism. In [32], detailed ontologies were designed to represent user non-functional attributes and were integrated with WSMO. Despite having detailed ontologies, a proper selection algorithm was lacking, and there was little explanation of how user preferences would be included in the approach. [33] proposed a QoS based service selection using non-functional attributes and context attributes of the services.

There is no standard model for the representation of QoS attributes of services; different researchers have proposed different approaches to represent QoS attributes in their work. The WS-QoS class [34] is to represent quality parameters of a service. In this approach, several QoS parameters have been defined, and network level QoS parameters such as packet loss and network delay with the service were defined as well. In a real world case, the service user would be interested in the quality of

ID	QoS attributes	Description	Unit	Tendency
1	Response time	Indicates the expected delay time required to send a request and receive response	Ms	Low
2	Availability	Successful invocation divided by total invocation	%	High
3	Throughput	Number of invocations in a period of time over the network.	Ips	High
4	Success Rate	Number of responses divided by number of request	%	High
5	Reliability	Probability that a request is responded and processed correctly and consistently by the service provider (i.e. Ratio of successful messages to the total messages)	%	High
6	Compliance	Matching with WSDL description	%	High
7	Best Practices	Matching with WS-I Basic Profile	%	High

8	Latency	Processing duration for a given request	Ms	Low
9	Documentation	Amount of documentation in a WSDL file	%	high
10	Cost	Or price; the amount of money that needs to be paid to the service provider for a certain service invocation	$	Low
11	Reputation	The measure of trustworthiness. It is normally based on requesters' direct or indirect experiences and rankings.	%	High

Table 2.2.1 Web Service Quality Attributes

services, but not much in the network level details. Table 2.1 shows a sample list of widely used QoS parameters.

2.4 QoS Based Service Selection

Every service normally provides some functionality which focuses on the functional properties of the composed service, and how the functionality is provided can be determined by it is non-functional parameters, which can be denoted as QoS. For instance, QoS parameters like response time identify the time required for a service to respond to the requester, or cost parameter states the amount of money required in order to invoke that service. Therefore, for service discovery and selection, considering functional requirements is not enough. Non-functional requirements play

an important role as well. The decision of the service consumer should reflect both functional and QoS requirements.

The literature contains various approaches to considering QoS properties for the service discovery and selection process. Multi-criteria-based approaches, like Analytic Hierarchy Process/Analytic Network Process (AHP/ANP), outranking, multi-attribute utility theory (MAUT), and optimization techniques like linear programming approaches, graph-based approaches, network topology based approaches, game theory based approaches, swarm intelligence based approaches, and genetic algorithm based approaches, are explained below.

- AHP/ANP-based approach: The AHP [35] [36] is a web service ranking method that forms a hierarchy to organize service information and dependencies among service criteria. There are three main phases which the method depends on. They are decomposition, comparison judgement, and synthesis. In the decomposition phase, the hierarchy is formed to model dependencies among decision elements, as shown in Figure 2.2. In the comparison judgement phase, same level comparison is applied among elements to find the influence of elements on the higher levels. Finally, in the synthesis phase, priorities of elements are synthesized to prompt the priorities of alternative A and B. However, the ANP [37] [38] is considered an extension of the AHP by providing solutions to problems that cannot be hierarchically structured. With ANP, criteria and alternatives are clustered instead of layered, as shown in Figure 2.3. The arrows between clusters shows the influence between criteria and alternatives.

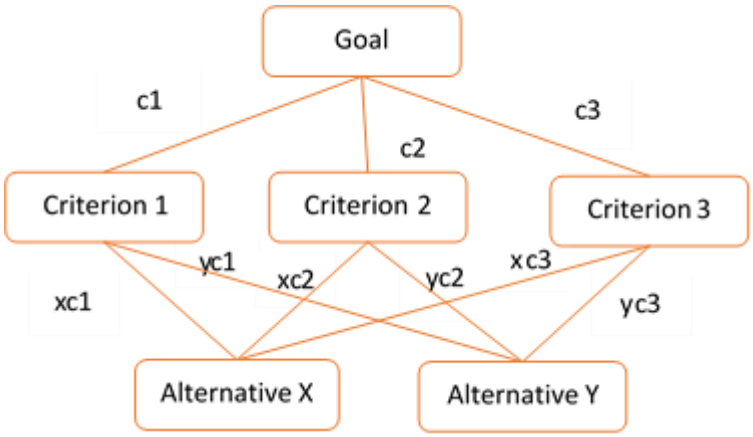


Figure 2.2 AHP Hierarchy

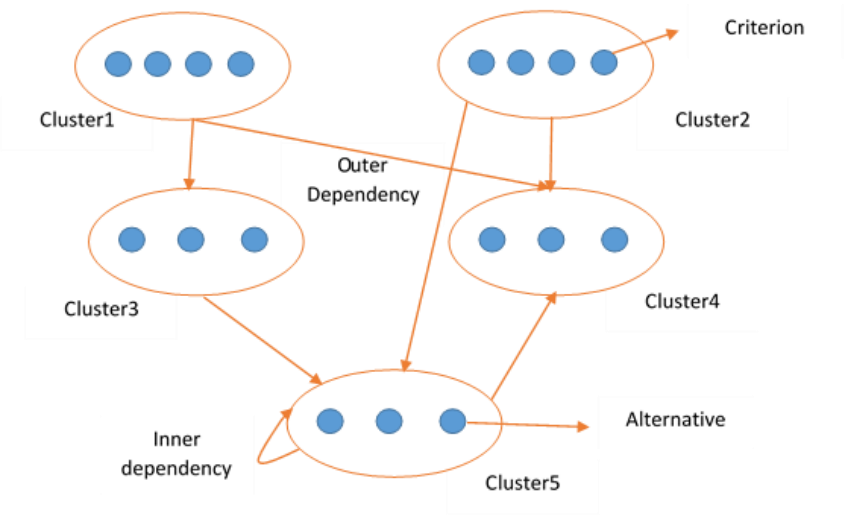


Figure 2.3 ANP network

- MAUT based approach: Unlike AHP, which is used for comparisons between a pair of criteria, Multi-Attribute Utility Theory (MAUT) depends on utility functions [36]. In a utility function, the preference of the consumer is considered, and the criterion is evaluated based on the

consumer's preference. [39] discussed cloud service selection, in which a multi-attribute utility function was used in the selection process to trade-off between maximized gain and minimized cost. In this approach, the selection process was performed in two levels. In the first level, a list of services was selected that meets user requirements using Simple Additive Weight, and in the second step a service was selected from the list that reached the trade-off between the gain and the cost.

- Outranking-based Approach: Another method from the MCDM family is the outranking method [40]. This method can be defined by example as follows: Alternative A_i can outrank A_j if on a great part of the criteria, A_i performs minimally as well as A_j (concordance condition), while it is worst performance in the other criteria is still acceptable (non-discordance condition). It examines the degree of dominance of one candidate over others at the criterion level. The key difference between MAUT and outranking is that the former returns the best choice while the latter provides a short list of alternatives.
- Linear Programming Approaches: The aim of QoS-based service selection is to maximize or minimize the utility function in parallel with the satisfaction of the QoS constraints and requirements. The process can be considered as Mix Integer Programming (MIP) where objectives and constraints are linear. Zeng [41] proposed a linear programming approach to solve the problem of service selection. In the approach, the execution plan was presented using Direct Acyclic Graph (DAG) in which each service was represented using a QoS vector. The aggregated QoS value

was computed for each service using Simple Additive Weighting (SAW) in order to capture the optimal solution. Within the approach, the computation cost was very high, especially in the case of large computational pools. In addition, the service ratings were collected directly from customers, and the values aggregated without any consideration of the context of the service and consumer. Branch and Bound (BB) algorithms is an approach usually used to solve the problem of the Integer Programming (IP). Yu et al. [42] proposed a BB-based approach for composite service selection while considering the sequential, parallel and conditional invocation types. In addition, the authors proposed a heuristic approach to reduce the computation time.

- Dynamic Programming: Dynamic programming is primarily used to tackle the optimization problem by breaking down complex problems into smaller sub-problems, then solving those smaller problems and saving the result in a memory-based data structure. Chang at el. [43] proposed a dynamic programming-based approach to select cloud storage providers based on the survival probability of stored data with a fixed budget. In the approach, a mathematical formula was included for cloud service provider selection in which the object function and cost measurement were defined. Two methods were used to analyse the availability of the storage service, minimum failure probability for a given budget, and maximum validity with a given budget. The earlier one claimed a solution to minimize failure probability for a fixed budget.

- Prediction based approaches: The prediction of QoS-based approaches is another method for dealing with service selection. Zheng et al. [21] proposed a CloudRank algorithm, which is a personalized QoS ranking prediction framework based on the past service invocations of other consumers to predict the QoS ranking of cloud services in the process of service selection. Firstly, the similarity between an active consumer and a training consumer for the same set of services is calculated by using the Kendall Rank Correlation Coefficient (KRCC). Secondly, according to the similarity value, two greedy methods CloudRank1 and CloudRank2 were proposed to calculate ranking prediction for current consumers. In CloudRank1 consumer preferences for two different services would be calculated based on the QoS parameters of both services, then all the services would be sorted based on the consumer preferences of the pair services using the greedy method. Finally, all services would be ranked from highest to lowest, with the service pair with the highest consumer preferences being ranked the highest. In CloudRank2, a confidence value, which measures the difference between preference values, is considered for overall consumer preferences. The confidence value can be used to increase prediction accuracy as it determines consumer preferences for different services.
- Game theory-based approaches: Game theory approaches have been used in the process of QoS-based service selection. Esmaeilsabzali et al. [44] proposed a game theory-based service selection using a multi-dimensional single rounded auction in which the auction was repeated multiple times.

For the selection process, three QoS parameters, availability, reliability, and price, where used. Every provider submits a bid for a consumer request, with similar approach to auctions. The consumer selects the bid best suited to the requirements. Shen et al. [45] also proposed a game theory-based method in which the approach response time was determined by the performance of the provider and the number of consumer requests to the service. The author used a strategy in which the selected service would provide minimum execution time with certain probability.

2.5 Partially Matched Services Selection

Currently, we are witnessing an increase in the number of available services in the IoT environment which requires an automatic and scalable approach for the discovery and selection process. There are number of approaches for the discovery and selection of services in the IoT environment [46] [47] [48] [49]. However, discovery and selection is a challenging process, especially when available services have a similar functionality but different QoS parameters, such as reliability, security and availability. In line with this the challenge is furthered by an increasing number of available services. Many approaches concentrate on discovery or functional matchmaking, such as in studies [50], [51] [52], and [53]. Concentration on semantic description of services in the form of Input/output and neglecting QoS parameters resulted in good performance, however the accuracy of the selection process was inadequate.

A number of approaches for selection and ranking techniques have been proposed that take QoS parameters into consideration during discovery and selection, as in the

papers of Chao et al. [54], Huang et al. [55], Ngan et al. [56] and Kiritikos et al. [57]. With the diversity of QoS parameters consumers will select a service based on their QoS preferences. For example, a consumer for flight services may be more interested in price and safety, while a consumer for printing services would be more interested in colour matching and speed of printing.

In a similar approach, Liu, Ngu and Zeng [58] used average ranking, which treats consumer QoS preferences equally. This negates the priority of different consumers in defining QoS properties. In a paper by Estrella et al. [59], WSARCH was used. WSARCH is based on a service oriented architecture that provides services with verifiable QoS attributes. The approach monitors service providers and analyses data to provide suitable services for the consumer's QoS request. In Kritikos and Plexousakis [60], two non-functional matchmaking techniques were proposed. The first one relies on exploiting similarity relationships between offers to organize them, while the second one relies on organizing service offers based on non-functional metrics.

The approaches mentioned consider QoS parameters to be precise and to be provided by the consumer. But in general, QoS parameters from consumers are imprecise and are expressed in different forms. In addition, little or no detail is given on how to calculate the dependencies between QoS parameters, which will influence service ranking and selection. Fuzzy logic has been used to deal with ambiguous QoS parameters. [61], [62], and [63] used fuzzy logic to deal with QoS parameters. However, the approaches provide solutions to ambiguous and imprecise QoS parameters and do not address any dependencies among QoS parameters.

In [64], a middleware service selection approach titled SSM_EC was proposed. SSM_EC utilizes an outranking ELECTRE algorithm. QoS from the consumers and providers are collected. Based on that information, the concordance index, the discordance index, and the credibility degrees are calculated. With the concordance index, the reliability of the outranking relation between two candidates can be presented for a given criterion. The discordance index is used to determine the correctness of the outranking relation based on the performance difference between the two alternatives by using the criterion. The credibility degree combines both the concordance and discordance index to provide the outranking relation for the whole set of criteria. Finally, ascending and descending ordering of the services is performed, and the rank of the services is calculated.

Other approaches [65] [66] used Simple Additive Weight (SAW)-based methods to rank alternatives for cloud service adoption. Based on decision making theories, in [65], the authors analysed problems that might be encountered by consumer's criteria during the service selection process and identified solutions that can be used in that process. For the proposed service selection framework—MADMAC (i.e. Multiple Attribute Decision Methodology for Adoption of Clouds), the author introduced an attribute hierarchy which consisted of six attributes to define the criteria for the decision making process. With the proposed method, experts' opinions were also taken into consideration and were used to determine the value of the criteria. Finally, a SAW-based method was used to calculate the rank of the services.

Zhao et al. [67] proposed SPSE (Service Provider Search Engine), for service selection and scheduling. The approach focuses on service selection in SOA and cloud

computing environments with the consideration of users' personalisation and multiple objectives. The proposed method includes four basic operations: search, filter, rank, and update. In the first step, by using indexing technology, services with required service attributes and available service providers are searched. The found services are then filtered via a Pareto optimal-based selection method to improve scheduling efficiency. In the third step, services are ranked; each parameter of the service is first ranked considering the values given by the service provider. The ranked values and consumer preferences are then added to arrive at the final ranking of the services. The update operation automatically calculates and updates consumer preferences according to the principal requirements of consumers and their subsequent choices of services.

Unfortunately, many of the existing approaches use a precise value comparison for QoS parameters. Any missing QoS parameters of a service lead to exclusion of that service in the process of ranking and selection. The suggested ranking techniques should be tolerant, with partial matching of parameters. This tolerance, however, should be exercised with care, as it might lead to failure of the selection process and consequently fail to satisfy the needs of the service consumer. In contrast to SPSE [67] and SAW based [65] approaches, we develop an automatic method to autonomously determine the weight value of QoS attributes by calculating correlation dependencies among different QoS attributes, and a new service ranking algorithm, which will be presented in the following section, is introduced by considering partial matching of services.

2.6 Web Service Latent Topic Modelling

Nowadays the number of services is increasing rapidly, and the web is moving to the web of services. Likewise, the number of service consumers is increasing, and they have a greater diversity of requirements. As a result, service discovery and selection has become a major challenge. There are limitations in keyword-based discovery. For example, it is difficult for consumers to select a suitable service for their requirements, as the number of functionally equivalent services is huge. In addition to this, keywords are described by natural languages and cannot express the semantic concepts of a particular service. To overcome the limitations of keyword-based discovery, a probabilistic topic model has been used in our work that treats each service description as a vector for the recommendation of web services. Probabilistic topic models are a suite of algorithms whose aim is to discover the hidden thematic structure in large archives of documents. Probabilistic topic models treat textual data that humans have not been able to categorize manually, and it uses hidden variables to find latent semantic structure in a corpus of textual data.

In this section, we introduce some basic concepts of parameter estimation and provide a brief introduction to two of the most commonly used probabilistic topic models Probabilistic Latent Semantic Analysis (PLSA) and Latent Dirichlet Allocation (LDA).

2.6.1 Parameter Estimation

This is the process of estimating the parameters of a selected distribution using available sample data. There are two inference issues. Firstly, the issue of estimating a set of distribution parameters θ from a set of observed variables w_i (i.e. $P(\theta \mid W)$).

The second issue is prediction which is to predict the probability of new observations $\tilde{w}$ given previous observations (i.e. $P(\tilde{w}|W)$.

There are different parameter estimation methods available. We start from simple maximum likelihood estimation, then maximum posteriori estimation, and finally Bayesian estimation.

2.6.1.1 Maximum Likelihood Estimation:

Maximum Likelihood (ML) estimation obtains the most likely values of parameters for a given distribution. It is mathematically formulated as

$$L(\theta \mid w_1, w_2, ..., w_R) = \prod_{i=1}^{R} f(w_i \mid \theta)$$

where θ is the unknown parameter that is to be estimated from *R* observed variables $w_1, w_2, ..., w_R$.

$$L(\theta \mid W) = \prod_{w \in W} P(W \mid \theta) \qquad (2.1)$$

It is often simpler to use the log likelihood, L=*Log* ℓ , in which case the MLE can be written as

$$\theta_{ML} = \arg\max_{\theta} L(\theta \mid W) = \arg\max_{\theta} \sum_{w \in W} \log p(w \mid \theta) \qquad (2.2)$$

To determine the probability of new observation $\tilde{w}$ given the data W , the following equation can be used:

$$P(\tilde{w}|W) = \int_{\theta \in \phi} P(\tilde{w}|\theta)P(\theta|W)d\theta \quad (2.3)$$

$$= \int_{\theta \in \phi} P(\tilde{w}|\theta_{ML})P(d\theta)d\theta = P(\tilde{w}|\theta_{ML}) \quad (2.4)$$

where the next observation is anticipated to be distributed with the estimated parameters θ_{ML}.

2.6.1.2 Maximum a Posteriori Estimation

Maximum a posteriori probability (MAP) estimating is a kind of a posterior distribution. The MAP is useful to get a point estimate of a hidden quantity by using empirical data. It is similar to ML estimation, but allows the addition of some prior beliefs about the parameters by weighting them with a prior distribution $P(\theta)$. Therefore, it can be considered a regularization of ML estimation.

$$\hat{\theta}_{MAP} = \arg\max_{\theta} P(\theta|W) \quad (2.5)$$

By using the Baye's rule, the equation can be rewritten as

$$\hat{\theta}_{MAP} = \arg\max_{\theta} \frac{P(W|\theta)\mathrm{P}(\theta)}{P(W)} \qquad where \qquad P(W) \neq f(\theta)$$

$$= \arg\max_{\theta} P(W|\theta)P(\theta)$$

$$= \arg\max_{\theta} \{L(\theta|W) + \log P(\theta)$$

$$= \arg\max_{\theta} \{\sum_{w \in W} \log P(W|\theta) + \log P(\theta)\} \quad (2.6)$$

In comparison to Equation 2.2, a prior distribution is added to the likelihood. The prior $P(\theta)$ can be utilized to add extra knowledge and prevent overfitting by enforcing preferences to the simpler models.

2.6.1.3 Bayesian Estimation

Bayesian estimation (BE) is an estimator in estimation theory that minimizes the posterior expected value and maximizes the posterior expectation of a utility function. BE is an extension of MAP estimation by using a distribution over the parameter set θ instead of carrying out a direct estimate. Calculation of the posterior is the main step in this approach:

$$P(\theta \mid W) = \frac{P(W \mid \theta)P(\theta)}{P(W)} \qquad (2.7)$$

In Bayesian estimation, there is no restriction for finding the maximum, so one must calculate the normalization term using the probability of the observed $P(W)$ in Equation 2.7. The value can be expressed by the total probability of parameters, as follows:

$$P(W) = \int_{\theta \in \phi} P(W \mid \theta)P(\theta)d\theta \qquad (2.8)$$

With addition of new observed data, the posterior in Equation 2.7 is automatically changed and is ready for the analysis of its statistics. Since the normalization integral in Equation 2.8 is the complex part of Bayesian estimation, it can be investigated further. Bayesian estimation extends MAP by ensuring exact equality to overcome the prediction problem, as below.

$$P(\tilde{w} \mid W) = \int_{\theta \in \phi} P(\tilde{w} \mid \theta)P(\theta \mid W)d\theta$$
$$\int_{\theta \in \phi} P(\tilde{w} \mid \theta)\frac{P(W \mid \theta)P(\theta)}{P(W)}d\theta \qquad (2.9)$$

where the posterior $P(\theta|W)$ substitutes an explicit calculation of parameter θ. With the integration over θ, the prior belief is automatically added into the prediction. This is a distribution over $\tilde{w}$, and can be analysed within the variance.

2.6.2 Probabilistic Topic Modelling

With the increasing popularity of services, service discovery and selection have become a challenging issue. With service discovery, services will be found based on functional requirements that can return a list of functionally equivalent services. QoS plays a very important role in recommending and selecting the best available service for the consumer from the discovered list of services. In recommendations, all the discovered services will be ranked based on the QoS parameters and returned to the consumer.

A machine learning probabilistic topic model is one of the approaches for extracting latent variables from service contents and for using the latent variables to describe services. Similarities between the service latent variables and consumer queries determine the relevancy of the service to the consumer query. We are going to discuss two commonly used probabilistic topic models, PLSA and LDA.

2.6.2.1 Probabilistic Latent Semantic Analysis (PLSA)

Probabilistic latent semantic analysis (PLSA) is a statistical technique for the analysis of two-mode and co-occurrence data. It is based on an aspect model. For web services, observations can be considered in the form of words and services (s_i, w_j). PLSA models the probability of each co-occurrence as a mixture of conditionally independent multinomial distributions, as follows:

$$P(s_i, w_j) = \sum_{f=1}^{k} P(z_f)P(s_i \mid z_f)P(w_j \mid z_f) \qquad (2.10)$$

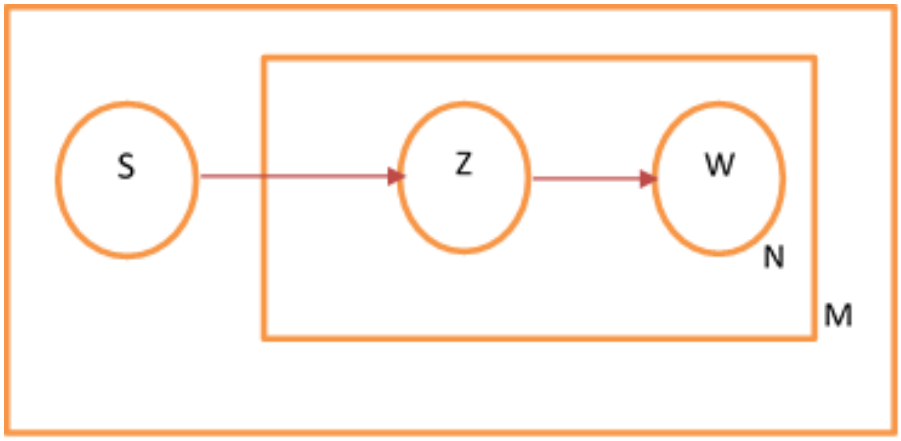


Figure 2.4 Probabilistic Latent Semantic Analysis Model

In formula 2.10, s_i represents i^{th} service in the corpus, $z_f \in Z$ is a latent topic drawn from the service's topic distribution, and $w_i \in W$ is a word from the word distribution. Both *S* and *W* are observed variables, while *Z* represents latent variables. It is assumed that the description of the service and words are conditionally independent for a given latent factor. After the identification of the latent variables $z_f \in Z$, services can be described by a probability distribution of latent topics $P(z_f \mid s_i)$, and the service s_i can be a part of a certain topic group. With PLSA, the first step is the formation of the observed probability $P(s_i, w_j)$. The latent parameters $P(Z), P(S, Z), P(W \mid Z)$ can be discovered using estimation methods.

2.6.2.2 Latent Dirichlet Allocation (LDA)

Latent Dirichlet allocation LDA is a generative probabilistic model for the collection of discrete data, such as text corpora. LDA is a three-level hierarchical Bayesian model in which each item of a collection is modelled as a finite mixture over an

underlying set of topics [68]. Each document can be described by multiple topics. The LDA model tries to find those topics in a corpus of documents, or as in our case, of services. For example, Figure 2.5 shows a document about "Seeking Life's Bare (Genetics) Necessities" that is about a data analysis approach to determine number of genes required for the survival of a living organism.

"Figure removed for copyright reasons".

The figure can be found in the below paper, page number 78

David M. Blei, " Probabilistic topic models," *Communications of the ACM,* vol. 55, no. 4, p. 77–84, 2012.

Figure 2.5 The intuition behind LDA model, Source: David M. Blei [69]

After removing stop words from the documents, such as *the, can, be, and,* etc., the remaining words have been highlighted in different colours based on topics that they relate to. There are three main topics that the document is about. They are (1) data analysis, like "computer, numbers", which all are highlighted in blue, (2) genes, like "Genome, sequence", which are highlighted in yellow, and finally (3) evolutionary biology, like "organism, survive", which are highlighted in red. For service *s,* each word *w* can be generated by using a latent variable *z* from the topic distribution and

then getting a word from the topic-word distribution. The distribution of the topics in the document determine the groups that the document belongs to. It is assumed that a number of topics with a distribution of words exist in the corpus. While the aim of LDA, which is mapping of high-dimensional count vector to lower dimensional variables in latent semantic space, is similar to that of PLSA, LDA provides a way to improve PLSA by introducing Dirichlet prior to the service-topic distribution.

LDA provides the probability of words in services. The following equation shows the probability of w_i for a given service:

$$P(w_i) = \sum_{f=1}^{k} P(w_i \mid z_i = f) P(z_i = f) \qquad (1.11)$$

Where w_i is probability of i^{th} word by using the latent factor z_i, $P(z_i = f)$ is the probability of drawing w_i from topic f, and $P(w_i \mid z_i = f)$ is the probability of finding w_i in given topic f.

2.7 Hybrid Service Recommendation

Web service recommendation and selection has been an essential research issue since the emergence of Web service technologies. With a recommendation system, consumer preferences can be represented for suggesting services to consumers. Several techniques have been proposed for service recommendations, such as collaborative filtering, content-based, knowledge-based, and others. Sometimes, to improve performance, some of these methods have been combined to make a hybrid method. The content-based approaches recommend web services similar to those that a consumer appreciates based on the service's properties. However, the collaborative

filtering approaches recommend services based on the similarities of different consumers.

In different approaches, QoS can be considered a means to support improving service recommendation and selection. It considers different QoS parameters from different services that perform similar functions, and in addition, it considers consumer preferences in the recommendation process [20] [70]. In these approaches, the quality of a service has been considered based on it's QoS parameters. It is assumed that QoS parameters like response time are available and easily accessible, but this is unlikely for most QoS parameters, as mentioned by Zheng et al. in [6]. Some QoS parameters change based on the location, time and the quality of the network (example, response time, latency, availability, etc.). Those parameters might affect the guarantees offered by a service provider, who as a result might not be able to satisfy those QoS guarantees. In order to avoid this issue, in our approach, we consider an efficient method that recommends services based on the correlation of similar services and users in addition to the sematic content of the services.

However, collaborative filtering is one of the most widely used approaches in service recommendations. It collects ratings, or the recommendations of services from consumers based on ratings, and based on the inter-consumer comparisons, generates new recommendations for services. In collaborative filtering, typically a consumer profile consists of lists of services and their ratings based on previous invocations of the services by the consumer. Collaborative filtering systems have been used widely like Netflix and Amazon [71], [72], and [73]. Zheng et al. [6] proposed a collaborative filtering approach for the prediction of missing QoS parameters by using the

information of similar Web consumers and services. The work in [74] presents RegionKNN, a collaborative filtering algorithm that is structured for large-scale Web service recommendations. This approach takes into consideration the service consumers' physical locations and offers a regional model by considering the QoS parameters of services. Collaborative systems can be divided into two types. Memory-based systems compare users and items against each other using correlations or similarity. Examples of memory-based approaches are item-based approaches like [71] and [75] and user-based approaches like [4] and [13]. The second type is a model-based approach in which prediction is made based on the rating data from previous interactions. A few learning techniques have been used in model-based approaches, like clustering models [76], neural networks [77], and latent semantic models [78] [79]. The strength of collaborative filtering systems is that they are independent of the representation of the services being recommended, which can be used for recommendations in complex systems. Nevertheless, these models must be restructured when new users or items are added to the system.

The content-based approach creates consumer profiles based on the estimation of consumer preferences against each of the content features from previous ratings of consumer interactions and the service properties [9]. New services are recommended to the consumer based on the similarity of the service content to the consumer's constructed profile. Different approaches in topic modelling have been proposed in which a topic is a probability distribution over words to extract the contents of services and consumer profiles. The most commonly used are Probabilistic Latent Semantic Analysis (PLSA) [80], and LDA [68]. PLSA is a generative statistical model for examining the co-occurrence of data. It is based on the aspect model [80]. With PLSA,

consumer preferences and service properties are ruled by few latent variables. It has been used in both automatic recommendations [81] and collaborative filtering [78]. However, LDA is a probabilistic model that uses a generative probabilistic model for groups of discrete data [68]. LDA is an effort to develop the PLSA by introducing a Dirichlet prior onto the service-topic distribution. LDA has been used in different fields, like finding scientific topics [82], information retrieval [83], and collaborative filtering [84]. One of the issues with recommender systems is cold starts. A cold start happens when a recommender system tries to recommend a service whose properties are unexplored to the consumer [85]. With content-based or hybrid recommender systems, where there is a profile description, the cold start problem can be overcome by understanding the service or consumer with such content information. For example, the author [86] proposed a probabilistic model that combines item content and collaborative filtering for recommendations to deal with the item-side cold start problem. To address the user side cold start problem, [74] proposed a consumer info aspect model based on consumers' information, such as age and gender. In collaborative filtering there is no content information. The only way to overcome the cold start issue is to better understand both the consumers and the services from limited and sparse rating records. For example, in order to enhance the recommendation performance under cold start conditions, [87] designed a heuristic similarity measure using the minute meanings of services.

In the approach, we present a hybrid approach for better web service recommendations by systematically combining the methods of correlation and content-based. In particular, we propose a model that considers both QoS ratings and the semantic content of Web services. User preferences are modelled as a set of latent variables in

the aspect model [10], which can be statistically estimated using the expectation maximization (EM) method. To the best of our knowledge, this is the first approach that combines correlation and content-based approaches for Web services recommendation.

2.8 Service Discovery and Selection in a Decentralized Environment

Currently there is a plethora of web services available that have been created by industries and companies, and new services continuously emerge. This increase leads to diversity in the quality of web services where large number of web services with similar functionality but different QoS properties, and also leads to diversity in the requirements of the consumer where different consumer requests different requirements. There are two main approaches for service discovery and selection. The approach can either be centralized or decentralized.

With centralized approaches, the main entity has global knowledge about services and resources in the system; this entity provides coordination for service management and discovery. As in [88], [89], [90] and [91], centralized approaches are suitable for systems with a smaller number of web services and a limited number of consumer requests, since it is always prone to bottleneck delays and single point failures. In [92], keyword based searching has been used for service discovery in which a provider receives a keyword request for a service, then a list of discovered services will be returned to the consumer to select from. Yet the accuracy of the keyword search is not the best since it does not consider the semantics of services. However, there are approaches that use the semantic web to improve the accuracy of service discovery

and selection. Examples are found in Agarwal and Studer [50], Plebani and Pernici [52] and Klush, Fries and Sycara [53].

Decentralized approaches can address the issues that exist in the centralized approaches. With a decentralized approach, all nodes are considered equal, without any centralized control; each node can act as a requester and provider. The author of [93] proposed a decentralized resource discovery in which a self-configuring architecture was proposed, and the resources of the system were distributed on network nodes. A requester provides a query to a local node and the local node checks it is own repository. If the node finds the resource, it returns this to the requester; otherwise, the query will be forwarded to another node. The process of query forwarding continues until a resource is discovered or until the Time To Live (TTL) value decreases to zero. In the paper, four query forwarding algorithms were used: random, experienced plus random, best neighbour, and experienced plus best neighbour. Based on the results in the paper, the experienced plus random method had the best performance among the four methods. But the method has some drawbacks. It used First-Found-First-Served (FFFS), in which the first found service with desirable properties is allocated to the search and returned to the requester without searching other resources. In addition to this, the flooding method which is used in the paper for message generation creates a large overload in the system as it floods the query message to all neighbour nodes, and as a result, it raises the issue of scalability.

Sanya et al. [94] also proposed a decentralized resource discovery. Their algorithm was based on TTL reservations and a unicast request forwarding algorithm. The experienced plus random forwarding algorithm has been used. With the method, a

reservation algorithm was added, which resulted in more resource matching by using TTL values with the user's request message. The life time of the resource discovery process is determined by the value of TTL. To avoid any selection overhead from the requester, the method only returns one resource to the requester even if the number of available resources is more than one. The results of the algorithm were compared to those from the FFFS algorithm. Per these results, the scalability of both algorithms in handling the number of requests was similar. However, there were improvements in the utilization of resources in the algorithm, but with an increase in the number of hops. Hong et al. [95] proposed decentralized resource discovery based on node clustering and keyword combinations. The algorithm supported knowledge requests from the user in the form of multi-keywords. Based on the user's knowledge requests, hot key combinations were produced, then the requests were forwarded to the clusters with high correlations to the request. The algorithm improved discovery performance, as the request was forwarded only to the targeted clusters.

A self-organized decentralized system was proposed by the author of [96]. That author proposed an ant colony algorithm to support a P2P overlay network by minimizing the number of links between the nodes of the system. In this approach, six types of ants were used to form the colony: discovery, update, construction, unlink, optimization, and ping. In the discovery process, the ant visits nodes in a random walk process and stores information about the visited nodes in an alpha-table. The construction process is carried out during the addition of new node. If the destination node is full, it will forward the request to a neighbouring node, and a link will be constructed with the neighbouring node. Optimization is performed when a node wants to form a link and the node meets the optimization requirements of the

connection rule. Any node leaving the network will result in the unlink ant, which removes all the existing connections to the node. With the ant overlay network algorithm, enough information about other nodes is collected and stored in the local cache. However, the mechanism suffers from additional overhead, which will affect the response time in the system. To improve on the security and scalability concerns, the author in [97] proposed a decentralized resource discovery method by utilizing programmable networking hardware. The algorithm solved issues like single point failure and management of the entire network, and provided autonomy to the nodes.

Other approaches use P2P for service discovery and selection in a decentralized environment, such as the super node [98]. With super nodes, it is always possible that there will be a failure in the nodes, and replacing them with less qualified nodes results in a decrease in the overall performance of the system. The Distributed Hash Table (DHT) [99], [100], and [101], has been proposed in a P2P environment that uses a key identifier to a document or a service. Different approaches for DHT, like those of Chord [98], Pastry [102] and CAN [103], have been proposed. The differences between the approaches are mainly in the distribution of keys and the mechanism of joining or leaving the nodes. The cost of the search process in DHT is high, and maintenance of the tables is time consuming, especially during the joining and leaving of nodes in the system.

In [104], a decentralised service discovery approach was presented by considering local information in the service discovery process and using homophily between nodes to establish the links. Our work is similar to that approach, but in this approach, homophilic similarity was used for bootstrapping and establishing community in the system. The discovery process is not only based on homophilic similarity but also

considers previous interactions and the social behaviour of the nodes, which creates a more dynamic environment for joining and leaving nodes. Besides the discovery process it also considers the selection of the best available service by taking into account the QoS properties of the services using a partial service selection algorithm.

2.9 Summary

In this chapter, we provided an overview of web service selection and recommendation in different platforms. We began with an overview and understanding of web services and definition of it is related technologies. We also addressed the overview of SOA conceptual framework which is used to build systems for web service publishing, discovery, and invocation. Then the classification of service selection was identified. In the thesis, it is claimed the non-functional service selection is the main focus of our work, in particular QoS based web service selection. After that, the literature of QoS based service selection and partially matched service selection is discussed. In addition, an overview of latent topic modelling was discussed which included parameter estimation and probabilistic topic modelling. Besides that, related work about hybrid service selection was discussed. Finally, literature review about service discovery and selection in decentralized environment has been discussed. The comprehensive overview and literature review laid a solid ground for our research.

Chapter 3 Partially Service Selection in Internet of Services

3.1 Introduction

With the increasing popularity of Internet of Things (IoT) hardware becoming smaller, cheaper, and more powerful, however majority of them have computation and communication capabilities which they use to connect, interact and exchange information with surrounding environments [46]. The proliferation of wireless systems such as Bluetooth, Radio Frequency Identification (RFID), Wi-Fi, telephone data services, embedded sensors and actuator nodes, have allowed the IoT to develop from infancy and it is on the verge to transform current static internet to a fully integrated Future Internet [11].The interaction between all these devices will lead to a large amount of data which needs storing, processing, analysing, and presenting in an efficient, convenient and useable format. In the IoT environment, dynamic network query, discovery, selection, and on-demand provisioning of services are of crucial importance.

One application area of IoT is the emergency and accident management services. The number of emergency cases increases with increasing population and increasing hazard potential from e.g. the number of cars on the roads. According to Health and Social Care Information Centre (HSCIC) [105] the number of accident and emergency attendances in England for 2012-2013 is more than 18.3 million. To provide the most suitable service for emergency cases and achieve better performance, Emergency Management Services can provide a unified platform to connect all local and private sector emergency command centres. Concurrently, wireless sensor networks can be

used for surveillance and precise automated data collection regarding the emergency case and required services.

There are varieties of attributes which determine the type of emergency service available for a given case. The data is generally dynamic in nature such as crew members, and therefore capabilities, in particular vehicles on a particular shift. While the majority of the data is dynamic, the change rate it is not synchronised. For example, some of the data is slowly changing e.g. vehicles owned by a county service; some is moderately changing e.g. qualifications and capabilities of crew members and some is rapidly changing e.g. current location and status / availability. To search for the most suitable service, consideration needs to be given to the different service types: fire, police, ambulance, coast guard, anti-terrorist unit, etc. and then the attributes which would include vehicle type (cycle, motorcycle, car, transit vehicle, aircraft), vehicle capability (four wheel drive, number of seats, number of beds), vehicle equipment (defibrillator, oxygen, breathing apparatus, underwater search equipment, access equipment), personnel capability (fully qualified / part qualified / undertaking training in e.g. resuscitation, working at altitude, crash investigation), current service location, current service availability, owner of service provision (county, borough, private, etc.).

The number of required attributes is significant and not all would apply to the available services which would lead to a need for partial matching and more importantly, best service selection. An example could be for a motorway vehicle accident involving chemical transportation (injuries, vehicle instability, hazardous materials, volatile air supply, chemical clean up, distance to services etc.). In this example a service was not

specified, only a partial attribute list. The search would need to be automated based on a reported incident and call handler recording

For the emergency support service existing technologies in service-oriented systems may be leveraged to provide partial solutions. Within Service Oriented Architecture (SOA) services are considered as self-contained, self-describing, modular applications that can be published, found, and invoked across the web [106]. SOA stores all the services in repositories, those services can be selected and invoked automatically. Therefore it can be suggested that it needs more effort to build a more accurate and efficient service selection method to overcome the challenges facing the process of emergency support service.

In this paper we propose an efficient and dynamic algorithm for selection of disaster services based on partial matching of service QoS attributes. Furthermore, an accurate ranking algorithm is provided by considering the deviation of QoS values from disaster nominal requirements.

From the mentioned points we can identify three crucial issues as challenges in disaster service selection process,

- Partial matching - where the available emergency services do not fully match every QoS requirements. The selection algorithm should avoid excluding those partially matched disaster services.
- Scalability - the algorithm should consider scalability techniques in order to manage large number of emergency services and diversity in the kind of requirements.

- QoS requirements - is an important part of the selection process, it should be taken into account precisely; the selection process should reflect those requirements for different kinds of emergency services.

3.2 Contributions

Based on the listed challenges of partially matching, scalability, and consumer preferences contributions in this paper can be concluded as;

- A service selection algorithm has been proposed that provides emergency cases a suitable service which matches their requirements.
- The algorithm can achieve a high accuracy since it considers all the available services in the selection process. In the case of not finding an exact match, the algorithm considers partial matching of services.
- The system is able to support a large number of services, and by providing distance correlation weighting mechanism it can support different QoS requirements.
- Utilising real world services, the experimental results show that the proposed algorithm improves the quality and performance of the selection process.

The rest of the chapter is structured as follows: Section 3.2 shows and analyses some related work in the field of service selection. Section 3.3 illustrates system architecture and provides detail of ranking algorithm and recommendation of services to the consumer. Section 3.4 provides experimental results of the work. Finally section 3.5 presents the conclusion.

3.3 System Model

3.3.1 QoS Parameters

As the number of services increases, it becomes inevitable that there will be multiple services with similar functionality performing same task but with different quality. For clarification, the 'task' describes the process of providing information and the 'quality' describes how the task is satisfied. Quality, when used in this context, refers only to the task and not the actual data returned. QoS in Internet of Services is usually used to represent non-functional characteristics of services. Different non-functional QoS attributes of services can be divided into two parts, one part as the user-independent properties which have identical values for different users (e.g., price, popularity, etc.), and the second part as the user-dependent properties which have the different values for different service users (e.g., response time, throughput, etc.). The QoS values are affected not only by the service providers, but also by the environmental factors in services computing (e.g., server workload, network condition, etc.).

Researchers have proposed numerous different approaches regarding the use of QoS criteria in services, summaries of many of the methods including benefits and limitations have also been published [30]. It is common in many of the methods to rely only on QoS attribute values published by the service providers. Attributes such as response time and availability may change over time independently of the published values and the values might only be accurate when the service is used within the original network context. All attribute values given by providers are considered subject to validity and integrity.

In the consumer requirements, each QoS parameter can be either mandatory or optional. Optional values are also given a weight factor to determine priority. The tendency of the parameter as shown in Table (1) indicates whether high or low values are regarded as more desirable in an ideal scenario. For example, high Availability and Throughput values imply a better service whereas low Response time and Latency values would also generally indicate better service.

3.3.2 Architecture

The aim of our proposed system is to provide an efficient service selection tool in which service consumer is able to choose the best available service based on consumer requirements and service QoS attributes. Figure 3.1 shows the model within more detail.

Consumer Request Module: The process begins with the consumer supplied requirements, which are different for each consumer and purpose. These requirements initially define the task and desired QoS. As examples, for a given output, a project may prioritise service cost above update rate whereas another project might consider higher cost to be justified to obtain a higher update rate.

Request Manager Module: This module gets the QoS requirements from consumer requirement module and communicates with service repository manager to return the list of available offers which perform consumer required functionality, and extract QoS parameters of the offered services.

QoS Analyser Module: This module analyses the offers and collects the QoS parameters belong to offers. The parameters will be used in constructing accuracy matrix and calculation of offers ranks.

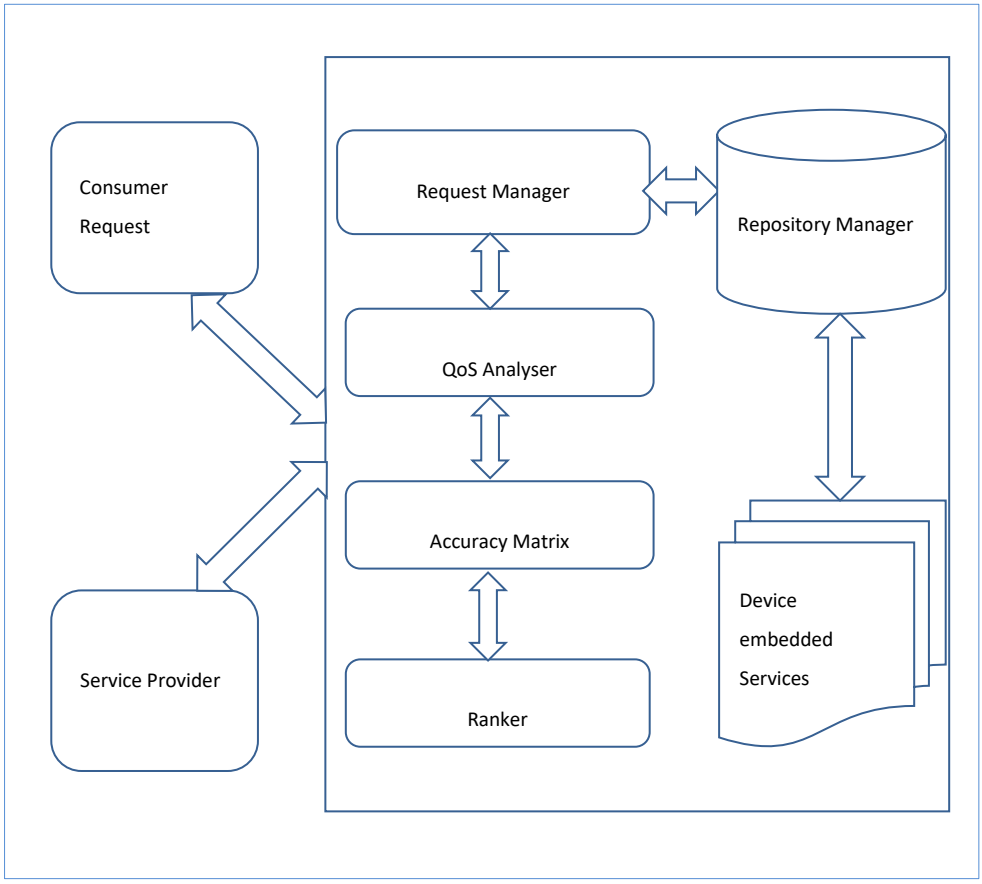


Figure 3.1 Service Selection Model

Accuracy Matrix Module: From the list of services and their QoS parameters a matrix will be formed. The matrix will be normalized based on consumer requirements. Within this module services will be filtered and candidate for the ranking and selection process.

Ranker Module: Based on the Accuracy Matrix each candidate service will be evaluated and ranked. The detail of the ranking process is discussed in the Ranking Algorithm.

Implementation Module: The list of ranked services will be returned to the consumers, and consumers will select the most desired service based on their requirements. In case of failure the implementation, the module will be responsible to find alternative services for the consumer.

3.3.3 Ranking and Recommendation Algorithm

This section explains how our algorithm and accuracy matrix are applied to services with only partially available data in order to generate a more complete, preference ranked service listing. For each selection process there is a list of services which have similar functionalities and may be able to satisfy the task requirement.

Each service can be represented as

Service= {*s_id, s_name, p_id, s_type, I, O, QoS*}

where , *s_id* is identifier for the service, *s_name* is any given name to the service, *p_id* is the identifier for the provider of the service, *I* is a set of service input, *O* is a set of service output, and *QoS* is a list of QoS parameters of the service.

Consumer request is used to define QoS requirements,

Request={*request_id,consumer_id,i_data,s_type, req*}

Where *request_id* is an identifier for the consumer request, *consumer_id* is an identifier of the consumer, *i_data* is a set of data submitted to the provider, *s_type* defines the type of the service the consumer requires, and *req* defines list of QoS requirements from the consumer.

Let,

- Q_R be a set of consumer QoS requirements, $Q_R = \{r_1, r_2, r_3 \dots r_n\}$ where $n \in N$.
- S be a set of potential services with similar functionality, $S = \{s_1, s_2, s_3 \dots s_m\}$, $m \in N$.
- Each set of S candidate services has Q_S property matrices $Q_S = \{Q_{S1}, Q_{S2}, Q_{S3} \dots Q_{Si}\}$, where $Q_{Si} = \{q_{i1}, q_{i2}, q_{i3} \dots q_{ij}\}$, $i,j \in N$. Q_{Si} represents quality matrices for service i.

During the process of service selection its unlikely consumer QoS requirements Q_R has same number of matrices as services QoS parameters Q_{Si}. Q_R is taken as the baseline and quality matrices are arranged as follows,

- Any consumer Q_R which is lacking in service Q_{Si} will be assigned with 0.
- Removing any Q_{Si} which is not presented in Q_R.

If n consumer QoS requirements Q_R has been identified, and m potential services can satisfy consumer functional requirements, the m-by-n requirements matrix is constructed, *R*, as shown in Fig. 3.2. Each column in the matrix represents consumer

QoS requirements Q_R and each row represents a potential service for the selection process.

With the consideration of the original consumer QoS requirements, all services that do not satisfy mandatory requirements are removed from selection process as shown in Algorithm 3.1. The remaining candidate services will be considered in the matchmaking process.

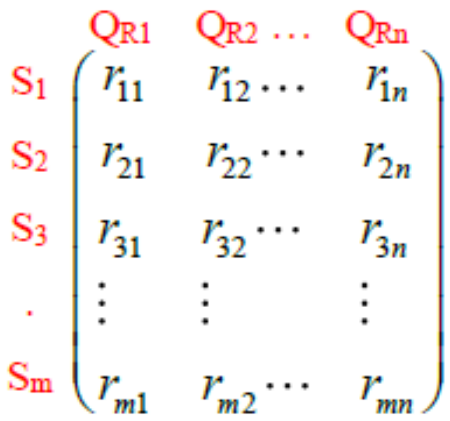


Figure 33.2 Requirements Matrix, R

The calculation of the accuracy matrix, A, is dependent on the tendency of QoS parameters. Tendency explains how the numeric value of a service changes for the service to be perceived as better. The tendency for the majority of values in our example is expected to be high, only Response time and Latency are expected to be low.

Using the consumer specified QoS range and the service derived QoS offered, each element of the accuracy matrix, A, is calculated using the case dependant formulae (3.1) – (3.6).

For values with high tendency:

$$\frac{R_{ij}}{R_l} \qquad \text{When } R_{ij} < R_l \qquad (3.1)$$

$$\frac{R_{ij} - R_l}{R_h - R_l} + \alpha \qquad \text{When } R_l \leq R_{ij} \leq R_h \qquad (3.2)$$

$$\frac{R_{ij}}{R_{\max}} + \beta \qquad \text{When } R_{ij} > R_h \qquad (3.3)$$

For values with low tendency:

$$\frac{R_h}{R_{ij}} \qquad \text{When } R_{ij} > R_h \qquad (3.4)$$

$$\frac{R_h - R_{ij}}{R_h - R_l} + \alpha \qquad \text{When } R_l \leq R_{ij} \leq R_h \qquad (3.5)$$

$$\frac{R_{\min}}{R_{ij}} + \beta \qquad \text{When } R_{ij} < R_l \qquad (3.6)$$

Where,

- R_{ij}, is the value of i^{th} QoS property of j^{th} service.
- R_l, is the lower limit of consumer requirement for an attribute.
- R_h, is the higher limit of consumer requirement for an attribute.
- R_{max}, is the maximum value of a QoS property offered by the services under consideration.

- R_{min}, is the minimum value of a QoS property offered by the services under consideration.

- α and $\beta \in \{1, 2, 3 \ldots\}$ where $\alpha < \beta$.

Algorithm: Partial Service Ranking and Recommendation based on QoS properties

Input: CR (Consumer Requirements), SL (Service List)

For all service s $\in$ {1, 2, 3 … m} in SL **do**

If (s satisfies mandatory consumer requirements)

Add s to CL (Candidate List)

// Candidate List CL is a list of all relevant services

end if

end for

For all service s in CL

For all QoS parameter q

Find Minimum and Maximum value of q from SL

//Min and Max will be used in formation of accuracy matrix.

end for

end for

For all service s in CL

Calculate normalised value of QoS property

// normalized values will be calculated using equation 1-6

end for

For all service s in in CL

Calculate total score for each service s

$$R_s = \sum_{j=1}^{n} A_{ls} * W_s$$

End for

Based on the total score Rank all services order services in CL

Return CL

Algorithm 3.1: Service recommendation and ranking algorithm

Figure 3.3 shows the three different ranges of interest: preferred, tight and loose. Figure 3.3a shows increasing of values for high tendency QoS properties, while Figure3.3b shows decreasing of values for low tendency QoS properties.

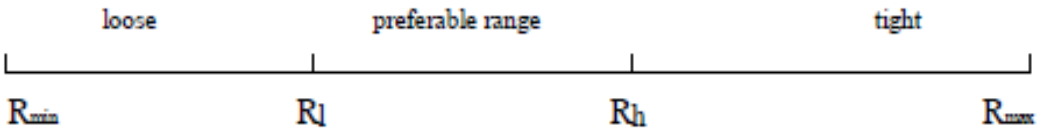


Figure 3.3a QoS property measurements- high tendency

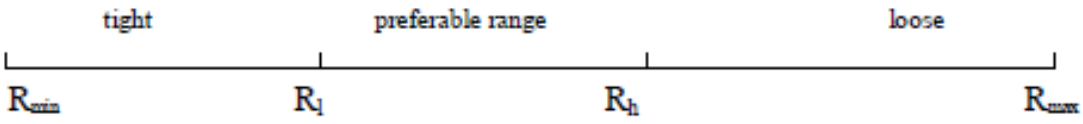


Figure 3.3b QoS property measurement- low tendency

The equations (3.1)-(3.6) generate results which are normalised in the range 0 to 1. The constants α and β are introduced in order to discriminate between the three ranges. For results in the loose range, the value remains between 0 and 1. For results in the preferable range, α is added to the equation and the results are in the range α to (α+1). For results in the tight range, β is added and the results are in the range β to (β+1).

For consistency, we have normalized values of QoS properties to be in a small range using equation (3.1-3.6), in which all the values in the accuracy matrix lies in the range of (0, β+1). In the selection process there are thousands of services, the main purpose of the algorithm is to arrange all the QoS values for services based on minimum and maximum values , and arrange the values between (0, β+1).

On the other hand, every value in the algorithm is considered precisely based on the range. Considering a value for availability; let's assume preferable range is (80% - 90%). Any service having availability value less than 80% will be counted as loose, those between 80%-90% will be counted as preferable range and α will be added, finally any value larger than 90% will be counted as tight and β will be added. Since $0<\alpha<\beta$ it guarantees that the higher range always has higher value than the other two ranges.

The results of the calculations are used to populate the accuracy matrix, A. The matrix shows how accurately each advertised service matches the overall consumer requirement without yet considering the desired preference of each attribute.

3.3.4 Weight Calculation

Collecting weight values from service consumers and calculating weight values using arithmetic and geometric methods might not be an efficient or understandable choice. They are not the most suitable choice for complex selection process with tens or hundreds of consumer requirements with higher weights. Dependency between parties is always considered important. QoS properties have different values and there are dependencies between these parties. Values in QoS properties might affect each other, which will directly affect performance and accuracy of the service selection process.

The work considers these dependencies to determine weighting value of QoS properties using distance correlation between QoS properties. Distance correlation can detect different types of dependencies, the value will be one when the two QoS properties are totally dependent on each other, and zero when the two QoS properties are statistically independent.

The distance correlation for a sample (X,Y)={(X_k, Y_k): k=1…p} from the pairwise distribution of random vectors X and Y defined as follows:

$$a_{jk} = \| X_j - X_k \| \ , \ b_{jk} = \| Y_j - Y_k \| \quad \text{j,k=1,2,..,p}$$

It is worth to note that $| \ |_p$ denotes the Eculidean norm, similarly,

$$A_{j,k} := a_{j,k} - \bar{a}_{j.} - \bar{a}_{.k} - \bar{a}_{..} , \quad B_{j,k} := b_{j,k} - \bar{b}_{j.} - \bar{b}_{.k} - \bar{b}_{..}$$

Where $\bar{a}_{j.}$ is the mean of j^{th} row, and $\bar{a}_{.k}$ represents k^{th} column mean, and $\bar{a}_{..}$ represents grand mean of the distance matrix of the X, it is the same for the b values.

The distance covariance of the two distributions X and Y is computed as follows:

$$dCov_p^{\ 2}(X,Y) := \frac{1}{p^2} \sum_{j,k=1}^{p} A_{j,k} * B_{j,k} \qquad (3.7)$$

Finally, with the equation below the distance correlation between the two distributions X and Y is calculated as following:

$$dCor(X,\mathrm{Y}) = \frac{dCov(X,Y)}{\sqrt{dCov(X) * dCov(Y)}} \tag{3.8}$$

With computing pairwise distance correlations for different QoS attributes, it produces the following dependency matrix, the overall dependency of QoS_i with respect to other quality attributes is determined as follows:

$$dep(QoS_i) = \frac{1}{p}\sum_{j=1}^{p} dCor_{ij} \tag{3.9}$$

The weight of each quality attributes is then derived from the dependency value as follows:

$$W(QoS_i) = \frac{dep(QoS_i)}{\sum_{i=1}^{m} dep(QoS_i)} \tag{3.10}$$

After the A matrix is fully populated, the rank of each service is calculated by summation of the QoS attribute and weight product for that service using (3.11).

$$R_i = \sum_{j=1}^{n} A_{ij} * W_j \tag{3.11}$$

Where R_i represents rank of service i, A_{ij} represents the accuracy value of j^{th} QoS property of service i, and W_j represents weight of j^{th} QoS property.

3.4 Example Scenario

We are taking a real world example and going through it step by step to clearly demonstrate the service ranking algorithm.

The plan is to use emergency support service. For the sake of simplicity we assume there are three QoS requirements as in Table 3.1.

Number	QoS Requirement	Preferred value
1	Distance	7-32 km
2	Availability	80-90 %
3	Reliability	70-85 %

Table 3.1 Consumer quality of service requirements

Assume the list of available emergency support services available are as below in Table 3.2,

No.	Service Name	Distance (km)	Availability (%)	Reliability (%)
S_1	EmergencySupport	4.93	42	73
S_2	PluralEmergency	64.5	86	85
S_3	FastEmergency	10.3	85	90
S_4	GlobalEmergency	28.5	90	70
S_5	countyEmergency	50	97	73

Table 3.2 List of available web services

By using equation (3.1)-(3.6) we can calculate QoS property values for each service as shown in Table 3.2. According to the emergency case preferred values, S_1 does not comply with all QoS requirements, S_1 value for Availability is 42% which is lower than the preferred range, and this is true for both S_2 and S_5. Since none of the QoS requirements is mandatory, they can be considered in the selection process.

No.	Service Name	Distance (km)	Availability (%)	Reliability (%)
S_1	EmergencySupport	3	0.525	1.2
S_2	PluralEmergency	0.496	1.6	2
S_3	FastEmergency	1.868	1.5	3
S_4	GlobalEmergency	1.14	2	1
S_5	countyEmergency	0.64	3	1.2

Table 3.3.3 Values of QoS for each candidate service

Based on the results from Table 3.3, scores of each service can be calculated by using Equation 3.11. It is considered the corresponding weight value is given for the request, and the weight value for the three QoS requirements are assumed to be {5, 3, 2} respectively.

Serivces	Final Score
S_1	18.975
S_2	11.28
S_3	19.84
S_4	13.7
S_5	14.6

Table 3.3.4 Rank of each service

From the table 3.4 it can be understood that services returned to the consumer are ranked as $S_3 > S_1 > S_5 > S_4 > S_2$. Service S_3 is having a distance of 10.3km with Availability of 86% and Reliability of 90%. While S1 has value of 4.93km for distance which is better than distance value of S3, but S1 availability value is 42% and Reliability is 73%. Based on our approach by considering the overall criteria, S_3 is the best suited to the consumer request.

3.5 Experimental Results and Evaluations

Experiment Setup: In this section experimental results are shown for the algorithm. For the experiment a large scale dataset, WS_DREAM, has been used. The WS-DREAM dataset three [107] is a large scale dataset which has more than 4500 real world services. Each service was invoked by 142 service consumers in 64 different time intervals. Each service has different QoS properties like Response time and

Throughput. Based on the QoS properties service ratings is calculated by using multi-attribute utility function [108].

The tests in this chapter all carried out on a HP ProBook laptop corei5 2.50GHz, with 4GB of RAM on Windows 7 Enterprise. In the evaluation a set of experiments were conducted. Experiments are to evaluate the algorithm based on accuracy and performance.

3.5.1 Performance Measurement

3.5.1.1 Discount Cumulative Gain (DCG)

In the proposed algorithm the rank of services were calculated based on QoS properties. To measure the performance the algorithm was compared to proposed algorithm in [66] which used the probabilistic flooding-based method, by combining Simple Additive Weighting (SAW) technique and Skyline filtering. In addition it was compared to SPSE algorithm, which uses the Pareto optimal-based solution method [67]. In the experiment, a real world services were used.

The DCG method was used, which is a common method for information retrieval and quality ranking [109]. The method calculates the quality of services depending on their rank in the list; the gain is accumulated at the upper ranks and discounted in the lower ranks. Services with better quality should be shown earlier in the ranking method.

The equation is as follows,

$$DCG_p = \sum_{i=1}^{p} \frac{2^{Q_i} - 1}{\log(1+R_i)} \qquad (3.12)$$

Where Q_i is the QoS for the i^{th} service, and R_i is the rank of i^{th} service. The higher the DCG_P value the higher the QoS parameters of the Top-P service. The experiment compared the ranking value of the top 50 services and is summarised in increments of 10.

Top P	Acc. Matrix	SPSE	SAW
5	7.569091655	4.646180572	3.0730903
10	10.38841135	6.38846729	3.9442336
15	12.71343601	8.372440175	4.9362201
20	14.89674501	10.34730971	5.9236549
25	16.85575592	12.17659888	6.8382994
30	18.62296682	14.09352166	8.1967608
35	20.40988547	15.89264909	10.496325
40	21.8047093	17.3287577	11.864379
45	23.0047093	19.6287577	13.314379
50	24.8047093	22.1287577	16.514379

Table 3.5 DCG ranking for accuracy matrix, SPSE, and SAW

Table 3.5 shows the results of DCG for top-50 services returned for the selection process. From the results we can see the quality matrices of Accuracy Matrix top-P services are higher and in many cases, nearly double to SPSE and SAW results. For example in the analysis category, the top 5 services for Accuracy Matrix algorithm is 7.569 while for SPSE is 4.646, and for SAW is 3.073. It is clear the top 5 services are the most important services to the consumer, since most of the times consumer selects services from the top 5 candidates. By examining the returned results it can be understood that the proposed algorithm returns higher quality services to the consumers and provide more satisfactory results to their requirements, this is due to the high accuracy of our approach in which each QoS property plays important role in the ranking and recommendation process.

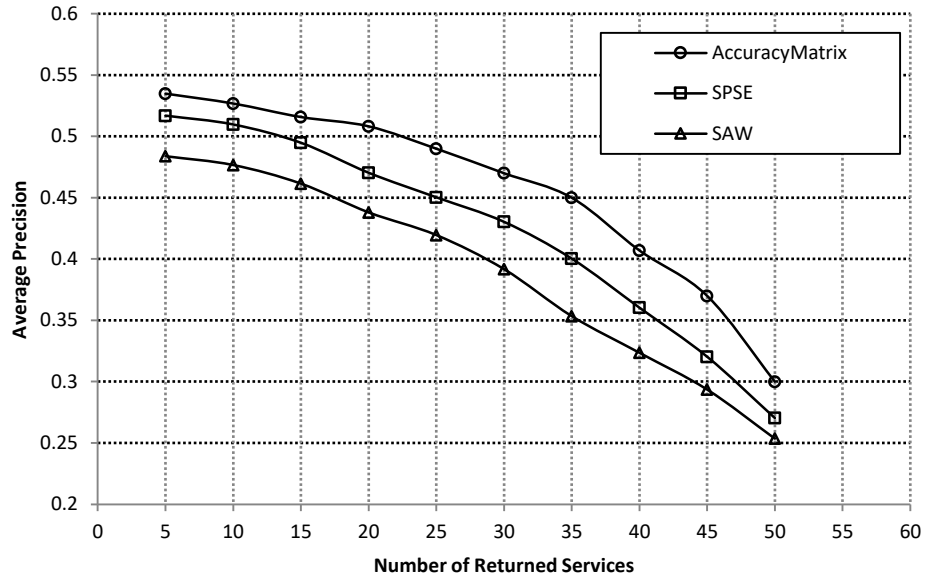
3.5.1.2 Precision and Recall

The precision and recall are standard measurements that have been used in information retrieval for measuring the accuracy of a discovery or recommendation method or a discovery engine. The precision is defined as the ratio of the number of returned correct services to the total number of all returned services [110]. By contrast, recall is defined as the ratio of the number of returned correct serviced to the number of all correct services as shown in equation (3.12).

$$\text{Prec} = \frac{Rtrel}{\text{Re}\,l}, \text{Rcal} = \frac{Rtrel}{Rt}$$

Where Prec represents precision, Rtrel represents a list of returned relevant services, Rel represents relevant services, and Rt represents returned services.

To assess the method, we categorized the services into different categories based on the functionalities they provide. From one category we randomly selected a set of 60 services with different QoS properties, of which 40 services were relevant to the query. We conducted 10 different tests each of them repeated 100 times. In the first test no service had unmatched properties, in the second test at least one of the properties was partial match to the service query. The numbers of partial match properties were



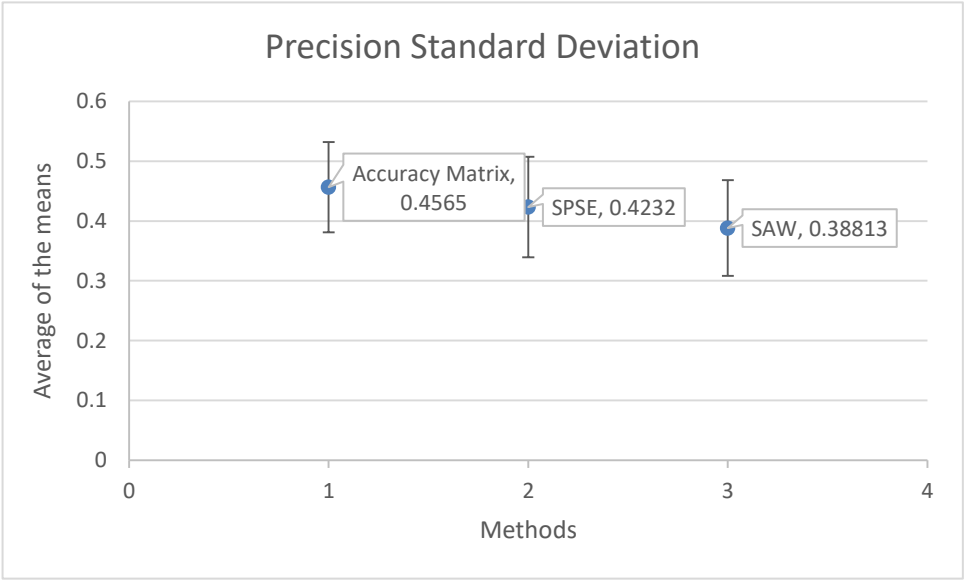
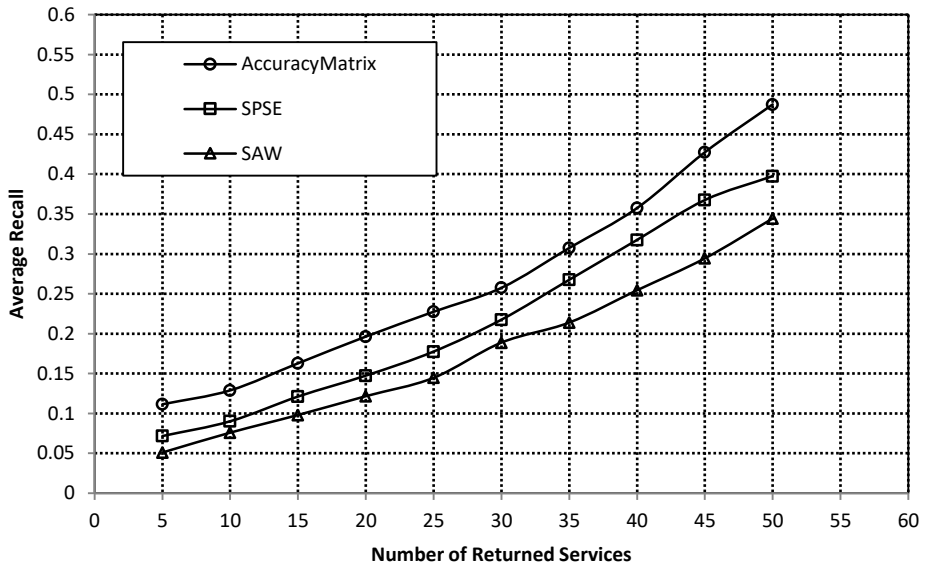


Figure 3.4 Comparison of precision between SPSE, SAW, and Accuracy matrix



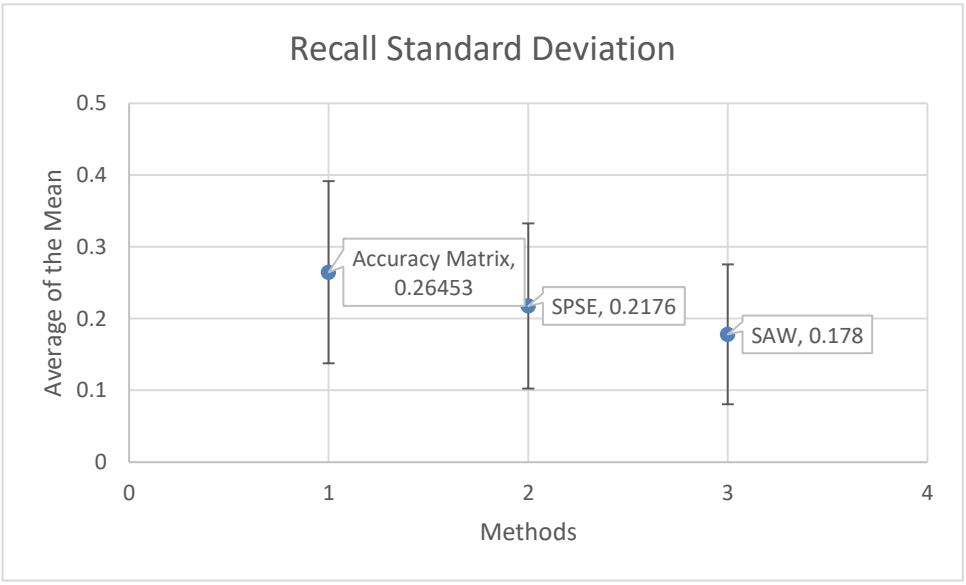


Figure 3.5 Comparison of recall between SPSE, SAW, and Accuracy matrix

increased through remaining tests. In Figure 3.4 it can be depicted that the precision is affected by the number of returned services, it decreases as the number of returned services increase. On the other side, the recall increases by increasing number of

returned services as shown in Figure 3.5. From Figure 3.4 and Figure 3.5 it is apparent that the Accuracy Matrix algorithm improved both precision and recall, the percentage of relevant services in the Accuracy Matrix is higher than SPSE and SAW. The improvement is due to the inclusion of partially matched services in the Accuracy Matrix algorithm. This effect is very clearly observed when the recall for SPSE and SAW do not reach as high as our algorithm; it is because of the limitation of service matchmaking in both algorithms by exclusion of partially matched services. This confirms that our approach deals more accurately with the service query, and returns relevant service to the consumer even in the case where the candidate services only partially match the consumer requirements.

3.5.2 Scalability

In this part of the experiment we evaluate the scalability of our algorithm with increasing number of services. The test focused on selection time with increase in number of services, while number of QoS parameters fixed. A real world dataset were used, which had up to 4500 services. The selection time was measured while increasing the number of services.

From results in Figure 3.6, it is clear that the algorithm linearly increases time with an increasing number of services, and it is also clear from the figure that SPSE method had marginally better performance comparing to the our method, this is due to the use of automatic weighting technique to estimate values of consumers preferences, and also the increase in the number of candidate services in the selection process. For example, with 1000 services, our algorithm returned 56 services in 12.2 seconds, while SPSE returned 36 services in 10.6 seconds. The figure also shows the direct

proportionality between execution time and services considered. The slope of this linearity strongly implies that our approach is rather scalable with increasing number of services.

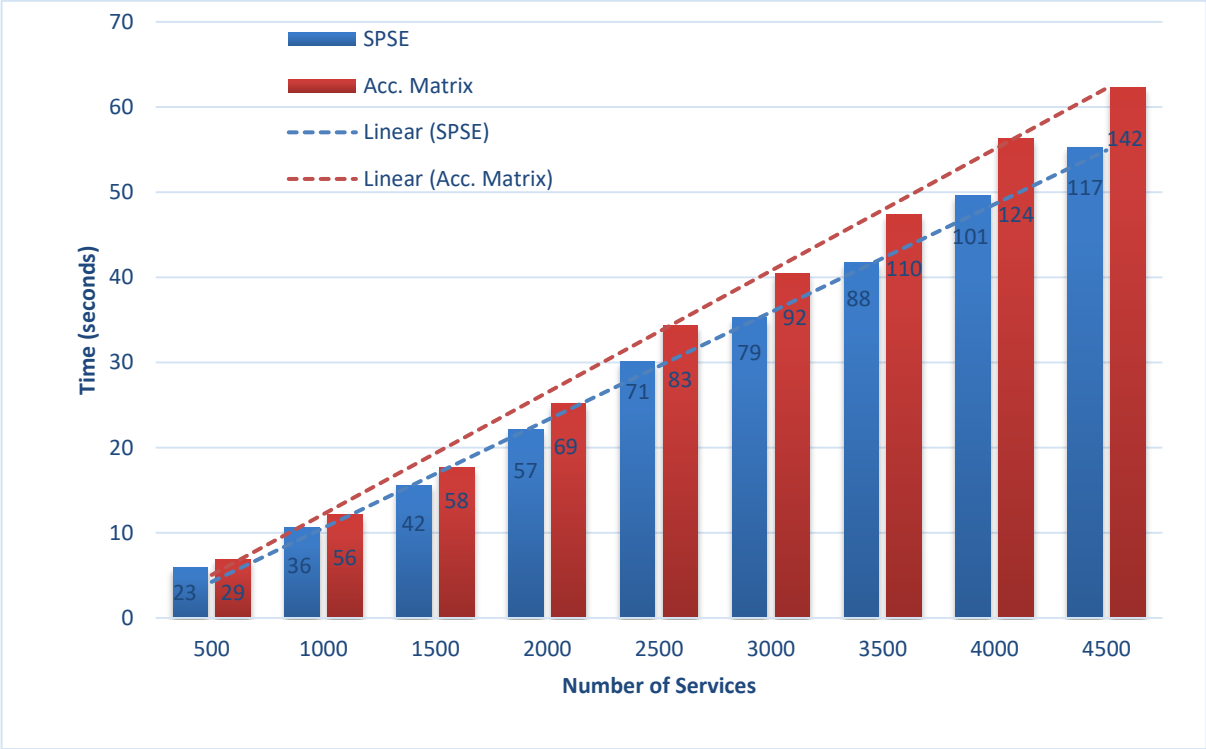


Figure 3.6 Performance evaluation with fixed number of QoS parameters, and number of returned services for both methods

3.6 Summary

In this work, we proposed an efficient algorithm to service selection in IoT environment. By considering all QoS properties, the algorithm includes consumer preference and allows the consumer to select the best available service for their task. The design of an accurate ranking algorithm for services based on matching consumer requirements identifies those services which are candidates for use in the selection process.

When an initial list of services is returned which only partially matches with optional consumer requirements, our algorithm includes them in the decision and recommendation process. This holds true even in the case where the optional attributes barely match. This action allows more services to be included and accurately ranked, providing the best choice to the consumer. Experimental results show that in extreme circumstances, our algorithm recommends services when other methods would return an empty list. This feature can be used and applied to other service applications such as service discovery and composition.

The solution is generic and can deal with multiple QoS properties and large numbers of services. Experimental results show a significant improvement in the number of relevant services recommended by our algorithm when compared to the SPSE algorithm. The scalability was tested by increasing number of services and QoS parameters. There was no significant performance degradation when reducing large datasets with multiple QoS attributes. Our experiments rendered results that strongly support its use as a tool for service selection and recommendation processing.

This chapter explained the detail of partially matched service selection method which helps consumers to select the best available service to their requirements. In the next chapter correlation and content analysis of services will be used in a personalized service recommendation and selection approach.

Chapter 4 Personalized Service Recommendation

4.1 Introduction

Web services are self-contained, self-describing, loosely coupled, modular software applications that can be published, found, and invoked across the web [111]. \the number of web services is continuously increasing and it is becoming one of the standard technologies for software and data sharing. The rapid increase in the number of web services with similar functionality and different Quality of Service (QoS) properties requires a novel and efficient approach for recommendation and selection of web services to satisfy consumer requirements.

Web service recommendation can be considered an automatic process of discovering and recommending a number of web services to the consumer based on their QoS properties and the behaviour of the consumer. Collaborative filtering [5] [6] [7] [8]is used as a significant approach in the process of service recommendation by exploiting consumers and services interactions with the system. There are two types of collaborative filtering approaches, user-based and item-based approaches [12]. Additionally, a hybrid approach has also been proposed [7] [8] by combining both user-based and item-based approaches.

All the current collaborative filtering approaches make recommendations based on mutual relationships between user and services, but neglect the latent interests of the users and services.

The latent interest from the user can be considered as a requirement of the user which will avoid the user providing his/her requirements as it is always prone to error. Whereas latent interests of the service reduces the search dimension by describing each service by a list of latent interests (topics), and those topics represent the behaviour of the service. Another problem is that well-known approaches did not take the relationship among neighbouring services into consideration during process of service recommendation; this correlation has a big impact on improving the accuracy of predicting missing QoS values.

In the process of service recommendation there are three main requirements needed to conduct an effective service recommendation task which is considered as motivation for our approach:

- **Accuracy:** A good recommendation system should recommend more favourite Web services and fewer disliked ones, particularly in the situations where available information might be not sufficient (e.g. missing QoS of some services).
- **Fairness:** Recommending services that are well-known to a user is often found unsatisfactory or meaningless. If the recommended services are unfamiliar to a user, the chances of finding new Web services that match the user's requirements would increase.
- **Cold-start problem:** Solving this problem not only enables users to find newly-deployed Web services, but also enhances the recommendation diversity.

In this chapter, we propose a hybrid approach for Web service recommendation by combining correlation-based and semantic content-based recommendation. With correlation-based, relationships among neighbouring web services is calculated based on the list of users that invoked both services, and the value was used to generate a correlation matrix to improve QoS prediction accuracy. On the other hand, with the semantic content-based method, latent interests have been extracted by using Latent Drichlet Allocation (LDA), based on the latent interests each web service is represented by a list of topics and each user requirement is viewed by a list of topics, based on the service-topic-users relationship the next possible interest of the user is predicted and list of recommended services is returned to the user. Our approach exploits the advantages of both techniques by proposing a hybrid method that considers both correlation and content information of Web services. The main contribution of the work can be stated as follows:

- Proposing a hybrid approach for web service recommendation that combines service correlation-based and content-based recommendation of services.
- Unlike other conventional service recommendation approaches, our approach considers correlation between web services based on co-invocation by different consumers, and exploits the LDA (Latent Dirichlet Allocation) model that simultaneously considers the similarities of users and content of web services and extracts latent interests of users and services.
- Next possible user preferences can be statistically estimated by using a list of user's latent interests.
- Extensive experiments were performed by using real-world Web services to verify the proposed method. A dataset consisting of 5,825 web services were

examined and 2,675 live Web services were selected and utilized in the experiments. The experimental results show that our approach achieves better recommendation performance than the conventional correlation and content-based methods.

The rest of the chapter is organized as follows, section 4.2 provides an outlook on the related work in the area, and section 4.3 gives the detail of the proposed hybrid method. Section 4.4 shows the experimental results, and finally section 4.5 concludes the paper.

4.2 Hybrid Web Service Recommendation

In this section, we discuss the extension of the recommendation process by utilizing correlation-based and content-based recommendation. In the first step we discuss the correlation-based recommendation of Web services. With the correlation, the relationship between web services can be acquired by utilizing the number of co-invoked web services by different service consumers which represents service dependant preference. Based on the service correlation the Top-k neighbours are used in predicting QoS properties. In the second step, content-based recommendation is discussed. From the content of web services latent information about services is extracted by using inference of the LDA model. Two matrices are formed to predict missing QoS attributes for recommendation processes based on the relationship

between the latent topics of services and the latent interests of users. Finally, by combining correlation-based and content-based a hybrid method is formed, the method is used to rank candidate services and return the Top-K recommended services.

4.2.1- Correlation based recommendation

Some of the well-known approaches ignore relationships among Web services in similar neighbour selection which have an important impact on QoS prediction accuracy. In this approach, which is based on our previous work [112] we used a correlation-based top-*k* recommendation approach for Web services to improve the process of finding *k* neighbours by using correlation between Web services instead of the similarity.

The relationship between services can be achieved by using the number of co-invoked web services by different consumers. The preferred dependency between services can be defined as;

$$dep_{i,j} = \begin{Bmatrix} \{u_k : (qos_{k,i} \neq \varnothing) \wedge (qos_{k,j} \neq \varnothing)\} & (i \neq j) \\ \varnothing & (i = j) \end{Bmatrix} \quad (4.1)$$

$$i = 1,...,N;\ j = 1,...,N;\ k = 1,...,\mathrm{M};$$

$$DEP_{i,j} = | dep_{i,j} | \quad (4.2)$$

Where $dep_{i,j}$ is a list of service consumer who have invoked both i^{th} and j^{th} service. $qos_{k,i}$ and $qos_{k,j}$ are lists of QoS parameters for both *i* and *j* service which has been invoked by user *k*, *M* and *N* are the number of users and services consequently. Finally, $DEP_{i,j}$ is a *N*N* matrix, where each entry is a $dep_{i,j}$ value.

In addition, the considered correlation of services is not only affected by co-invoked service consumers, but also by the other service consumers (i.e. the correlation of $DEP_{i,j}$ should be different from the correlation of $DEP_{j,i}$. As a result the *DEP* matrix can be enhanced using the following equation

$$Cr_{i,j} = \begin{cases} \dfrac{|dep_{i,j}|}{\sum_{l \in N} |dep_{i,j}|} & i \neq j \\ 0 & i = j \end{cases} \qquad (4.3)$$

Where *Cr* is a matrix of *N***N* elements which is derived after enhancement of DEP matrix. *Cr* shows a stochastic matrix where every entry represents the correlation value between two services. Based on the correlation value for service a correlation graph G [113] will be constructed as shown in Figure 4.1, the nodes of the graph show the services and the connection between them shows the correlation value between services. The connection between nodes are bidirectional and built based on *Cr* matrix, which shows different values of correlation between a pair of services in each direction. The service correlation can propagate through the graph, and the flow of relation of web services can be used to transfer neighbours of high QoS services to good quality services.

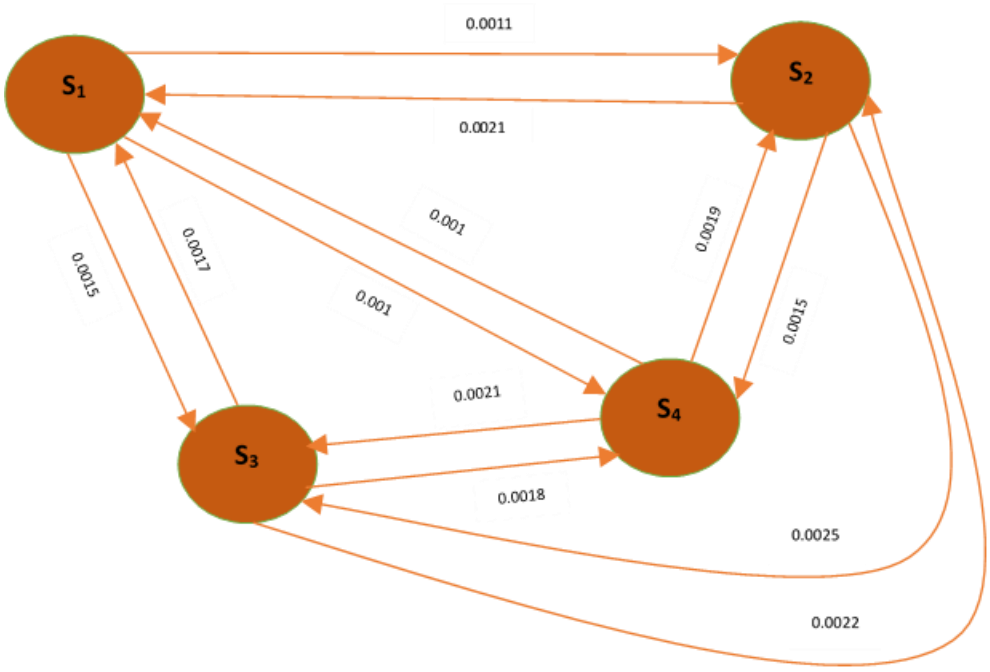


Figure 4.1 Sample of Correlation Graph for Service Recommendation

The characteristics in the process of service correlation propagation graph are similar to PageRank [114], which has two important characteristics, propagation and attenuation. In the scenario of service recommendation, important scores can be considered as the rank for Web service *n*. Thus, PageRank equation can be extended by the following equation with Biased PageRank as;

$$SRM_{k,j} = c. \sum_{q \in E(n_j)} SR(q) Cr(q, n_j) + (1-c).d_{u_k} \qquad (4.4)$$

Where $SRM_{k,j}$ is the rank of Web service j for user k and $Cr(q, n_j)$ is the correlation value that service q is connected to service n_j in matrix Cr. $E(n_j)$ is a set of services which are connected to service n_j directly. Vector du can be tuned in order to bias the PageRank by boosting nodes corresponding to high value entries and matric Cr controls the propagation and attenuation mode. Then, d_u is a vector for user u in which

non-negative entries have summed up to 1. In addition *c* is a decay factor which is usually set to 0.85. As in Table 4.1 a sample of service rank was shown, as it is clear different consumers have different ranks for the services. Each entry in the table shows the rank of a service for corresponding service consumer. Interested readers can refer to the previous work in [112] for the detail of the correlation graph recommendation, and the ranking matrix.

Users	**Web services**				
	S_1	**S_2**	**S_3**	**S_4**	**S_5**
U_1	0.00065	0.00105	0.00055	0.00081	0.00146
U_2	0.00069	0.00098	0.00049	0.00086	0.00155
U_3	0.00076	0.00109	0.00056	0.00079	0.00153
U_4	0.00071	0.00101	0.00059	0.00083	0.00148

Table 4.1Sample of Ranked Services

4.2.2 Service Content Extraction

Every web service has a WSDL (Web Service Description Language) which contains the description of the web service. WSDL is a XML-based language, which describes web service based on the W3C (World Wide Web Consortium) standard. Each web service has two types of descriptions, first one is WSDL description which describes "how" a service should be used, and the second one is the free text which describes "what" the service does.

The process of content extraction starts by checking the availability of the web service and validating it is content. And then getting the WSDL file directly from the URI and read the content of the file. The following are several steps in the process of content extraction;

- **Content extraction and Tokenization:** In this step all important features will be extracted from WSDL file, such as name, inputs, outpts, documentation, messages, and operations. Based on the extracted contents each service can be represented by a list of descriptors which are called tokens. $D_s=\{t_1,t_2,\ldots,t_n\}$ where t_i is token i in service s. Some tokens usually consist of a sequence of words which need to be handled. The first letter of each word in the sequence is capitalized (e.g., getItemPrice, or findAddressByPostcode). As a result the descriptors are separated into different tokens using regular expressions (e.g., get, Item, Price, find, Address, By, Postcode).
- **Removing Stop Words:** In this step all words are removed which are considered as stopwords and do not have substantial semantics. Example of stopwords are, WSDL related like SOAP, type, element, binding, get, set, and etc., and all the other stopwords like a, the, what, where, and etc. for the stopword removal the Standford POS Tagger4 has been used which removed all stopwords and left only nouns, adjectives, and verbs.
- **Analysis and Matrix Construction:** After extraction of the content of the web services we calculate the frequency of each term for each web service. TF/IDF (Term Frequency/Inverse Document Frequency) [115] algorithm has been used to represent the corpus of WSDL documents. The TF/IDF is a common mechanism in Information Retrieval (IR) for generating a set of

robust representative terms for a corpus of documents [115] . We use TF/IDF to convert WSDL document to Vector Space Model (VSM), which is a technique to represent each web service as a vector of terms and form service matrix. Each row of the matrix represents a service description, and each column represents a term from the whole corpus, and each entry of the matrix represents the frequency of the item in that particular service. The TF/IDF equation: $W_{ij}=tf_{ij} * log(n\ n_j)$,

where, W_{ij}, is the weight of the term j in the service document i,

T_{ij} is the frequency of term j in the service i,

n, is the total number of service documents in the corpus,

n_j, is the total number of services contain that term j.

4.2.3 Content-based Recommendation

Content-based service recommendation is based on the analysis of the similarities of the content (e.g., WSDLs and short descriptions) between services. There have been two main approaches in content-based Web services recommendation: *syntactic* based approaches and *semantic* based approaches. We discuss only semantic based approaches in this paper since syntactic based approaches have limitations in suggesting high quality recommendations.

The semantics of a Web service s can be represented by a set of semantic attributes: i) functional category $F(s)$, ii) functional parameters (i.e., inputs $IP(s)$ and outputs $OP(s)$), and iii) requirements (i.e., preconditions $P(s)$ and effects $E(s)$). We assume that these attributes can be provided by domain ontology through semantic

annotations, which will ensure to provide users with recommendations that are semantically similar to Web services previously invoked. It is possible to construct domain ontology by analysing Web service descriptions (WSDLs and free text descriptors).

Service recommendation aims at providing services to consumers which matches their requirements. The list of functionally equivalent services is recommended to the consumer and ordered based on their QoS properties. In our approach for content based recommendation, inspired by work in [84] we use Latent Dirichlet Allocation (LDA), which is a probabilistic topic model uses a generative probabilistic model for collections of discrete data [68], to extract latent interests (topics) from service descriptions.

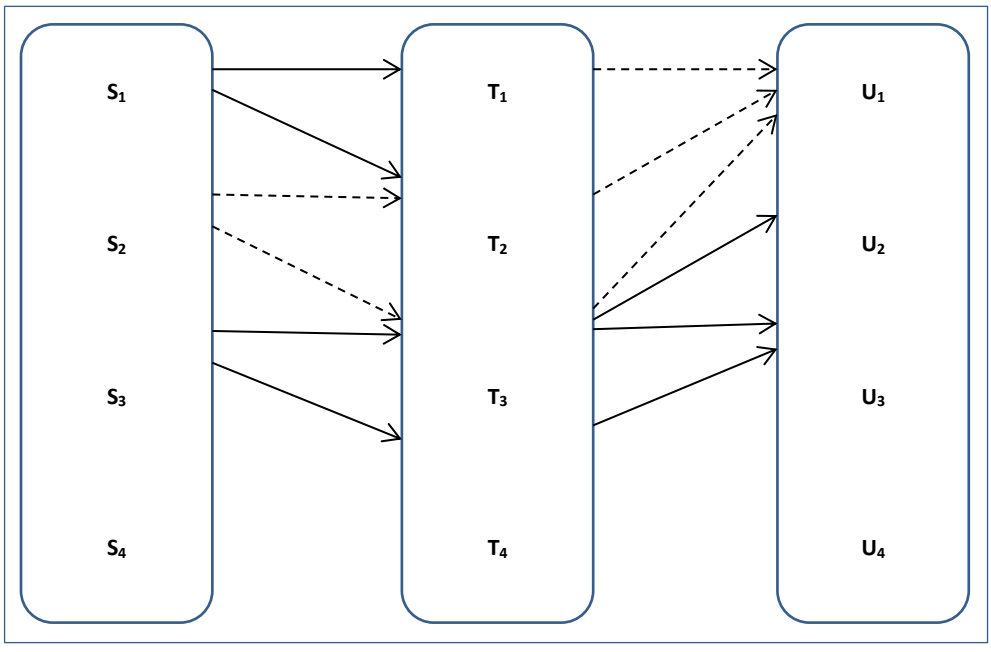


Figure 4.2 Service-Topics-User representation

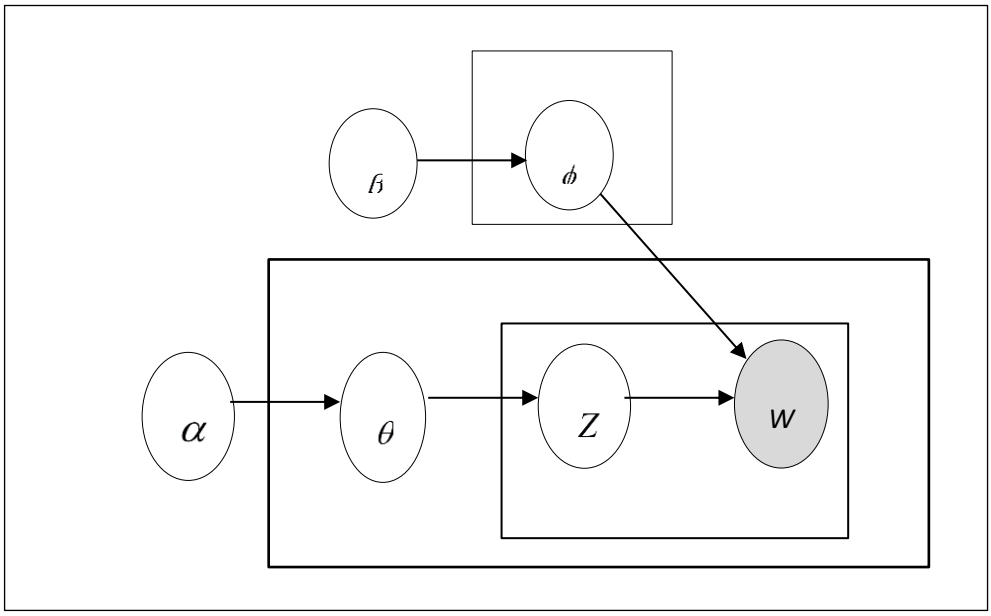


Figure 4.3 LDA Graphical Model Representation

In the recommender system model ratings of services depends on an underlying set of topics. The model has a three layer representation, service-topics-users, as it is shown in Figure 4.2. Each service is represented as a probability distribution over topics and each topic is a probability distribution over users.

Topic modelling is a type of statistical modelling, which is used to find hidden topics (football, weather, holiday, and etc.) that occur in a collection of documents, and it was initially proposed in machine learning and natural language. It is important to distinguish LDA from a simple Dirichlet-multinomial topic model, LDA involves

three levels, and notably the topic node is sampled repeatedly within the document. Under this model, documents can be associated with multiple topics.

The LDA model can be represented by a probabilistic graphical model as depicted in Figure 4.3. The grey shaded variable represents observed variables while the unshaded variables are latent variables. Each service s_i from the list of services $S= \{s_1,s_2,s_3,\ldots, s_m\}$, is represented as a bag of words W, $W=\{w_1,w_2,w_3, \ldots, w_n\}$, each word w_j is seen as an observed variable. Observed variables are generated from multinomial distribution over ϕ_t specific to topic *t*, and topic *t* is selected from a multinomial distribution over topics θ_i specific to the service. Both ϕ and θ are put by the Dirichlet distribution with hyper-parameters β and α respectively.

The principle of LDA is mapping high-dimensional count vectors to a lower dimensional representation in latent semantic space. Each word *w* in a service description *s* is generated by sampling a topic *z* from topic distribution, and then sampling a word from topic-word distribution. The probability of the i^{th} word occurring in a given service is given by Equation 4.5:

$$P(w_i) = \sum_{f=1}^{k} P(w_i \mid z_i = f) P(z_i = f) \qquad (4.5)$$

Where z_i is a latent factor (or topic) from which the i^{th} word was drawn, $P(z_i = f)$ is the probability of topic *f* being the topic from which w_i was drawn, and $P(w_i|z_i = f)$ is the probability of having word w_i given the f^{th} topic.

A k-dimensional Dirichlet random variable θ can take values in the (k −1)-simplex (a k-vector θ lies in the (k−1)-simplex if $\theta_i \geq 0$, $\sum_{i=1}^{k} \theta_i = 1$), and has the following probability density on this simplex:

$$p(\theta, z, w \mid \alpha, \beta) = p(\theta \mid \alpha) \prod_{n=1}^{N} p(z_n \mid \theta)\, p(w_n \mid z_n, \beta) \qquad (4.6)$$

Where $p(z_n \mid \theta)$ is simply θ_i for the unique i such that z_i n = 1. Integrating over θ and summing over z. The parameters α and β are corpus level parameters, assumed to be sampled once in the process of generating a corpus.

Extracting service topics from LDA is an inference process for latent variables, which is to reverse the generative model and generates latent variables from provided observations. After the inference process the value of latent variables should maximize the posterior distribution of the service observations. There are many approximation methods, Gibbs Sampling [9], variational inference [8], and others. In this paper Gibbs Sampling technique has been chosen as it is easy to implement and provides an efficient method for extracting a set of topics from a large rating set.

4.2.4 Hybrid Method and Ranking of Recommended Services

As mentioned before, correlation-based and content-based approaches can complement each other very well to analyse user-service QoS rating matrices and recommend services to users. Accordingly, in this subsection we integrate the two types of methods into our hybrid prediction model, which is defined based on Equation (4.1) and Equation (4.3), as shown in the formula 4.4. After obtaining the

predicted values for those missing elements in the user-service QoS rating matrix, personalized Web service recommendation can be easily performed based on the complete matrix. According to a given target user's non-functional requirements or preference on QoS rating, all candidate Web services are sorted in a certain order. For example, the recommender arranges the values of candidate Web services in ascending order if the target user focuses on response time, while it will return the results in descending order when considering availability. Eventually, the top K Web services in a sorted list with respect to QoS rating are recommended to the target user.

$$rank(services) = \alpha * correlation(services) + (1-\alpha) * contents(services) \quad (4.7)$$

In the model, the customization parameter α is in the interval of [0, 1] for model weight adjustment that can adapt to different application scenarios.

4.3 Experimental Results

In the experiment section, to increase the reliability of the results a large scale dataset is required, however constructing such a dataset is time consuming and beyond the scope of our research. Fortunately, there is a piece of work in the WS-DREAM project, which is a large scale real-world web service dataset provided by Zheng et al [6] [12]. With WS-DREAM, web crawler engine, all available web service WSDL files were crawled from the internet. Besides the services some QoS parameters were observed by 339 distributed computers in 30 different countries in Planet-Lab with more than 1.97 million invocation records. The dataset was used in the experiment after performing the following preparations,

1- All 5,825 WSDL addresses were traversed from the dataset; by using a crawler software 2,675 live services were retrieved. After that, as explained in service content extraction section, service descriptions are extracted and used to determine the topics/interests describing web services. For the LDA implementation we used JGibbLDA [116] [117], which is a Java implementation of Latent Dirichlet Allocation (LDA) using Gibbs Sampling technique for parameter estimation and inference. For hyperparameters of α and β we used the same value from previous works [118] [83] which have found that α =50/*T* where *T* is the number of topics and the default value for β =0.1. All services rating scores were determined from 1 to 5 depending on their QoS parameters (e.g., response time, throughput) using a multi-attribute utility function.

2- To perform the QoS prediction performance more sufficiently, we divide the 399*2,675 matrix into different sub matrices by randomly choosing web services from 399*2675 matrix to construct new sub matrices with different numbers of services *NS*. A consumer usually invokes a small number of services, to simulate a real world dataset distribution we randomly remove entries from the sub matrices by using different *density* parameters (0<density≤1). For each sparse matrix we separate into two parts, training set (80% of the records in the matrix) and the test set (the remaining 20% records in the matrix, the values in the training set will be used to predict missing values in the test set. Furthermore, a parameter k is used to denote the number of neighbours for web services.

The tests in this paper all carried out on a HP ProBook laptop corei5 2.50GHz, with 4GB of RAM on Windows 7 Enterprise. In the evaluation a set of experiments were

conducted. Experiments are to evaluate the algorithm based on prediction accuracy and recommendation of services to consumer.

4.3.1 Evaluation Matrics

In this paper, Mean Absolute Error (MAE) and Normalized Mean Absolute Error (NMAE) metrics [12], [119] are used to measure the prediction accuracy of our approach. The definitions of the metrics are defined as:

$$MAE = \frac{\sum_{u_k,i} | r_{u_k,i} - \hat{r}_{u_k,i} |}{L} \quad (4.8)$$

$$NMAE = \frac{MAE}{\sum_{u_k,i} r_{u_k,i} / L} \quad (4.9)$$

Where, $\hat{r}_{u_k,i}$ is the predicted QoS value of service i invoked by service user u_k while the $r_{u_k,i}$ is the real QoS value of service item i invoked by service user u_k. In addition, L is the number of predicted values. And the low values of these metrics represent the high prediction accuracy of corresponding approaches.

We compare our approach with seven other well-known approaches which includes,

- UPCC [12], which is a user-based collaborative filtering method using Pearson Correlation Coefficient to measure similarity between users.
- IPCC [12], which is an item-based collaborative filtering method using Pearson Correlation Coefficient to measure similarity between items.
- WSRec [12] , which is a QoS-aware hybrid Web service recommendation approach by combining both UPCC and IPCC with confidence weight

- BiasSVD [120], which is latent factor model using Singular Value Decomposition (SVD) with the addition of biases to user and item
- GM [121], which is a greedy method for ordering items
- CloudRank2 [21], which is a cloud service ranking method using confidence levels of different preference value

Methods	**Density**			
	0.1	**0.2**	**0.3**	**0.4**
UPCC	0.86436	0.84087	0.83125	0.79215
IPCC	0.79241	0.77498	0.75175	0.72198
WSRec	0.78985	0.75871	0.72478	0.71875
GM	0.74394	0.70214	0.68241	0.65097
CloudRank2	0.75756	0.73589	0.70423	0.68847
2RHyRec	0.69026	0.67543	0.64891	0.62345
Hybrid	0.62376	0.61375	0.60345	0.58794

Table 4.2 MAE values for performance comparison of different methods

- 2RHyRec [122], which is a ranking-oriented hybrid approach which combines collaborative filtering and latent factors.

Methods	**Density**

	0.1	**0.2**	**0.3**	**0.4**
UPCC	0.92695	0.89453	0.87548	0.86435
IPCC	0.84098	0.79248	0.76476	0.72439
WSRec	0.81785	0.77621	0.74835	0.70986
GM	0.77284	0.73452	0.70864	0.68432
CloudRank2	0.78584	0.76432	0.72086	0.69893
2RHyRec	0.75542	0.69326	0.66304	0.66109
Hybrid	0.68235	0.64286	0.61045	0.59345

Table 4.3 NMAE values for performance comparison of different methods

Table 4.2, and Table 4.3 shows the MAE and NMAE values of seven recommendation approaches based on response time, with different densities 01, 02, 03, and 0.4. In addition, we set *c=0.85*, *k=30*, α *=50/T*, β *=0.1* in the experiment. We repeated the experiment 10 times and the average value was taken. Based on the results in Table 4.2 and Table 4.3, our Hybrid approach significantly outperforms other six approaches under both MAE and NMAE metrics consistently, indicating that the prediction accuracy can be improved by employing correlation-based and semantic content recommendation. Recommendation approaches with ranking-oriented methods like(GM, CloudRank2, 2RHyRec, and Hybrid) obtain smaller MAE and NMAE values while recommendation approaches based on rating-oriented methods obtain bigger MAE and NMAE values. For all the methods, gradually increasing matrix

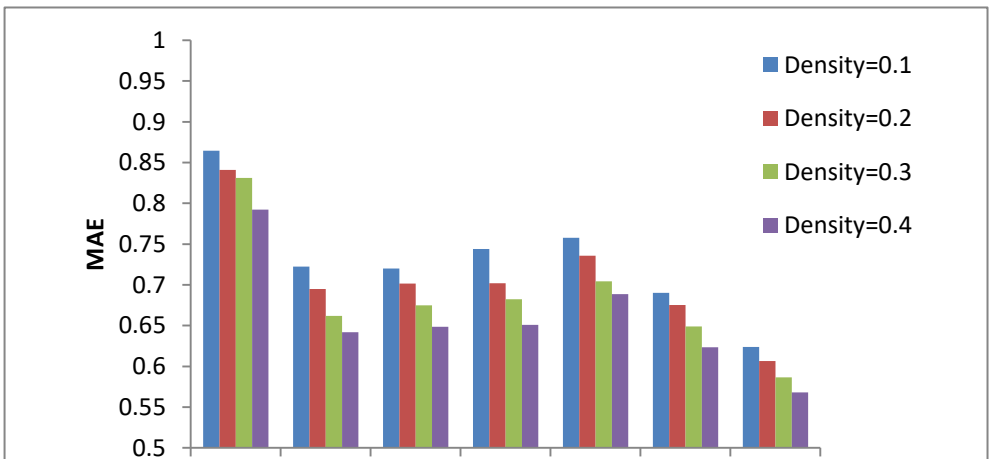
density values from 0.1 to 0.4 with the step value of 0.1 improve the prediction accuracy. This observation shows that increase in the quantity of training matrix has an important influence on prediction accuracy.

4.3.2 Matrix Density Effect

To study the impact of training matrix density, we set *c=0.85*, *k=30*, α *=50/T*, β *=0.1*, *density=0.1*, *0.2*, *0.3*, and *0.4* respectively. Figure 3a, and 3b show the experimental results with MAE and NMAE, in which seven approaches have better prediction accuracy with both MAE and NMAE when *density* increases from 0.1 to 0.4. This observation shows that bigger training matrix density can provide more QoS information to enhance the prediction accuracy. Especially, with same experimental settings and just by changing the *density* range from 0.1 to 0.4, the same as the results illustrated in Table 4.2 and Table 4.3, our approach obtains smallest MAE and NMAE values than other six approaches, which illustrates more accurate prediction performance.

4.4.3 *Top K* Neighbours Effect

In recommendation approaches based on *top K* neighbours, the number of neighbours is an important factor influencing the prediction performance. To determine the



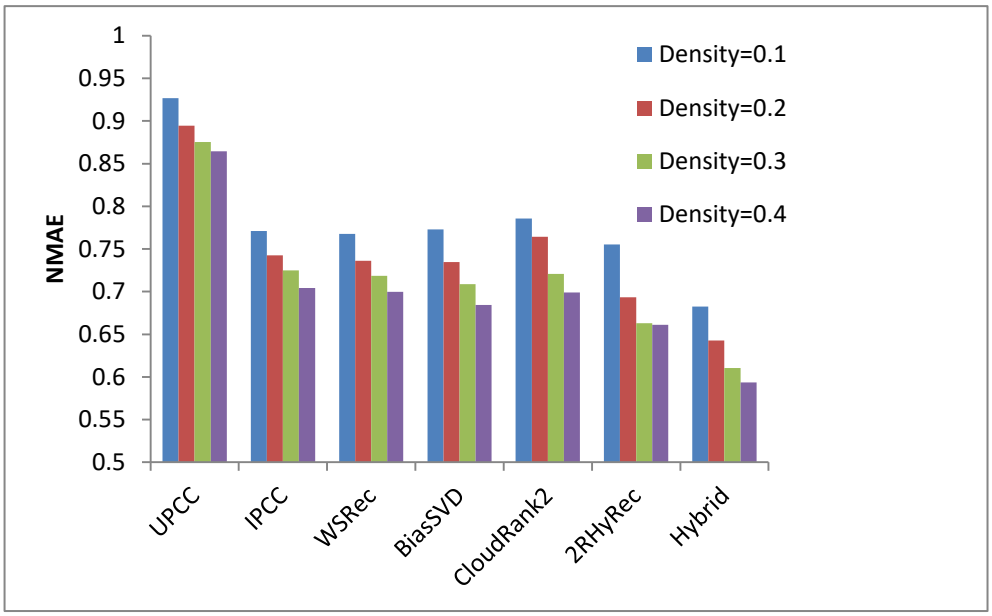


Figure 4.5 Effects of matrix density for NMAE

4.4.3 *Top K* Neighbours Effect

In recommendation approaches based on *top K* neighbours, the number of neighbours is an important factor influencing the prediction performance. To determine the impact of *top K* on the accuracy of the hybrid model we set $c=0.85$, $\alpha = 50/T$, $\beta = 0.1$, where value of *top K* changes from *5* to *60* with steps of *5* and *10*. As illustrated in Figure 4.4, the effect of *top K* value on the hybrid method was shown with respect to response time; the X-axis shows the *top k* value while the Y-axis shows the MAE

and NMAE values. Figure 4.4, and 4.5 show that the four curves of *density=0.1, 0.2, 0.3,* and *0.4* respectively experience similar trend with the increase in the value of *top K*. The figure for both NMAE and MAE shows a downward trend with the increase of the *k* neighbours from 5 to 60, which shows that the prediction accuracy can be increased by providing more neighbours. In addition, after number of *top k* reaches 40 the trend shows a constant value with the increase of *top k* neighbour, this implies more growth of *top K* after certain value adds more neighbours that are not very similar and it does not contribute enough in the accuracy of QoS prediction.

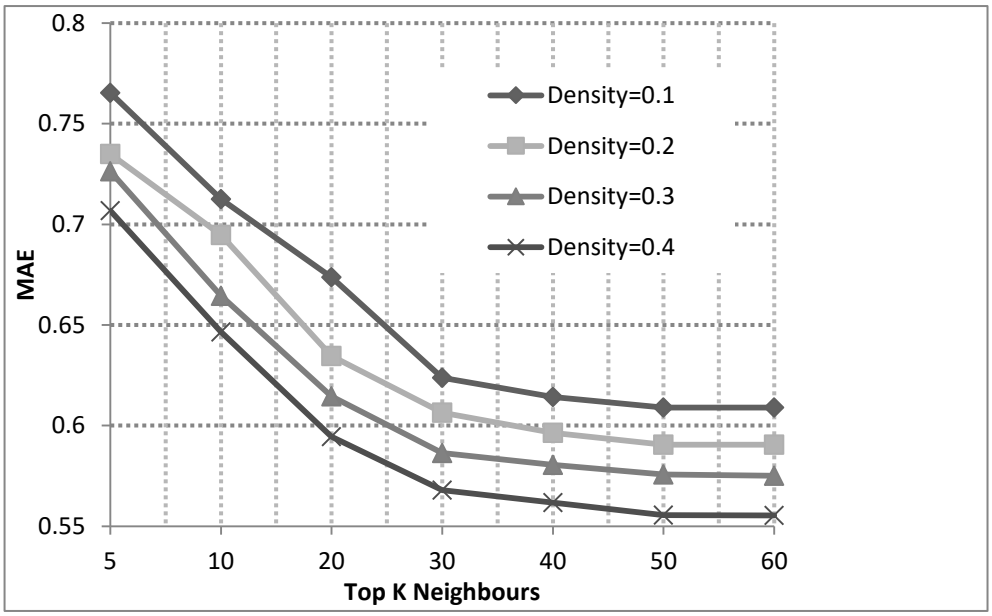
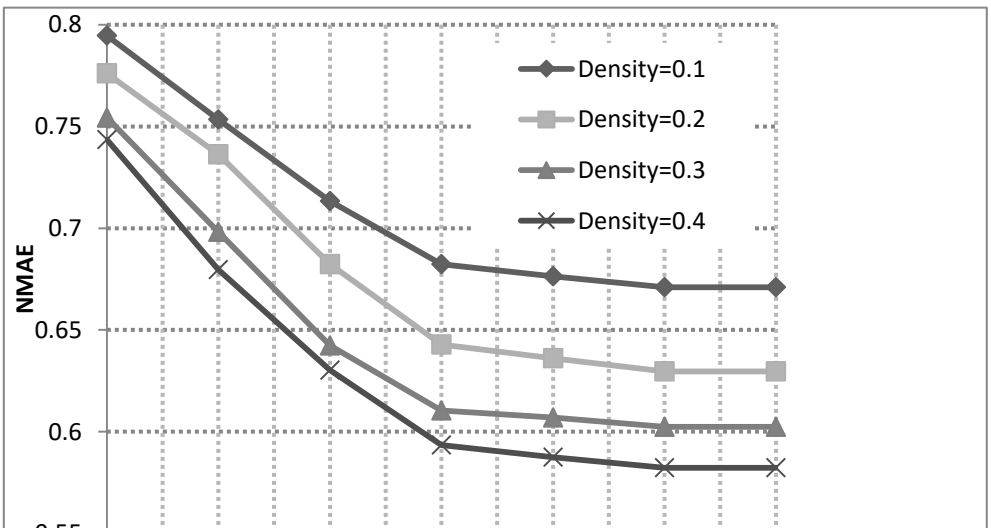


Figure 4.6 Effects of top K neighbours for MAE based on different matrix



4.3.4 Recommendation Evaluation

In order to study the recommendation performance, we compared our proposed hybrid recommendation approach with the other three methods: GM, CloudRank2, and 2RHyRec.

The recommendation performance was evaluated by examining the quality of top *x* rankings of Web services (*x*= 5, 10, 15, and 20). Specifically, after each model is learned. The average precisions and average recalls for top *x* recommendations were used as the evaluation metrics. Average precision was calculated as the ratio of the number of top *x* recommendation hits to the recommendation size; and average recall was calculated as the ratio of the number of top *x* recommendation hits to the size of the user's validation item set. We calculated the average precisions and recalls of all users for three different methods.

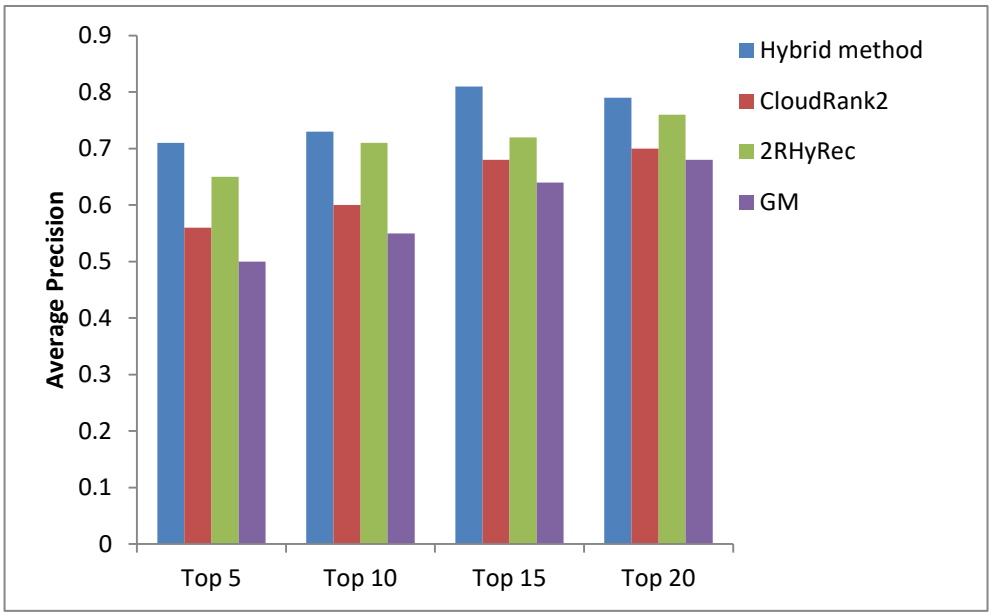


Figure 4.8 Average Precision for Recommendation performance comparing Hybrid, Collaborative Filtering, Top K correlation, and Content-based

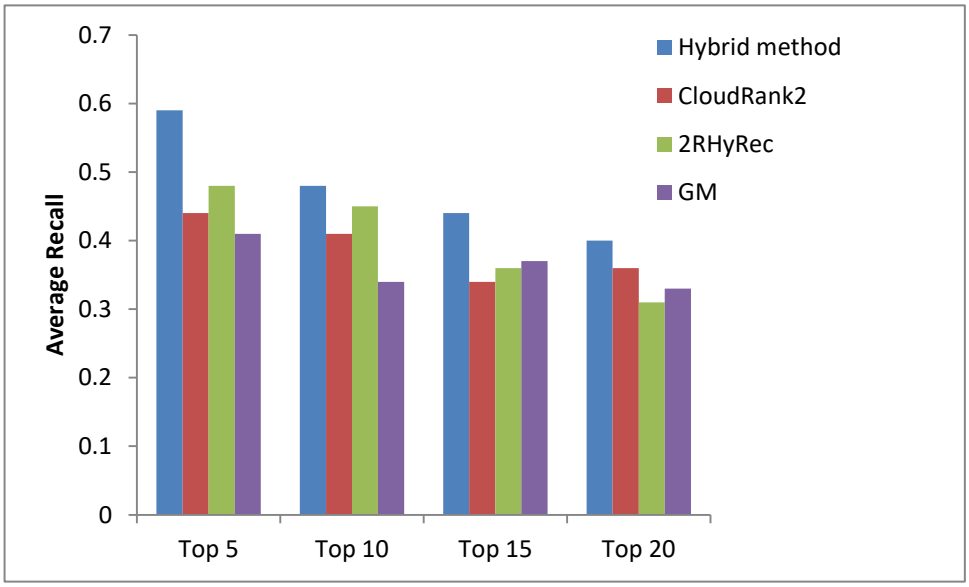


Figure 4.9 Average Recall for Recommendation performance comparing Hybrid, Collaborative Filtering, Top K correlation, and Content-based

Figures 4.5 and 4.6 show the result. From Figure 4.5 we can see that, the top *x* precision values of our hybrid approach are higher than GM, CloudRank2, and 2RHyRec. In Figure 6 the top *x* recall values of Hybrid approach are also higher than

the other three approaches. It is evident that our approach outperforms the other two approaches and more relevant Web services can be recommended by our approach.

4.4 Summary

Web services recommendation and selection is a fundamental issue in service oriented computing. In this paper, we have proposed a hybrid approach for effective Web services recommendation. Our approach exploits a model that systematically combines correlation-based and semantic content-based recommendation. In the first step the correlation-based recommendation of Web services has been discussed. With the correlation the relationship between web services can be learnt by using the number of co-invoked web services by different service consumers. Based on the correlation the Top-k neighbours are used in the QoS prediction. In the second step, content-based recommendation has been used, for which latent information about services is extracted using LDA topic modelling. From the latent interests two matrices were constructed for service-interest and consumer-interest relationships to predict missing QoS attributes. Finally, using a hybrid method which combines correlation-based and content-based, the Top-K recommended services are returned to the consumer.

Our approach is validated by conducting several experimental studies that used publicly available real-world web services datasets. The experimental results show that our approach outperforms state-of-the-art methods including the collaborative filtering, content based, and other hybrid methods in terms of QoS prediction and ranking performance.

In this chapter a personalized service recommendation approach has been proposed to let customers select suitable services for their requirements. In the next chapter we propose a service discovery and selection approach in a self-organized decentralized environment.

Chapter 5 Decentralized service discovery and selection

5.1 INTRODUCTION

In the recent years distributed technologies are introduced through Service Oriented Architecture (SOA), Grid computing and Cloud computing, all of these technologies have been used throughout organizations and companies of different sizes. With the advances of web service technology the web is moving from data- oriented web to service oriented web. Continually the number of web services increases, for example current cloud computing trend which is a Service-Oriented application has been growing in number of web services such as Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS). The increasing number of web services has led to critical issues like efficiency and scalability, however most of the current approaches focus on centralized architectures, in which a single service registry or multiple service registries synchronize together. With centralized approaches, you can have standardized hardware and software, resources can be efficiently administered, and there is no need for redundant cost and resources. However, centralized approaches lead to bottlenecks, single points of failure, and scalability limitations. In addition, one of the main drawbacks of centralized approaches is having a global knowledge about the system which is not presented in decentralized environments. While availability and demand for services grows significantly, the inherited disadvantages in centralized environments prevent web services from being applied in large scale service networks. As Service Oriented

Computing environments are largely distributed, a decentralized approach appears to be a suitable way to address the issues and achieve scalable, reliable and robust service discovery and selection.

Peer-to-Peer (P2P) has been used worldwide for resource sharing and received great success such as in (Gnutella, Chord, Freenet, JXTA and etc.). P2P technology is one of the areas which provides a universal approach in improving reliability, scalability, and robustness of distributed systems by weakening centralized control. In a P2P environment each node can play the role of provider or consumer depending on the need at any moment of time. P2P is located in an overlay network and distributed without any centralized control. There are approaches which use a decentralized system without central supervision to discovery services as in [123], [104], and [124]. The challenge is to find the service with fewer steps, which considers only local and neighbour information. In this approach we are designing a novel self-organized P2P architecture and social enhanced model for collaborated, dynamic and context-aware service discovery and selection to improve the scalability and efficiency of traditional methods. In this approach homophily was used, meaning that the most noticeable properties of services which determine similarity between nodes in the system is utilized as a method of bootstrapping and creating links between nodes. The discovery process is based on the similarity, previous interaction and social behaviour of nodes, which will help the discovery process and create a more dynamic environment for joining and leaving of the nodes. It is not only about discovering services, it also considers the selection of the best available service by taking into account the QoS properties of the services.

5.1.1 Motivation

Web services have become a de facto in the field of computing, and seemingly enter every aspects of life. SOA, Grid, and cloud computing are different paradigms of

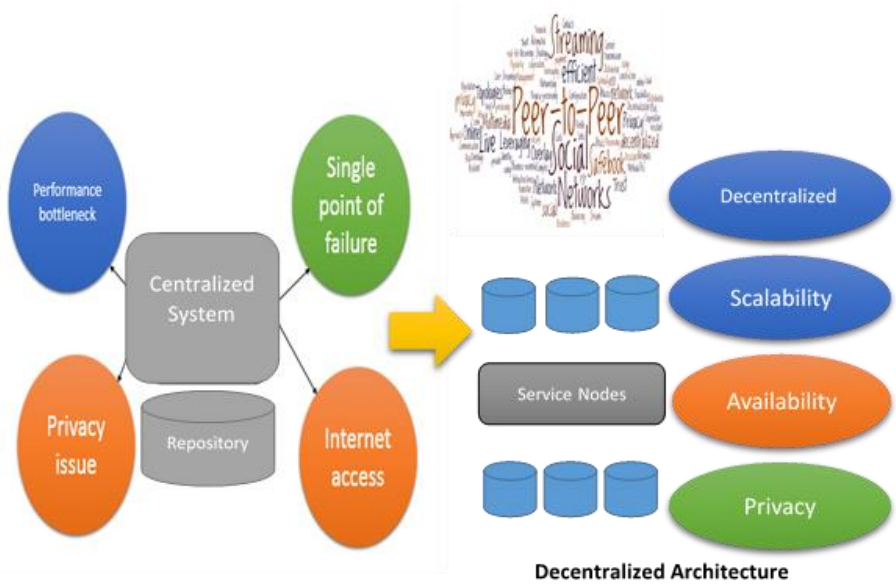


Figure 5.1 Centralized and Decentralized Service Model

computing based on web services. With the increase in the number of web services, the requirements of consumers increase seamlessly. A centralised approach cannot cover all aspects of practically unlimited consumer requirements, because it is not feasible for any service repository or service provider to meet all consumer demands. On the other hand, a centralized approach is always prone to single point failures and bottlenecks, which hugely affect the availability and reliability of web services. Therefore, a decentralized P2P approach is proposed where each peer node can represent a cloud platform and provide different services as shown in Figure 5.1. A practically unlimited number of services can be found outside each cloud platform and can be outsourced based on the consumer request. The nodes can dynamically join and leave the network which provides an elastic environment for limitless number of

services; as a result each node can outsource services from other nodes based on the context and social behaviour. By this way consumer demands will be fulfilled, and the issue of single point failure will be solved using different number of nodes each of them act as a provider and requester for services.

5.1.2 Contributions

The paper presents a service discovery and selection algorithm in a decentralized environment which will improve the efficiency and performance of the discovery and selection process in comparison to the previous works in this area. Contributions can be summarized as

- Designing a service discovery algorithm in a decentralized P2P environment considering homophily for node distribution and using social behaviour and node interaction to discover the most relevant services.
- Proposing a service selection algorithm to consider partial matching of the services based on QoS parameters, and a related ranking algorithm capable of ranking all the discovered services and returning the best available service to the consumer.
- Use of real world semantic web service dataset to provide experimental proof of the performance and efficiency improvement of the proposed algorithm when compared to the existing algorithms.
- Implementation of a prototype platform for Peer-to-Peer service discovery and selection in a real world environment.

The rest of the chapter is structured as follows: Sections 5.2 illustrate system structure and provides detail of the homophily similarity algorithm, and provides details on community creation in decentralized environment and also discuss service discovery and selection. Section 5.3 and 5.4 consider network simulation and evaluation. Final section 5.5 presents conclusion on the approach.

5.2 DECENTRALIZED SYSTEM STRUCTURE

The system consists of a set of nodes which are distributed in a decentralised manner; each node in the system does not have a global knowledge about other nodes except direct neighbours. This section explains the structure of the decentralized system, nodes, and the services as contents of the nodes.

Definition 1. The decentralized system can be formalized as $C = \{N,L\}$, such that $N = \{n_1,n_2,n_3,\cdots n_m\}$, where

N represents the list of nodes in the network, n_i represents the i^{th} node in the system and m is the total number of nodes. $L = \{n_i,n_j\}$, $(1 \leq i \leq m \text{ and } 1 \leq j \leq m \text{ where } i \neq j)$. L is a set of links between different nodes in the system, both i and j representing two different nodes.

Definition 2. Each node n_i in the system N can be formalized as a tuple of $\{S_i, Ne_i, K_i, Q_i, \mu_i\}$ where:

- S_i is the list of services in the i^{th} node
- Ne_i is the list of direct neighbours for i^{th} node, and ε is similarity threshold to determine if neighbour node content is similar to the i^{th} node content.

- K_i is the knowledge index for the i^{th} node, which contains the service categories of neighbouring nodes.
- Q_i is a received query at any given time in the i^{th} node. • μ_i is the selection function for i^{th} node to determine nodes which a query can be forwarded.

The network is decentralized, each node at least connected to one neighbour node based on the content similarity, and the environment is dynamic in which nodes can join and leave the network. The distribution of queries is not based on flooding or random distribution; it is based on the similarity of the query to the neighbour node which is determined by the function μ_i.

Definition 3. Service Distribution and node organization Let $S_i = \{s_1, s_2, ...s_n\}$ be list of services for i^{th} node, for each service there is a service tuple: Service tuple, $s_n = \{I_n, O_n, Cat_n, QoS_n\}$ Where, I_n is the list of service inputs; O_n is the list of service outputs; Cat_n represents the user defined service category; QoS_n is the list of quality of service properties for the service n. In the context of services, in every node in the network there is a list of services with different functionality and QoS properties. During the discovery process, based on the service request, each node checks its local services for a service which matches the service request. If there is any service matching the request it will return that service to the requester and forward the request to another node which has similarity with the service request. This process continues until the Time To Live (*TTL*), which is the maximum number of times a query can be forwarded, becomes zero.

5.2.1 Homophily in decentralized environment

Homophily was first used by Lazarsfeld and Merton [125] to define interactions and links between two individuals based on their similarities in set of social entities such as ethnicity, religion, or gender to establish links between them. Using same approach homophily became one of the most prominent properties present in complex networks [126]. In this paper homophily is used to determine the similarity between nodes in the decentralized system, which is based on the similarity of service values for Inputs, Outputs and Categories.

To determine the homophily similarity between nodes Cosine Similarity is used. Cosine Similarity is a measure of similarity between two vectors of an inner product space that measures the cosine of the angle between them. To find the relationship between two services, each service is constructed as a vector in the space of node. The cosine between these vectors gives a measure of similarity. Such functions have largely been used in the web space to identify similarity of documents and web pages. It has been one of the most preferred techniques in information retrieval, clustering and has also been applied to pattern recognition and medical diagnosis [20].

The following explains the details of Cosine homophily process. Given two vectors of attributes, S_k and $S{\cdot}_l$, the cosine similarity, cos (θ), can be determined:

$$Hcos(S_k, S{\cdot}_l) = cos(\theta) = \frac{S_k \bullet S{\cdot}_l}{\|S_k\| \|S{\cdot}_l\|}$$

$$Hcos(S_k, S{\cdot}_l) = \frac{\sum_{i=1}^{n} S_k * S{\cdot}_l}{\sqrt{\sum_{i=1}^{n} (S_k)^2} * \sqrt{\sum_{i=1}^{n} (S{\cdot}_l)^2}} \quad (5.1)$$

Where, vector S_k represents all attributes of S_k $\{s_k^{ins}, s_k^{out}, s_k^{cat}\}$, while vector $S\cdot_l$ represents all attributes of $S\cdot_l$ $\{s\cdot_l^{in}, s\cdot_l^{out}, s\cdot_l^{cat}\}$ as in Figure 5.2. The result $Hcos(S_k, S\cdot_l)$determines the similarity of two service vectors S_k and $S\cdot_l$ based on the similarity of their attributes.

$$\begin{matrix} S_k \\ S\cdot_l \end{matrix} \left\{ \begin{matrix} s_k in & s_k out & s_k cat \\ s\cdot_l in & s\cdot_l out & s\cdot_l cat \end{matrix} \right\}$$

Figure 5.2 List of inputs, outputs, and categories for two vectors in Cosine similarity

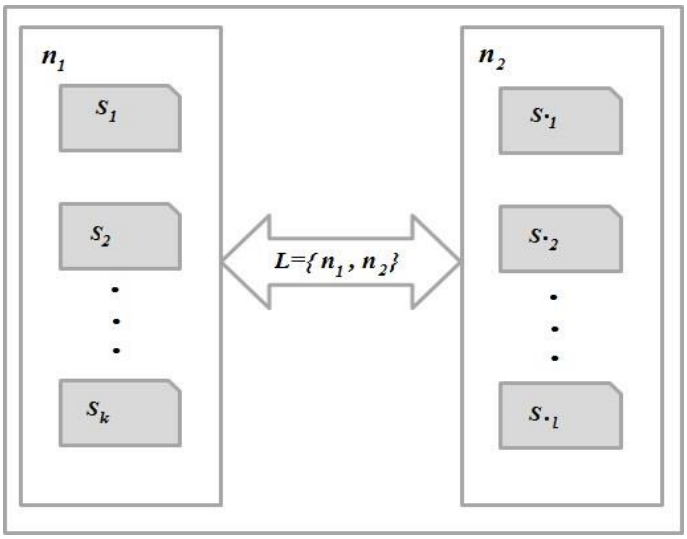


Figure 5.3 Relationship between two nodes n1 and n2

By using equation (5.1), similarity between services of both node n_1 andn_2 will be calculated as shown in Figure 5.3,

$$Hcos(n_1, n_2) = \frac{\sum_{k=1}^{q} \sum_{l=1}^{p} Hcos(S_k, S\cdot_l)}{p*q} \qquad (5.2)$$

Where $Hcos(n_1,n_2)$ represents the similarity between two nodes n_1 and n_2 and the value is normalized to be between [0,1]. The value of q represents number of services in n_1, while the value of p represents number of services in n_2. $Hcos(S_k,S\cdot_l)$ represents the similarity between service vector S_k in n_1 and service vector $S\cdot_l$ in n_2; k represents k^{th} service in n_1 and l represents the l^{th} service in n_2 .

To establish the connection between two nodes there is a similarity threshold, ε, the threshold can be set to determine if the similarity of two nodes is in the level to establish connection.

5.2.2 Community creation

Once Cosine homophily has been defined in the context of decentralized systems, and then it is described how it is included in both structure creation and service discovery process. Cosine homophily establishes a measure of similarity between two nodes based on services similarity inside the nodes. The similarity measurement is taken into account by nodes during community creation process.

- Each node that is part of the system is considered an entry point. At the beginning each node n_i in the system randomly connects to a number of other nodes in the system.
- The number of random neighbour is determined by the system. Each node n_i should know at least one node n_j that already present in the system.
- The establishment of connection between both n_i and n_j is based on the Cosine homophily between them.

The Probability T_l of establishing a connection between node n_i and node n_j is

$$T_l(n_i, n_j) = \left(\frac{1-Hcos(n_i,n_j)}{T}\right)^{-r} \quad (5.3)$$

To obtain the probability distribution, the cosine homophily between two nodes should be divided by an appropriate constant *T* that indicates the degree of precision to consider two nodes equal. The *r* parameter is a homophily regulator. When *r* is zero, the system shows no homophily (i.e, nodes are not grouped by similar services). As *r* grows, links tend to connect nodes with more similar services. Basically, *r* makes the system create communities with similar services.

Nodes have a greater chance of establishing connections with other nodes if they provide similar services in the system. As a result of this behaviour, communities of similar nodes are created in a decentralized way. The resulting system structure is a network based on homophily as shown in Figure 5.4, which grows according to a simple self-organized process. The construction process of a growing network ensures that the oldest nodes have a higher probability of receiving new links than the newest ones.

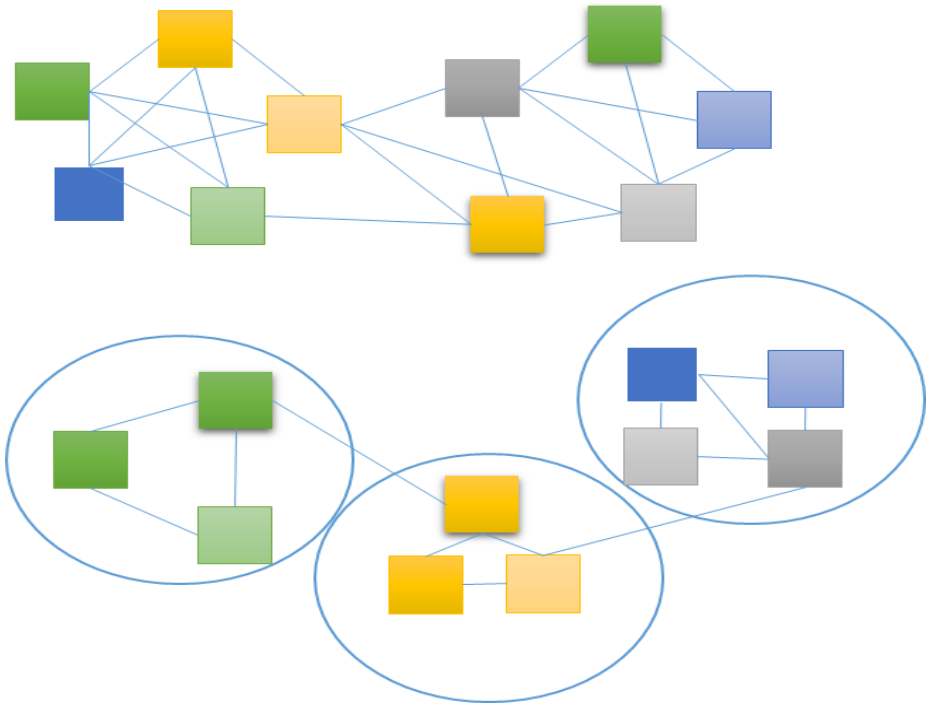


Figure 5.4 Homophily based community creation

5.2.3 Service discovery

In decentralized structure service discovery relies on the local information in the nodes, which is information about the content of the node and knowledge about neighbouring nodes. In this way it prevents central supervision, eliminates single points of failure in the system, and precludes the need to collect global knowledge about the structure of network which is not available in a large dynamic environment.

In the following section the process of service discovery is explained in a decentralized environment using local information of the nodes and knowledge about first neighbours of the nodes.

During service discovery, when a node n_t sends a request to node n_i, node n_i searches its local services to find any service similar to the request r_t by node n_t. If there is any

service similar to the request, it will be returned to n_t. The request will then be forwarded to most promising neighbours of n_i. The most promising neighbours are neighbours with highest score of similarity to the target node n_t. This process of forwarding query continues until *TTL* becomes zero.

The equation which calculates the most promising neighbour from node n_i is as follows,

$$\mu_i(n_t) = argmax\ Ps(\text{h}n_j,n_t\text{i}) \tag{5.4}$$

For each neighbour of n_j, *Ps* determines the probability that node n_j is a promising neighbour to forward the query

$$Ps(\langle n_j, n_t \rangle) = 1 - \left[1 - \left[\frac{Hcos(n_j,n_t)}{\sum_{n_j \in N_i} Hcos(n_j,n_t)}\right]\right]^{|N_j|} \tag{5.5}$$

For this probability Cosine homophily-based factors and degree based factors (number of neighbours $|N_j|$) is used to explore the network. For cosine homophily in the discovery process, we use information available in knowledge index of each node about neighbours. In the knowledge index there are categories of services provided by neighbours. These categories are user defined categories, and it is possible to have multiple categories for a single service in a node.

5.2.4 Service selection

There are two criteria which service selection is depending on; the first one is QoS from the service tuple, which determines the quality properties of each service. The

second one is node Knowledge, which is based on the previous interaction of the nodes and consumer rating for the services in the node.

The selection process considers partial selection algorithm from our previous work [4], which returns the best available web service to the consumer when the providers offer does not fully match every consumer QoS requirements. By having QoS properties from services and node knowledge on one side, and consumer specified QoS requirements on the other side, the normalized value of each QoS property can be calculated by using the case dependant equation (5.6) (5.11).

For values with high tendency

$$\frac{R_{ij}}{R_l} \qquad R_{ij} < R_l \qquad (5.6)$$

$$\frac{R_{ij}-R_l}{R_h-R_l}+\alpha \qquad R_l \leq R_{ij} \leq R_h \qquad (5.7)$$

$$\frac{R_{ij}}{R_{max}}+\beta \qquad R_{ij} > R_h \qquad (5.8)$$

For values with low tendency

$$\frac{R_h}{R_{ij}} \qquad R_{ij} > R_h \qquad (5.9)$$

$$\frac{R_h-R_{ij}}{R_h-R_l}+\alpha \qquad R_l \leq R_{ij} \leq R_h \qquad (5.10)$$

$$\frac{R_{min}}{R_{ij}}+\beta \qquad R_{ij} < R_l \qquad (5.11)$$

Where,R_{ij}, is the value of i^{th} property of j^{th} service. R_l, is the lower limit of consumer requirement for an attribute, while R_h, is the higher limit of consumer requirement for the attribute. R_{max}, is the maximum value of a QoS property offered by the web

services under consideration, and R_{min}, is the minimum value of a QoS property offered by the web services under consideration. In the equations the value of α and $\beta \in \{1,2,3,...\}$ *where* $\alpha < \beta$.

The calculation of the accuracy matrix is dependent on the tendency of QoS parameters. Tendency explains how the numeric value of a service changes for the service to be perceived as better. The tendency for the majority of values is expected to be high, others are expected to be low such as Response time and Latency. The tendency of a parameter indicates whether high or low values are regarded as more desirable in an ideal scenario. For example, high Availability and Throughput values imply a better service whereas low Response time and Latency values would also generally indicate better service.

The equations 5.6-5.11 generate results which are normalized in the range [0, 1]. The constants α and β are introduced in order to discriminate between the three ranges. For results in the loose range, the value remains in the range of [0, 1]. For results in the preferable range, α is added to the equation and the results are in the range [α, α + 1]. For results in the tight range, β is added and the results are in the range [β, β + 1].

For consistency, using equation (1-6) values of QoS properties were normalized to be in a small range, in which all the values in the accuracy matrix lies in the range of [0, β + 1]. In the selection process there are many services, the main purpose of the algorithm is to arrange all the QoS values for services based on minimum and maximum values , and arrange the values between [0,β + 1].

On the other hand, in the algorithm, every value is considered precisely based on the range. Consider a value for availability; assume the preferable range is [80% − 90%] .Any service having availability value less than 80% will be counted as loose, between [80% − 90%] will be counted as preferable range and α will be added, finally any value larger than 90% will be counted as tight and β will be added. Since $0 < \alpha < \beta$ it guarantees that the higher range always has higher value than the other two ranges. The results of the calculations are used to populate the accuracy matrix. The matrix shows how accurately each advertised service matches the overall consumer requirement without yet considering the desired preference of each attribute. After the matrix is fully populated, the rank of each service is calculated by summation of the QoS attribute and weight product for that service using equation (5.12).

$$\mathrm{R}^{i} = \sum_{j=1}^{n} A_{ij} * W_{j} \qquad (5.12)$$

Where R_i represents rank of i^{th} service , A_{ij} represents the accuracy value of j^{th} QoS property of i^{th} service, and W_j represents weight of j^{th} QoS property.

5.3 SIMULATION ENVIRONMENT

5.3.1 Performance metrics

Performance was evaluated with the following measures: Recall: the ratio of the number of found files to the number of all matched files in the network.

Average path length (APL) of searches: the average distance from the query originator to the targeted node which first finds a matched file. If none are found, the average path length of the search is set as four [*TTL* + 1] in the simulations. This metric is used to measure the speed of resource discovery. Other metrics are Recall per visited node,

Average number of found files, Number of visited nodes, Number of query messages. Figure 5.5 shows the detail of the network initialization and simulation.

5.3.2 Topology initialization and evolution

In order to better observe the evolution of network topology in the simulations, the process was started from a small-size random network with 100 nodes. Each peer node connected to four peer nodes bi-directionally based on similarity to generate a homophily topology. Since at the initial phase there is no interaction between peer nodes, each peer node has an empty knowledge index which can contain a maximum of 1000 entries between topics and associated peer nodes (if no other size is specified). In the simulations, a dynamic network was started with a small set of peer nodes. The peer nodes frequently transition between being present or absent from the network. In order to explore the effect of topology, random topology, ring topology and power law topology are also implemented to draw comparisons regarding the efficiency of service discovery.

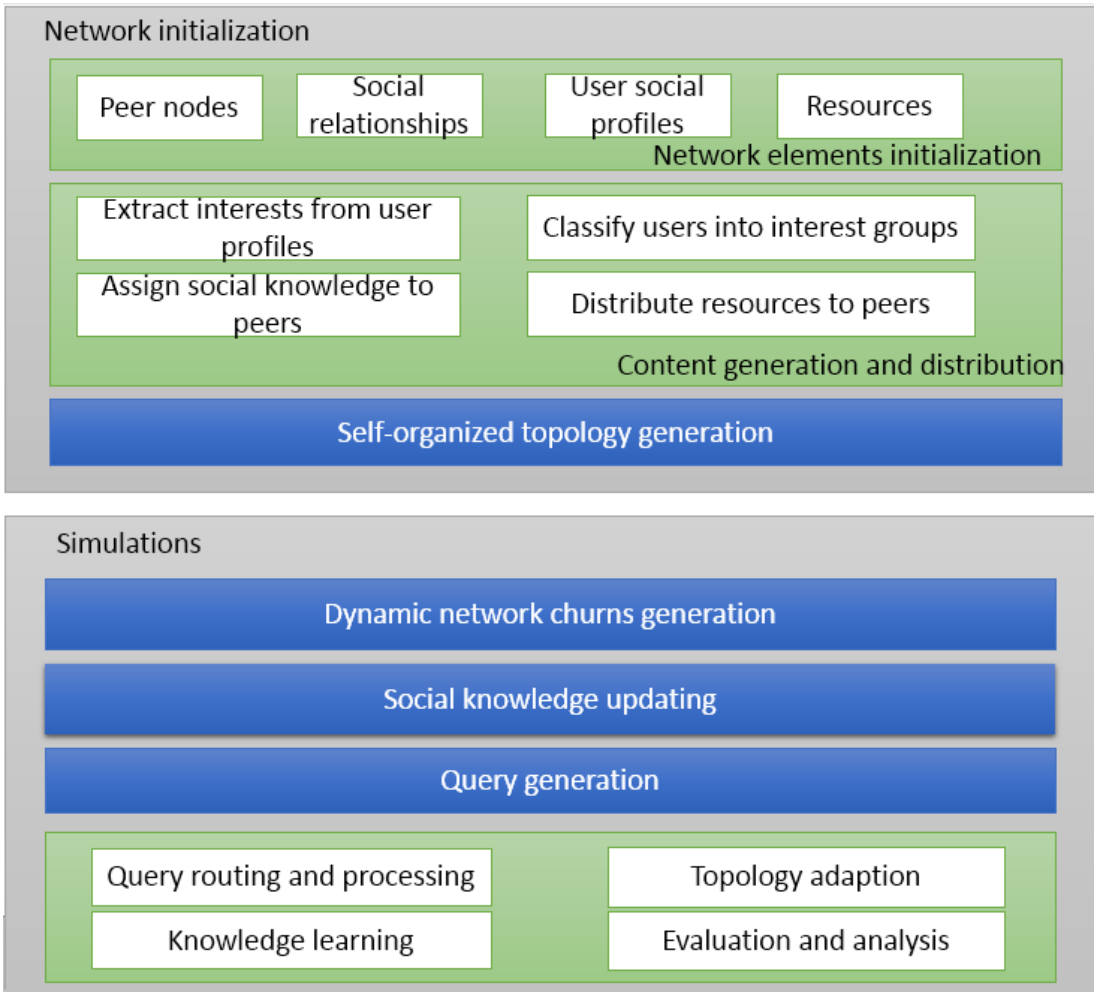


Figure 5.5 Network initialization and simulation diagram

5.3.3 Content generation and distribution

In the experiments, each network was initialized with 100 nodes. Each node offers 11 semantic web services which are uniformly distributed to the node. All the experiments were performed with real semantic services. The set of semantic services used for the experiments were from the test collection OWL-S TC4 [1]. Based on the OWL-S TC4 dataset, which has 1082 semantic web services, the semantic content of

[1] http://www.semwebcentral.org/projects/owls-tc

the services including ID, Name, Input, Output and discretion are extracted for the process of service discovery. Some key QoS parameters such as response time, availability, reliability, latency are created randomly and assigned to the services for the process of service selection.

5.3.4 Search network

In each time step of the simulations, an online node was randomly chosen as the requesting node and a search with a query was started. Queries were uniformly generated among all the agents. This means that all the agents in the system had the same probability of generating service queries. Each query was tagged with a *TTL* to limit the life time of a message to three hops in the simulations. The generated queries were propagated with the proposed routing algorithm. To decide the average degree of connection it is influence on the average path length was evaluated. As the average degree of connection increases, the paths get shorter and it is easier to locate the target agent since agents have available more possibilities to guide the search.

5.4 PERFORMANCE EVALUATION

5.4.1 Service discovery performance

In this section results for precision and recall were compared for the homophily topology against random, ring and power law topologies. In each case, a node was randomly chosen to be the requesting node from the 100 node vector and a search was started with a query selected randomly from the list of queries in the current dataset. Each query is tagged by a *TTL* to limit the lifetime of a message to three hops in our simulation. The selected queries are propagated with the Gnutella routing algorithm.

For the evaluation 24 queries were selected randomly from the list of queries in the dataset, for each query the simulations of Service discovery were run. The tests were repeated for different *TTL* values starting from 2 to 11.For performance measurement we used precision and recall. Precision and recall are standard measurements that have been used in information retrieval for measuring the accuracy of a discovery or recommendation method or a discovery engine. Precision is defined as the ratio of the number of returned correct services to the total number of all returned services [110] . By contrast, recall is defined as the ratio of the number of returned correct services to the number of all correct services.

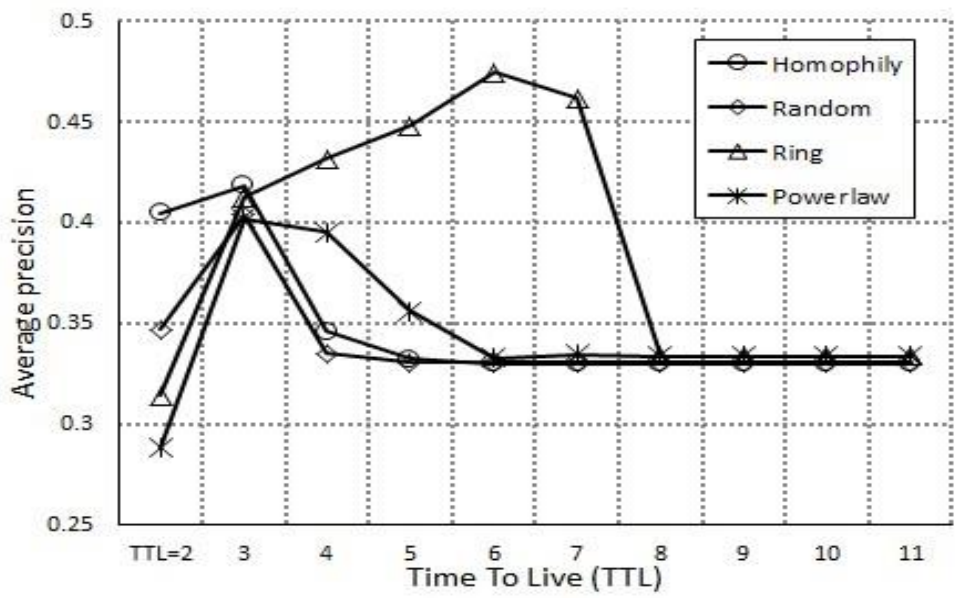


Figure 5.6 Average Precision calculation for Homophily, Ring, Power Law and Random Topology

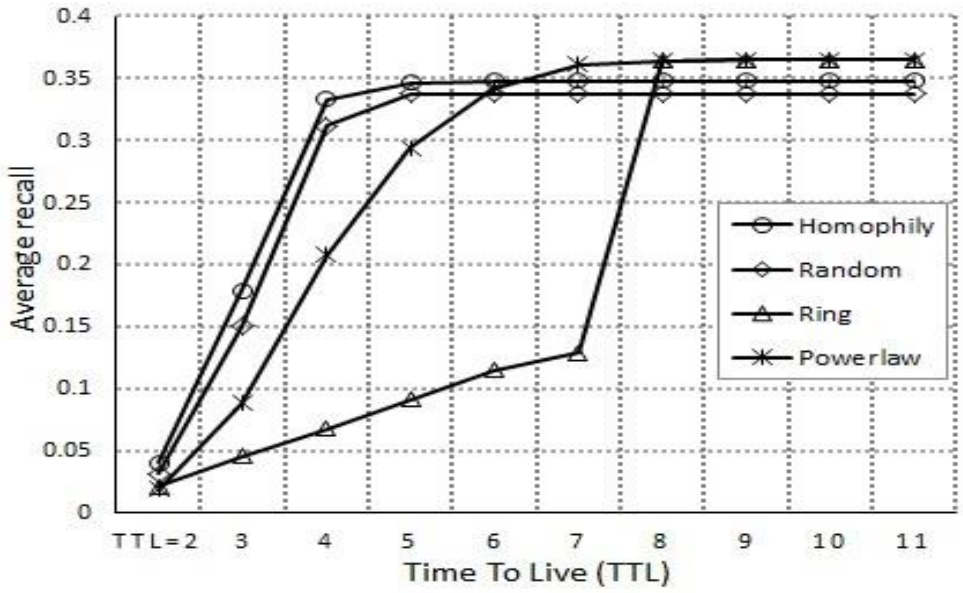


Figure 5.9 Average Recall calculation for Homophily, Ring, Power Law and Random Topology

From the results shown in Figure 5.6 and Figure 5.7 it is evident that homophily topology achieved better performance for lower TTL values. Increasing TTL values means increasing in the message exchange between nodes which in result increases overall system overload. For average precision homophily topology outperformed other topology for TTL values of 2, and 3. On the other hand for average recall homophily topology again outperformed other topologies for TTL values 2,3,4,5, and 6. If we consider values of both recall and precision, for example homophily topology has best precision (0.418) with TTL value 3, within the same TTL it has the best recall

(0.179) comparing to other topologies as well. But for ring topology it has best value (0.474) for TTL value 6, while it is recall value (0.114) for the same *TTL* is the worst among all the topologies. It is the same for Powerlaw topology; it is best precision

value (0.395) is for *TTL* 4 while it is recall for the same *TTL* is the second worst after ring topology with the value of (0.2). From the results we can see that homophily gets better precision and recall for lower *TTL* values and for the higher TTL the average of recall and precision is better than the other topologies.

5.4.2 Service selection performance

In the proposed algorithm the rank of web services was calculated based on QoS properties. To measure the performance the algorithm was compared to proposed algorithm in [123] which used probabilistic flooding-based method, by combining Simple Additive Weighting (SAW) technique and Skyline filtering. And also compared to to SPSE algorithm, which uses Pareto optimal-based solution method [67].

For the selection process the same set of tests were used as in the discovery process. Since the semantic dataset does not have any QoS properties attached to it, we randomly added four QoS properties to services in the dataset. The added QoS properties were Response Time, Availability, Reliability and Latency. For each test different requirements were used for QoS properties. Based on the requirements we checked number of returned services and relevant returned services.

From Figure 5.8 and Figure 5.9 it is apparent that the Accuracy Matrix approach improved both precision and recall compared to SAW and SPSE, the percentage of returned services in the Accuracy Matrix is higher than SAW and SPSE. In the same

way the percentage of returned relevant services in Accuracy Matrix is higher than both SAW and SPSE.

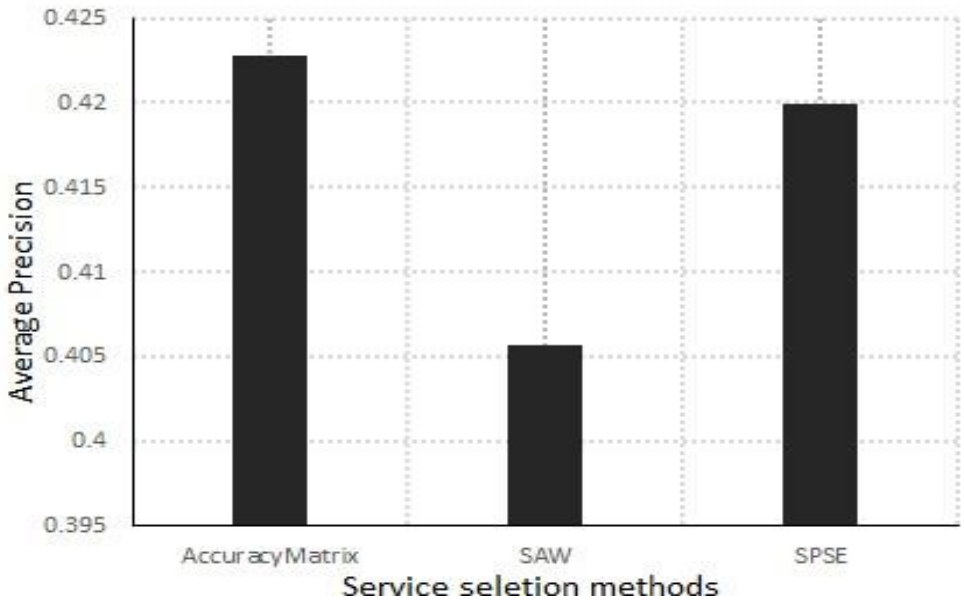


Figure 5.10 Average Precision for AccuracyMatrix, SPSE and SAW methods

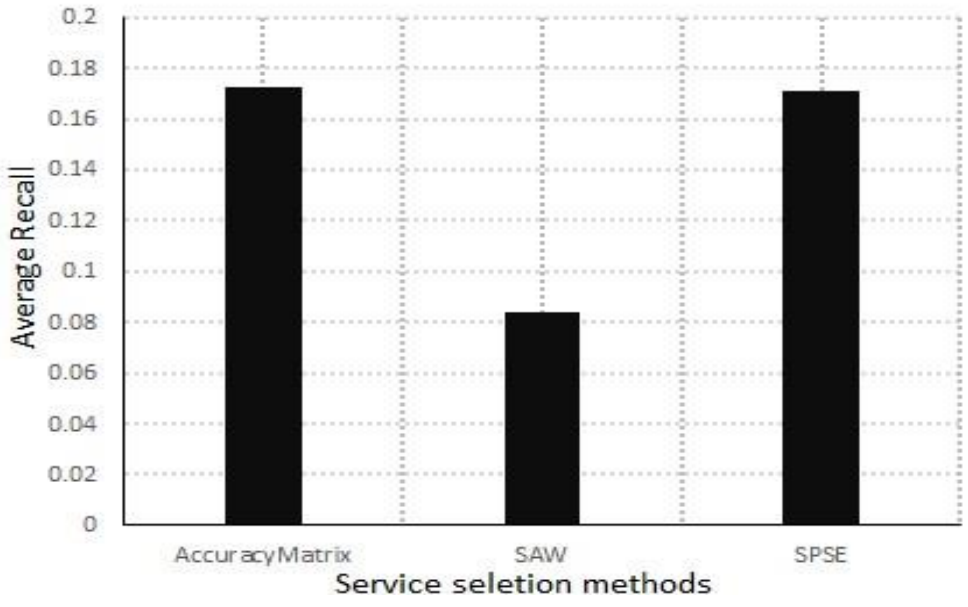


Figure 5.11 Average Recall for AccuracyMatrix, SPSE and SAW methods

The improvement is due to the inclusion of partially matched web services in the Accuracy Matrix approach. This confirms that our approach deals more accurately with the service query, and returns relevant service to the consumer even in the case where the candidate services only partially match the consumer requirements.

5.5 Summary

In this work, we proposed a context-aware service discovery and selection method in a decentralized environment. In the approach, each node in the network established connection to the other nodes based on homophily similarity between them. For the homophily a cosine similarity method was used which calculates service similarity based on inputs, outputs and categories of services. With cosine homophily nodes create a network based on similarity without having central supervision. Cosine similarity is used for bootstrapping and creation of network community, therefore it does not require any training period. With an increasing number of interactions between nodes the knowledge index gathers information about content of neighbouring nodes which in result helps the process of service discovery. In the experiment we compared Cosine homophily topology to Ring, Power Law, and Random topologies, the results showed improvements in both precision and recall in the service discovery. This is due to the use of homophily similarity which let the nodes to create community based on similarity of their contents.

One of the other features of this paper is the inclusion of partially matching services. When an initial list of web services is returned to the requesting node the algorithm

includes them in the selection process. The algorithm considers every service accurately and ranks them based on QoS properties and node knowledge. Experimental results show that Accuracy Matrix performed better comparing to SAW algorithm. Accuracy Matrix had better precision and recall

Chapter 6 Conclusion and Future Work

6.1 Conclusion

In the last few years services have become a popular research area that has attracted the attention of both the academic and industrial community. In the service oriented environment applications are developed in a loosely coupled way where consumers can find and invoke preferred services from a service pool. From an ever increasing number of services, finding a suitable service efficiently and effectively becomes a challenging issue. Especially, selecting a high quality service from a pool of functionally equivalent services where each has different quality.

In addition to functional properties, non-functional properties play an important role in the process of service selection and recommendation. Thus, a wide range of service selection and recommendation methods have been proposed. However, there is no real investigation on partially matched services (where consumers QoS properties partially matched providers service properties), what information affects quality of service properties or how to deal with the information provided for service selection and recommendation. The initial goals of our research were to introduce service selection and recommendation in centralized and decentralized environments. Based on the aim and objectives the overall thesis consists of three parts, in the first part we concentrated on the efficient algorithm for partially matched web services in the Internet of Services, for the second part the focus was on the personalized hybrid service recommendation algorithm using service correlation and semantic content features, while in the last part the focus was on a context-aware service discovery and selection algorithm in a decentralized platform. All three approaches in the thesis aim for the

improvement of service recommendation and selection approaches on different platforms.

In the first approach, service selection and recommendation in IoS was covered. Internet of Services is highly dynamic and continuously changing due to constant degrade, vanish and possibly reappear of the devices, this opens a new challenge in the process of resource discovery and selection. In response to increasing numbers of services in the discovery and selection process, there is a corresponding increase in the number of service consumers and consequent diversity of QoS available. Increase in both sides has lead to diversity in the demand and supply of services, which would result in the partial match of the requirements and offers. The approach proposed a service ranking and selection algorithm by considering multiple QoS requirements and allowing partially matched services to be counted as candidates for the selection process. Our algorithm includes partially matched services in the decision and recommendation process, this holds true even in the case where the attributes marginally match. It allows more services to be included and accurately ranked, providing the best choice to the consumer. Experimental results show that in extreme circumstances, our algorithm recommends services when other methods would return an empty list. In addition to that, the approach is generic and scalable which can deal with multiple QoS properties and large numbers of services. Experimental results from real world services showed that the proposed method achieved significant improvement in the accuracy and performance in the selection process. The scalability was tested by increasing number of services and QoS parameters. There was no significant performance degradation when reducing large datasets with multiple QoS

attributes. Our experiments rendered results that strongly support its use as a tool for service selection processing.

For the second approach, with the increasing number of web services having similar functionality but different QoS, it is challenging for service consumers to find out the best available web services that meet their requirements. In the approach, we proposed a novel personalised service recommendation method by using the LDA topic model, which extracts latent interests of consumers and latent topics of services in the form of probability distribution. In addition, the proposed method is able to improve the prediction accuracy of missing QoS properties of services by considering the correlation between neighbouring services using the number of co-invoked service consumers. As a result, the hybrid method which combines correlation-based and semantic content-based was used to return the Top-K recommended services to the consumer. Experimental results based on publicly available real world datasets showed that the proposed method achieved significant improvements in terms of accuracy and performance in comparison with state-of-the-art methods.

In the last approach, we are dealing with service discovery and selection in a decentralized environment. Traditional service discovery approaches are often supported by centralized registries that could suffer from single point failure, performance bottleneck, and scalability issues in large scale systems. To address these issues we proposed a context-aware service discovery and selection approach in a decentralized environment. In the approach homophily similarity was used for bootstrapping and distribution of nodes. The discovery process is based on the similarity of nodes, previous interaction and behaviour of the nodes, which helped the discovery process in a dynamic environment. Our approach is not only considering

service discovery, but also the selection of the best available web service by taking into account the QoS properties of the web services. Experimental results were based on a real world semantic web service dataset, the results showed that the approach achieved better performance and efficiency in both discovery and selection of web services.

6.2 Future Work

Although we have successfully achieved our research aims, but based on the promising research findings presented in this thesis, several future research directions has been recognized and planned. These future directions intend to make our approach more practically viable. The following works will be carried out in near future of our research,

- The partial service recommendation and selection algorithm discussed in chapter three is concentrating on single service recommendation and selection. Further investigation needed to be carried out about QoS properties in a composite service environment, where more than one service have to be invoked to carry out a specific job. Our partial matchmaking approach can be used as a promising tool for partially matched composite service selection and recommendation.
- The thesis used both functional and non-functional (QoS) properties of Web services which facilitate the automation of the discovery, matchmaking and recommendation processes. It should be possible to extend definition to deal with other properties of the Web service, such as a precondition and post condition which have to be satisfied to interact with a service. The

precondition may determine requirements that must be satisfied, such as, only flights from UK can be booked, or a consumer must be a registered with the service provider. The definition of the precondition of the Web service will take a form of a logical expression and must be satisfied in order to invoke the service.

- Further aspects of service selection process will be investigated, such as, concentrating on further refinery of our selection and recommendation algorithm results by the inclusion of consumer opinion, which will help in the prediction of the user QoS requirements and assigned weight value.
- For the personalized hybrid service recommendation algorithm we used real world dataset. In the future work more experiments will be carried out in a larger scale datasets and we will include more latent features to study the performance and scalability of the proposed approach in a larger scale. The recommendation process will be improved by integrating context-aware techniques in the approach.
- Service reputation is one of the important parameters of services and plays an important role in recommendation and selection of services. We are planning to provide a dynamic method for service calculation. In addition of the reputation calculation method, we will utilize reputation real-time data to evaluate parameters of consumer experience which measures consumer experience with a service. Finally integrate the reputation method with our recommendation method in both atomic and composite service recommendation.

- Adapting our decentralized service discovery and selection approach to Online Social Networks (OSN) environment, which becomes a popular internet applications where people communicate and share services in real time environment. Since OSN is a very dynamic environment, the discovery and selection algorithm should maintain and update real time knowledge independent of the history records.

References

[1] J. Zhang, J. Zhang and H. Cai., *Services computing. In Springer and Tsinghua University Press,* 2007.

[2] C. Petrie and C.Bussler, “Service Agents and Virtual Enterprises: A survey,” *IEEE International computing,* vol. 7, no. 4, pp. 68-78, 2003.

[3] J. Garofalakis, Y. Panagis, E. Sakkopoulos and a. A. Tsakalidis, “Web service discovery mechanisms: Looking for a needle in a haystack? I,” in *n In: International Workshop on Web Engineering*, 2004.

[4] J. Breese, D. Heckerman and C. Kadie, “Empirical Analysis of Predictive Algorithms for Collaborative Filtering,” in *in Proceedings of 14th Annual conference* , 1998.

[5] M. Gong, Z. Xu and L. Xu, “Recommending Web Service Based on User Relationships and Preferences,” in *IEEE 20th International Conference on Web Services*, 2013.

[6] Z. Zheng, H. Ma, M. R. Lyu and I. King, “Wsrec: A collaborative filtering based web service recommender system,” in *in Proceedings of IEEE International Conference on Web Services (ICWS 2009)*, 2009.

[7] T. M, J. Y and L. J, “Location-Aware Collaborative Filtering for QoS-Based Service Recommendation,” in *19th IEEE International Conference on Web Services (ICWS2012)*, 2012.

[8] Y. Jiang, J. Liu, M. Tang and X. Liu, “An Effective Web Service Recommendation Method Based on Personalized Collaborative Filtering,” in *18th IEEE International Conference on Web Services (ICWS2011)*, 2011.

[9] P. Lops, M. Gemmis and G. Semeraro, “Content-based Recommender Systems: State of the Art and Trends,” in *Recommender Systems Handbook*, Springer US, 2011, pp. 73-105.

[10] A. Popescul, L. Ungar, D. Pennock and S. Lawrence, “Probablistic Models for Unified Collaborative and Content-based Recommendation in Sparse-Data Environments,” in *in Proceedings of the 17th International Conference on Uncertainty in Artificial Intelligence (UAI’01)*, 2001.

[11] Routledge, The Internet of Things: From RFID to the Next-Generation Pervasive Networked Systems (Hardback), 2008.

[12] Z. Zheng, H. M, M. Lyu and I. King, “QoS-Aware Web Service Recommendation by Collaborative Filtering,” *IEEE Transactions on Service Computing (TSC),* vol. 4, no. 2, pp. 140-152, 2011.

[13] R. Jin, J. Chai and L. Si, “An Automatic Weighting Scheme for Collaborative Filtering,” in *in Proceedings of 27th International ACM SIGIR Conference* , 2004.

[14] G. Alonso, Web Services: Concepts, Architectures and Applications, Springer Verlag, 2004.

[15] T. Erl., ervice-oriented architecture : a field guide to integrating XML and Web services, Prentice Hall PTR, 2004.

[16] Gartner, Web Services, 2012.

[17] L. Li and I. Horrocks, “A Software Framework for Matchmaking based on Semantic Web Technologies,” in *In Proceedings of 12th International Conference on World Wide Web*, 2003.

[18] I.-H. Cho, C. Univ., J. D. McGregor and L. Krause, “A protocol based approach to specifying interoperability between objects,” in *In Proceeding of 26th Technology of Object-Oriented Languages*, 1998.

[19] k. Sycara, S. Widoff, M. Klusch and J. Lu, “Larks: Dynamic Matchmaking Among Heterogeneous Software Agents in Cyberspace,” *Autonomous Agents and Multi-Agent Systems,* vol. 5, no. 2, pp. 173-203, 2002.

[20] Y. Zhang, Z. Zheng and M. Lyu, “WSExpress: A QoS-Search Engine for Web Services,” in *Proceedings of 8th IEEE International Conference on Web Services (ICWS 2010)*, 2010.

[21] Z. Zheng, X. Wu and Y. Zhang, “QoS ranking prediction for cloud services,” *IEEE Transactions on Parallel and Distributed Systems,* vol. 24, no. 6, pp. 1213-1222, 2013.

[22] D. D'Mello, V. Ananthanarayana and T. Santhi, “A QoS Broker Based Architecture for Dynamic Web Service Selection,” in *Modeling & Simulation, 2008. AICMS 08. Second Asia International Conference*, 2008.

[23] M. Tian, A. Gramm, H. Ritter and J. Schiller, “Efficient Selection and Monitoring of QoS-Aware Web Services with the WS-QoS Framework,” in *In Proceedings. IEEE/WIC/ACM International Conference on Web Intelligence* , 2004.

[24] A. Jøsang, R. Ismail and C. Boyd, “A survey of trust and reputation systems for online service provision,” *Decision Support System,* vol. 43, no. 2, pp. 618-644, March 2007.

[25] P. Resnick and R. Zeckhauser, “Trust among strangers in internet transactions: Empirical analysis of eBay' s reputation system,” *The Economics of the Internet and E-commerce (Advances in Applied Microeconomics), Emerald Group Publishing Limited,* vol. 11, pp. 127-157, 2002.

[26] J. Yao, W. Tan, S. Nepal, S. Chen, J. Zhang, D. D. Roure and C. Goble, “ReputationNet: A Reputation Engine to Enhance ServiceMap by Recommending Trusted Services,” in *IEEE Ninth International Conference on Services Computing (SCC)*, 2012.

[27] G. Zacharia and P.Maes, “Trust management through reputation mechanisms,” *Applied Artificial Intelligence,* vol. 14, pp. 881-907, 2000.

[28] Y. Wu, C. Yan, Z. Ding, G. Liu, P. Wang, C. Jiang and M. Zhou, “A Novel Method for Calculating Service Reputation,” *IEEE Transactions on Automation Science and Engineering,* vol. 10, no. 3, pp. 634-642, July 2013.

[29] D. A. Menasce, “QoS issues in Web services,” *IEEE Internet Computing,* vol. 6, no. 6, pp. 72 - 75, 2002.

[30] M. Platenius, M. Detten, S. Becker, W. Schäfer and G. Engels, “A survey of fuzzy service matching approaches in the context of on-the-fly computing,” in *CBSE '13 Proceedings of the 16th International ACM Sigsoft symposium on Component-based software engineering*, NY, USA, 2013.

[31] G. Dobson, R. Lock and I. Sommerville, “QoSOnt: a QoS ontology for service-centric systems,” in *EUROMICRO Conference on Software Engineering and Advanced Applications*, 2005.

[32] I. Toma, F. D. Paoli and D. Fensel, “On Modelling Non-Functional Properties of Semantic Web Services,” in *Handbook of Research on Service-Oriented Systems and Non-Functional Properties: Future Directions*, IGI Global, 2012, pp. 61-85.

[33] Y. Badr, “Framework for web service selection based on non-functional properties,” *International Journal of Web Service Practices,* vol. 3, no. 2, pp. 94-109, 2008.

[34] M. Tian, A. Gramm, T. aumowicz, H. Ritter and a. J. Schiller, “A concept for QoS integration in Web services,” in *Web information systems engineering workshops*, Washington DC, 2003.

[35] T. Saaty, “How to make a decsion: the analytic hierarchy process,” *European Journal of Operational Research,* vol. 48, no. 1, pp. 9-26, 1990.

[36] S. Carg, S. Versteeg and R. Buyya, “SMIcloud: a framework for comparing and ranking cloud services,” in *In proceedings of the 4th IEEE/ACM International Conference on Utility and Cloud Computing (UCC'11)*, Melbourne, Australia, 2011.

[37] M. Buyukyazici and M. Sucu, “The analytic hierarchy and analytic network processes,” *Hecettepe Journal of Mathematics and Statistics,* vol. 32, pp. 65-73, 2003.

[38] M. Menzel, M. Schonherr and S. Tai, “(MC2) 2:criteria, requirements and a software prototype for cloud infrastructure decisions,” *Software: Practice and Experience,* 2011.

[39] w. Zeng, Y. Zhao and J. Zeng, “Clouod Service and service selection algorithm research,” in *In proceedings of the first ACM/SIGEVO Summit on Genetic and Evolutionary Computation*, 2009.

[40] J. Fulop, “Introduction to decision making methods,” Laboratory of Operations Research and Decision Systems, Computer and Automation Institute, Hungary, 2005.

[41] L. Zeng, B. Benatallah, A. Ngu, M. Dumas, J. Kalagnanam and H. Chang, “QoS-aware middleware for Web services composition,” *IEEE Transactions on Software Engineering,* vol. 30, no. 5, pp. 311 - 327, 2004.

[42] Y. T. and L. K. -J., “The design of QoS broker algorithms for QoS-capable Web services,” in *IEEE International Conference on e-Technology, e-Commerce and e-Service*, 2004.

[43] C. Chang, P. Liu and J. Wu, “Probability-Based Cloud Storage Providers Selection Algorithms with Maximum Availability,” in *41st International Conference on Parallel Processing*, 2012.

[44] S. Esmaeilsabzali and K. Larson, “Service Allocation for Composite Web Services Based on Quality Attributes,” in *Seventh IEEE International Conference on E-Commerce Technology Workshops*, 2005.

[45] Y. Shen and Y. Fan, “Cooperative mixed strategy for service selection in service oriented architecture,” in *IEEE International Conference on Systems, Man and Cybernetics*, 2007.

[46] D. Guinard, V. Trifa., S. Karnouskos, P. Spiess and D. Savio, “Interacting with the SOA-Based Internet of Things: Discovery, Query, Selection, and On-Demand Provisioning of Web Services,” vol. 3, no. 3, pp. 223-235, 2010.

[47] Y. Wu, C. Yan, L. Liu, Z. Ding and C. Jiang, “An Adaptive Multilevel Indexing Method for Disaster Service Discovery,” *IEEE Transactions on Computers,* vol. 64, no. 9, September 2015.

[48] C. Chang, S. Srirama and J. Mass, “A Middleware for Discovering Proximity-Based Service-Oriented Industrial Internet of Things,” in *2015 IEEE International Conference on Services Computing (SCC)*, 2015.

[49] J. Lindenberg, W. Pasman, K. Kranenborg, J. Stegeman and M. Neerincx, “Improving service matching and selection in ubiquitous computing environments: a user study,” *Personal Ubiquitous Computing,* vol. 11, no. 1, pp. 59-68, October 2006.

[50] S. Agarwal and R. Studer, “Automatic matchmaking of Web services,” in *in Proceedings of the IEEE International Conference on Web Services*, 2006.

[51] L. Liu, N. Antonopoulos, M. Zheng, Y. Zhan and Y. Ding, “A Socio-ecological Model for Advanced Service Discovery in Machine-to-Machine Communication Networks,” *ACM Transactions on Embedded Computing,* 2016.

[52] P. P. a. B. Pernici, “URBE:Web ServiceRetrieval Based on similarity evaluation,” *IEEE Trans. Know. Data Eng.,* vol. 21, no. 11, pp. 1629-1642, November 2009.

[53] M. Klusch, B. Fries and a. K. Sycara, “OWLS-MX: A Hybrid Semantic Web Service Matchmaker for OWL-S Services,” *Web Seman., Sci., Serv. AgentsWorld Wide Web,* vol. 7, no. 2, pp. 121-133, April 2009.

[54] K. Chao, M. Younas, C. Lo and T. Tan, “Fuzzy matchmaking for Web services,” in *19th International Conference on Advanced Information Networking and Applications*, 2005.

[55] C.Huang, K. Chao and C. Lo, “A moderated fuzzy matchmaking for Web services,” in *The Fifth International Conference onComputer and Information Technology*, 2005.

[56] L. Ngan and R. Kanagasabai, “Semantic Web service discovery: state-of-the-art and research challenges,” *Personal and Ubiquitous Computing,* vol. 17, no. 8, pp. 1741-1752, December 2013.

[57] K.Kritikos and a. D. Plexousakis, “Novel Optimal and Scalable Nonfunctional Service Matchmaking Techniques,” *IEEE Transactions on Services Computing,* vol. 7, no. 4, pp. 614-627, October- December 2014.

[58] Y. Liu, A. Ngu and L. Zeng, “QoS Computation and Policing in Dynamic Web Service Selection,” in *Proceeding 13th International ConferenceWorld WideWeb*, 2004.

[59] J. C. Estrella, R. H. Santana, M. Santana and S. Bruschi, “WSARCH: A Service-Oriented Architecture with QoS,” in *Handbook of Research on Service-Oriented Systems and Non-Functional Properties: Future Directions*, IGI Global, 2012, pp. 352-380.

[60] K. Kritikos and D. Plexousakis, “Towards Optimal and Scalable Non-functional Service Matchmaking Techniques,” in *IEEE 19th International Conference on Web Services (ICWS)*, 2012.

[61] B. Pernici and S. Siadat, “Selection of Service Adaptation Strategies Based on Fuzzy Logic,” in *IEEE World Congress onServices (SERVICES)*, 2011.

[62] M. AlMulla, K. Almatori and H. Yahyaoui, “A QoS-Based Fuzzy Model for Ranking Real World Web Services,” in *IEEE International Conference on Web Services (ICWS)*, 2011.

[63] B. Pernici and S. Siadat, “A fuzzy service adaptation based on QoS satisfaction,” in *In proceeding of: Advanced Information Systems Engineering - 23rd International Conference*, London-UK, 2011.

[64] S. Silas, E. B. Rajsingh and K. Ezra, “Efficient Service Selection Middleware using ELECTRE Methodology for Cloud Environments,” vol. 11, no. 7, pp. 868-875, 2012.

[65] P. Saripalli and G. Pingali, “MADMAC: multiple attribute decision methodology for adoption of clouds,” in *IEEE International Conference on Cloud Computing*, 2011.

[66] W. Lin, W. Dou, Z. Xu and a. J. Chen, “A QoS-aware service discovery method for elastic cloud computing in an unstructured peer-to-peer network,” *Concurrency and Computation,* vol. 25, no. 13, pp. 1843-1860, February 2013.

[67] L. Zhao, Y. Ren, M. Li and K. Sakurai, “Flexible service selection with user-specific QoS support in service-oriented architecture,” *Network and Computer Applications,* vol. 35, no. 3, pp. 962-973, May 2012.

[68] D. M. Blei, A. Y. Ng and M. I. Jordan, “Latent Dirichlet Allocation,” *Journal of Machine Learning Research,* vol. 3, pp. 993-1022, 2003.

[69] D. M. Blei, “ Probabilistic topic models,” *Communications of the ACM,* vol. 55, no. 4, p. 77–84, 2012.

[70] N. T. a. S. Karunasekera, “Automatic measurement of a qos metric for web service recommendation,” in *in Proceedings of 2005 Australian Software Engineering Conference*, 2005.

[71] G. Linden, B. Smith and J. York, “Amazon.com Recommendation: Item-to-Item Collaborative Filtering,” in *IEEE Internet Comput.*, 2003.

[72] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom and J. Riedl, “GoupLens: An Open Architecture for Collaborative Filtering of Netnews,” in *in Proceedings Conference CSCW*, 1994.

[73] H. Ma, I. King and L. M.R., “Effective Missing Data Prediction for Collaborative Filtering,” in *In Proc. of 30th International ACM SIGIR Conference*, 2007.

[74] X. Chen, X. Liu, Z. Huang and H. Sun, “Regionknn: A scalable hybrid collaborative filtering algorithm for personalized web service recommendation,” in *in Proceedings of IEEE International Conference on Web Services (ICWS 2010)*, 2010.

[75] B. Sarwar, G. Karypis, J. Konstan and J. Reidl, “Item-Based Collaborative Filtering Recommendation Algorithm,” in *in Proceedings of 10th International WWW Conference*, 2001.

[76] G. Xue, C. Lin, Q. Yng, W. Xi, H. Zeng, Y. Yu and Z. Chen, “Scalable Collaborative Filtering Using Cluster-Based Smoothing,” in *in Proceedings of 28th International ACM SIGIR Conference*, 2005.

[77] A. Jennings and H. Higuchi, “A User Model Neural Network for a Personal News Service,” *User Modeling and User-Adapted Interaction,* vol. 3, no. 1, pp. 1-25, 1993.

[78] T. Hofmann, “Latent Semantic Models for Collaborative Filtering,” *ACM Transaction for Info. Syst.,* vol. 22, no. 1, pp. 89-115, 2004.

[79] T. Hofmann, “Latent Semantic Models via Gaussian Probabilistic Latent Semantic Analysis,” in *in Proceedings of 26th International ACM SIGIR Conference*, 2003.

[80] T. Hofmann, “Probablistic Latent Semantic Analysis,” in *in Proceedings of International Conference of UAI*, 1999.

[81] H. Wu, Y. Wang and X. Cheng, “Incremental Probabilistic Latent Semantic Analysis for Automatic Question Recommendation,” in *in Proceedings of ACM Confereence of RecSys*, 2008.

[82] T. Griffiths and M. Steyvers, “Finding Scientific Topics,” in *National Academy of Science*, 2004.

[83] X. Wei and W. B. Croft, “LDA-based document models for ad-hoc retrieval,” in *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, Seattle, Washington, USA, 2006.

[84] Q. Liu, E. Chen, H. Xiong, C. Ding and J. Chen, “Enhancing Collaborative Filtering by User Interest Expansion via Personalized Ranking,” *IEEE Transactions on Systems, Man , and Cybernetics,* vol. 42, no. 1, pp. 218-232, February 2012.

[85] G. Adomavicius and A. Tuzhilin, “Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions,” *IEEE Transactions on Knowledge and Data Engineering,* vol. 17, no. 6, pp. 734 - 749, 2005.

[86] A. Schein, A. Popescul, L. Ungar and D. Pennock, “Methods and Metrics for Cold-Start Recommendations,” in *in Proceedings of 25th Annual International ACM Sigir Conference* , 2002.

[87] H. Ahn, “A New Similarity Measure for Colalborative Filtering to Alleviate the New User Cold-Starting Problem,” *Information Science,* vol. 178, no. 1, pp. 37-51, 2008.

[88] M. Ahmed, L. Liu, J. Hardy and a. B. Yuan, “An Efficient Algorithm for Partially Matched Web Services Based on Consumer’s QoS Requirements,” in *in IEEE/ACM 7th Internalional Conference on Utility and Cloud Computing (UCC)*, London, 2014.

[89] M. Klusch, B. Fries and a. K. Sycara, “Automated semantic web service discovery with OWLS-MX,” in *in AAMAS '06 Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, 2006.

[90] k. Kritikos and K. Plexousakis, “Novel Optimal and Scalable Nonfunctional Service Matchmaking Techniques,” *IEEE Transactions on Services Computing,* vol. 7, no. 4, pp. 614-627, 2014.

[91] Y. Chen, J. Huang and a. C. Lin, “Partial Selection: An Efficient Approach for QoS-Aware Web Service Composition,” in *2014 IEEE International Conference on Web Services (ICWS)*, 2014.

[92] D. Bachlechner, K. Siorpaes, D. Fensel and a. I. Toma, “Web service discovery - a reality check,” in *in In Proceedings of the 3rd European Semantic Web Conference*, 2006.

[93] I. Adriana and F. Ian, “fully decentralized resource discovery in grid environments,” in *in proceedings of the second international workshop on grid computing*, 2001.

[94] t. Sanya, K. Takahiro, K. Kenji, H. Hiroki and Y. Toshitsugu, “A time-to-live based reservation algorithm on fully decentralized resource discovery in Grid computing,” *Parallel Computing,* vol. 31, no. 6, p. 29–43, 2005.

[95] L. Hong and L. Lu, “A decentralized resource discovery based on keywords combinations and node clusters in knowledge grid,” in *in Proceedings of the intelligent computing 3rd international conference on advanced intelligent computing theories and applications*, Qingdao, China, 2007.

[96] B. Amos, M. Apostolos and H. Béat, “Enabling efficient information discovery in a self-structured grid,” *Future Generation Computer Systems,* vol. 26, no. 6, p. 838–46, 2010.

[97] K. Taskin and L. Daniel, “Design and analysis of a distributed grid resource discovery protocol,” *Cluster Computing,* vol. 5, no. 1, p. 37–52, 2012.

[98] Q. He, J. Yan, Y. Yang, R. Kowalczyk and a. H. Jin, “A Decentralized Service Discovery Approach on Peer-to-Peer Networks,” *IEEE TRANSACTIONS ON SERVICES COMPUTING,* vol. 6, no. 1, pp. 64-75, 2013.

[99] I. Stoica, R. Morris, D. Karger, M. Kaashoek and a. H. Balakrishnan, “Chord: A scalable peer-to-peer lookup service for internet applications,” in *in SIGCOMM '01 Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*, New York, USA, 2001.

[100] A. Rowstron and a. P. Druschel, “Pastry: Scalable, Decentralized Object Location, and Routing for Large-Scale Peer-to-Peer Systems,” in *in IFIP/ACM International Conference on Distributed Systems Platforms*, Germany, 2001.

[101] P. Maymounkov and a. D. Mazières, “Kademlia: A Peer-to-Peer Information System Based on the XOR Metric,” in *in First InternationalWorkshop, IPTPS*, 2002.

[102] I. Antony, R. T. and P. Druschel, “Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems,” in *in In Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms*, 2001.

[103] S. Ratnasamy, P. Francis, M. Handley, R. Karp and a. S. Shenker, “A scalable content-addressable network,” in *In Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM '01)*, New York, 2001.

[104] E. del Val, M. Rebollo and a. V. Botti, “Enhancing decentralized service discovery in open service-oriented multi-agent systems,” *Autonomous Agents and Multi-Agent Systems,* vol. 28, no. 1, pp. 1-30, 2014.

[105] “Health and Social Care Information Centre,” 2014.

[106] 1.-. J. R. a. X. Su, “A survey of automated web service composition methods,” in *in Proceedings of the First international workshop on semantic web services and Web process composition*, San Diego, California, USA, 2004.

[107] Y. Zhang, Z. Zheng and a. M. Lyu, “WSPred: A Time-Aware Personalized QoS Prediction Framework for Web Services,” *Software Reliability Engineering (ISSRE),* pp. 210-219, 2011.

[108] L. Zeng, B. Benatallah, M. Dumas, J. Kalagnanam and Z. Sheng, “Quality driven web services composition,” in *In Proceedings of the 12th international conference on World Wide Web (WWW '03)*, New York, 2003.

[109] k. K. J. Jarvelin, “Cumulated Gain-based evaluation of IR techniques,” *ACM Transaction on Information System,* pp. 422-446, 2002.

[110] S. Dimitrios, S. Dimitris, K. Verena and S. and Timos, “Efficient Semantic Web Service Discovery in Centralized and P2P Environments,” in *The Semantic Web - ISWC 2008*, Springer Berlin Heidelberg, 2008, pp. 583-598.

[111] J. R. a. X. Su, “A survey of automated web service composition methods,” in *in Proceedings of the First international workshop on semantic web services and Web process composition*, San Diego, California, USA, 2004.

[112] Q. Xie, Z. Zheng, L. Liu and M. Cui, “Correlation-based Top-K Recommendation for Web Services,” in *2015 IEEE International Conference on Pervasive Intelligence and Computing*, Liverpool, 2015.

[113] M. Gori, A. Pucci, V. Roma and a. I. Siena, “Itemrank: A random-walk based scoring algorithm for recommender engines,” in *in Proceedings of the 20th International Joint Conference on Artificial Intelligence*, 2007.

[114] S. Brin and L. Page, “The Anatomy of a Large-Sclae Hypertextual Web Search Engine,” in *Roceedings of World Wide Web Conference*, 1998.

[115] G. Salton, “Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer,” Boston, Addison-Wesley Longman Publishing Co., 1989.

[116] X. Phan, L. Nguyen and S. Horiguchi, “ Learning to Classify Short and Sparse Text & Web with Hidden Topics from Large-scale Data Collections,” in *In Proc. of The 17th International World Wide Web Conference (WWW 2008)*, Beijing, China , 2008.

[117] I. Biro, J. Szabo and A. Benczur, “ Latent Dirichlet Allocation in Web Spam Filtering,” in *In Proc. of The Fourth International Workshop on Adversarial Information Retrieval on the Web, WWW 2008*, Beijing, China, April 2008.

[118] W. Chen, J. C. Chu, J. Luan and H. Bai, “Collaborative filtering for orkut communities: Discovery of user latent behaviour,” in *in Proceedings of 18th internation conference of WWW*, 2009.

[119] X. Chen, Z. Zheng and X. Liu, “Personalized QoS-Aware Web Service Recommendation and Visualization,” *IEEE Transactions on Services Computing ,* vol. 6, no. 1, pp. 35-47, 2013.

[120] A. Paterek, “Improving regularized singular value decomposition for collaborative filtering,” in *in Proceedings of KDD Cup and Workshop*, San Jose, California, USA, August 2007.

[121] W. Cohen, R. Schapire and Y. Singer, “Learning to Order Things,” in *In the Proceedings of the NIPS*, Denver, CO, USA, 1997.

[122] M. Chen, Y. Ma, B. Hu and L. Zhang, “A Ranking-oriented Hybrid Approach to QoS-aware Web Service Recommendation,” in *IEEE International Conference on Service Computing*, 2015.

[123] W. Lin, W. Dou, Z. Xu and a. J. Chen, “A QoS-aware service discovery method for elastic cloud computing in an unstructured peer-to-peer network,” *Concurrency and computation,* vol. 25, no. 13, pp. 1843-1860, 2013.

[124] D. Skoutas, D. Sacharidis, V. Kantere and a. T. Sellis, “Efficient Semantic Web Service Discovery in Centralized and P2P Environments,” in *in The Semantic Web - ISWC 2008*, Berlin Heidelberg, Springer, 2008, pp. 583-598.

[125] P. Lazarsfeld, “Friendship as a social process: A substantive and methodological analysis,” in *Freedom and control in Modern Society*, 1954.

[126] M. McPherson, L. Smith-lovin and J. Cook, “Birds of a Feather: Homophily in Social Networks,” *Annual review of sociology,* vol. 27, pp. 415 -444, 2001.

[127] January 2008. [Online]. Available: http://www.uoguelph.ca/~qmahmoud/qws/.

[128] T. Yu, Y. Zhang and K.-J. Lin, “Efficient algorithms for Web services selection with end-to-end QoS constraints,” *ACM Transactions on the Web,* vol. 1, no. 1, May 2007.

[129] R. Krummenacher, M. Hepp, A. Polleres, C. Bussler and a. D. Fensel, “www or what is wrong with Web services discovery,” in *in Proceedings of the Third European Conference on Web Services, ser. ECO S ’05*, Washington DC, USA, 2005.

[130] H. Yu and S.Marganiec, “A backwards composition context based service selection approach for service composition,” in *IEEE International Conference on Services Computing*, 2009.

[131] X. Wang, T. Vitvar, M. Kerrigan and a. I. Toma, “A qos-aware selection model for semantic web services,” in *Proceedings of the 4th international conference on Service-Oriented Computing (ICSOC'06)*, 2006.

[132] Q. Ma, H. Wang, Y. Li, G. Xie and F. Liu, “A Semantic QoS-Aware Discovery Framework for Web Services,” in *Proceedings of the 2008 IEEE International Conference on Web Services (ICWS '08)*, 2008.

[133] A. Cibran, Maria, B. Verheecke, W. Vanderperren, D. Suvee and a. V. Jonckers, “Aspect-Oriented programming for dynamic web service selection, Integration and Management,” *Springer Science+Business Media,* pp. 211-242, 2007.

[134] P. Plebani and F. Ramoni, “A Quality Driven Web Service Selection Model,” in *Handbook of Research on Service-Oriented Systems and Non-Functional Properties: Future Directions*, IGI Global, 2012, pp. 142-164.

[135] Y. Mou, J. Cao, S. Zhang and J. Zhang, “Interactive Web Service Choice-Making Based on Extended QoS Model,” in *Computer and Information Technology*, 2005.

[136] L. Zhang, J. Zhang and a. H. Cai, *Services computing,* 2007.

[137] A. Averbakh, D. Krause and D. Skoutas, “Exploiting user feedback to improve semantic Web service discovery,” in *Proceedings of the 8th International Semantic Web Conference. Springer Verlag*, 2009.

[138] A. Srivastava, Abhishek and P. G. Sorenson, “Service Selection Based on Customer Preferences of Non-Functional Attributes,” in *Handbook of Research on Service-Oriented Systems and Non-Functional Properties: Future Directions*, IGI Global, 2012, pp. 280-296.

[139] G. Dobson, S. Hall and G. Kotonya, “A Domain-Independent Ontology for Non-Functional Requirements,” in *IEEE International Conference on e-Business Engineering ICEBE*, 2007.

[140] I. Sora, D. Todinca and C. Avram, “Translating user preferences into fuzzy rules for the automatic selection of services,” in *5th International Symposium on Applied Computational Intelligence and Informatics SACI '09*, 2009.

[141] E. Giallonardo and E.Zimeo, “More Semantics in QoS Matching,” in *In Proceedings of the IEEE International Conference on Service-Oriented Computing and Applications (SOCA '07)*, 2007.

[142] C. Shi, D. Lin, Ishida and Toru, “User-Centered QoS Computation for Web Service Selection,” in *IEEE 19th International Conference on Web Services (ICWS)*, 2012.

[143] L. A. Zadeh, “Fuzzy Sets,” in *Information and control*, 1965.

[144] D. K. a. I. Paraskakis, “Combining SAWSDL, OWL-DL and UDDI for Semantically Enhanced Web Service Discovery,” in *Proceedings of the 5th European Semantic Web Conference (ESWC’08)*, 2008.

[145] J. Y. a. G. Zhou, “Web Service Discovery and Dynamic Invocation based on UDDI/OWL-S,” in *Proceedings of the Third International Conference on Business Process Management (BPM’05)*, 2005.

[146] H. Zheng, J. Yang, W. Zhao and A. Bouguettaya, “QoS Analysis for Web Service Compositions Based on Probabilistic QoS,” in *in Proceedings of the 9th International Conference on Service Oriented Computing (ICSOC 2011)*, 2011.

[147] M. B. B. a. M. F. Nowlan, “A Web Service Recommender System Using Enhancing Syntactical Matching,” in *in Proceedings of IEEE International Conference on Web Services (ICWS’07)*, 2007.

[148] M. F. Porter, “An Algorithm for Suffix Stripping,” *In: Program 1980,* vol. 14, no. 3, pp. 130-137.

[149] J. O'Sullivan, D. Edmond and D. Hofstede, “What's in a service?,” *Distributed and Parallel Databases,* vol. 12, no. 2, pp. 117-133, September 2002.