2009

Ant inspired scalable peer selection in ontology-based service composition

Shuai Yuan
University of Wollongong, sy242@uow.edu.au

Jun Shen
University of Wollongong, jshen@uow.edu.au

Aneesh Krishna
Curtin University of Technology, aneesh@uow.edu.au

Ant inspired scalable peer selection in ontology-based service composition

Abstract

This work focuses on proposing a method of effectively dealing with P2P-based service selection and composition, especially when handling a large number of peers along with their diverse qualities. The QoS-aware peer selection is one of the major challenges faced in order to guarantee the success and enhance performance of distributed computing. Since many peer candidates provide overlapping or identical functionalities, though with different QoS evaluations, selections need to be rapidly conducted to determine which peers are suitable to join in the requested composite service. The main contribution of this paper is proposing a P2P-based service selection model, in which peer's non-functional properties are modeled with Web service modelling ontology (WSMO), and where ant colony optimisation (ACO) technique is adopted to facilitate and enhance the QoS-aware peers' composition. We present experimental results to illustrate the effectiveness and feasibility of the proposed method.

Disciplines

Physical Sciences and Mathematics

Publication Details

Yuan, S., Shen, J. & Krishna, A. (2009). Ant inspired scalable peer selection in ontology-based service composition. Proceedings of the Fifth World Congress on Services (Services 2009, Part II) (pp. 95-102). Los Alamitos, USA: IEEE.

Ant Inspired Scalable Peer Selection in Ontology-Based Service Composition

Shuai Yuan
School of Information Systems and Technology
Faculty of Informatics, University of Wollongong, NSW 2522, Australia
shuai.yuan@hotmail.com

Jun Shen
School of Information Systems and Technology
Faculty of Informatics, University of Wollongong, NSW 2522, Australia
jshen@uow.edu.au

Aneesh Krishna
Department of Computing
Faculty of Science & Engineering, Curtin University of Technology, WA 6845, Australia
A.Krishna@curtin.edu.au

Abstract

This work focuses on proposing a method of effectively dealing with P2P-based service selection and composition, especially when handling a large number of Peers along with their diverse qualities. The QoS-aware Peer selection is one of the major challenges faced in order to guarantee the success and enhance performance of distributed computing. Since many Peer candidates provide overlapping or identical functionalities, though with different QoS evaluations, selections need to be rapidly conducted to determine which Peers are suitable to join in the requested composite service. The main contribution of this paper is proposing a P2P-based service selection model, in which Peer's non-functional properties are modeled with Web Service Modelling Ontology (WSMO), and where Ant Colony Optimisation (ACO) technique is adopted to facilitate and enhance the QoS-aware Peers' composition. We present experimental results to illustrate the effectiveness and feasibility of the proposed method.

Keywords
P2P, QoS, WSMO, ACO, service composition

1. Introduction

P2P-based service selection and composition are viewed as important issues in dynamic decentralised e-service application and workflow [15]. In recent years, it has been apparent that the notion of distributiveness, dynamics and heterogeneity of services has become extremely important to both service requestors and service providers, thus Quality of Service (QoS) evaluation is accepted as an effective method to identify service providers' capability and performance so as to enhance the service composition for service clients over the real distributed service network. Furthermore, due to the complexity of QoS metrics, well-defined service description has not been thoroughly addressed. Most of the current work focuses on the definition of QoS ontology, vocabulary or measurements and to a lesser extent on a uniform evaluation of qualities, however, few works addressed on the model of P2P-based non-functional properties in ontology.

With regard to P2P-based service composition, to find the solution of such a QoS–aware composition problem is NP–hard. This is mainly because the Peer selection process over a scalable distributed network is difficult to make right decisions on which Peers are suitable for the requested services based on Peers' heterogeneous performances. Figure 1, depicts a simple example of the Peer selection process. Atomic Web Services (AWS) {AWS_1, AWS_2, AWS_3, AWS_4,} are requested to be executed in a sequential workflow, and each of them has got two possible Peer candidates amongst eight Peers, and every Peer's QoS for invoking the relevant atomic services is different from one another, in terms of Response Time and Cost. In the example, it may be not very difficult to find an optimal solution.

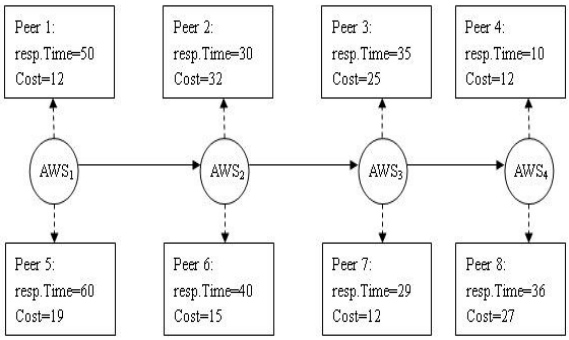

Figure 1: Example of workflow with bindings between Peers and atomic Web services

In reality, however, we are facing a much larger number of atomic services and Peer candidates as well as multiple QoS attributes (e.g. Response Time, Cost, Availability, Reliability, Reputation and etc.), to obtain the optimal Peers combination for the requested atomic services. This is by any means not an easy task. The searching process would be exhaustive and can be very time consuming provided a composition is related with a large group of Peers. Although past research efforts have looked at many pragmatic ways to effectively deal with the composition of Web services into executable workflows (e.g. [14, 16]), concern still remains as to how to optimise the P2P-based service selection and composition in terms of searching for solutions more quickly and effectively.

In this paper, we extend the Web Service Modelling Ontology (WSMO) description for the multiple and diverse QoS attributes for Peers, in order to facilitate the service

978-0-7695-3812-9/09 $26.00 © 2009 IEEE
DOI 10.1109/SERVICES-2.2009.15

selection process. Furthermore, we introduce Ant Colony Optimisation (ACO) approach and adopt the method to improve the selection process and composition performance comparing with classic integer programming.

The paper is structured as follows. Section 2 briefly presents background and related work on service description and composition. Section 3 introduces the approach of WSMO modelling and ACO approach for selection. Section 4 presents experimental results of proposed method. Conclusions remarks and future work is presented in Section 5.

2. Background

2.1 P2P-based Service Network Structure

Peer-to-peer (p2p) network is one of the most innovative information distribution architecture emerging in recent years. In this work, a p2p network is defined as the network of nodes where every node is equally in being responsible to share resources and the overhead of running the network.

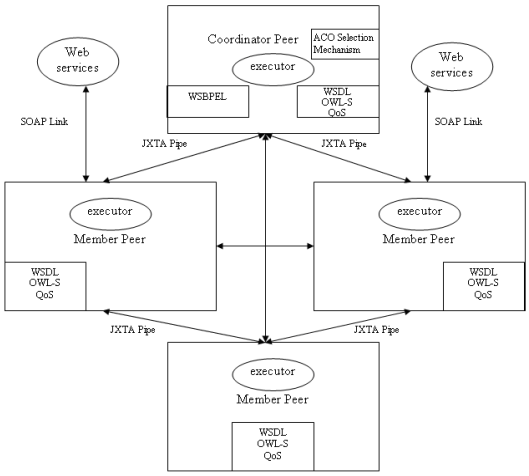

Figure 2: UOW-SWS System Model

There are many good P2P prototypes proposed by various researchers around the world. In this paper, we shall begin by introducing our own UOW-SWS as an example to describe the basic mechanism of P2P-based service prototype. UOW-SWS is a JXTA-based [6] Peer-to-Peer workflow information system upgraded from SwinDeW-B [10], which was designed and developed to overcome the problems like poor performance, poor scalability, unsatisfactory system openness, and lack of support for incomplete process. With Ontology extension, UOW-SWS has been functionally extended to incorporate ACO approach so as to facilitate Web services selection via QoS information.

Figure 2 depicts the overall architecture of the UOW-SWS's composite web service engines. The engines are composed of a P2P-based system built in the JXTA framework. BPEL [11] is chosen as the language to orchestrate composite web services. A composite web service is described by a BPEL file and a set of WSDL files. By processing the files a coordinator Peer obtains the knowledge of which activities are to be performed and the temporal order of performing them to complete a composite web service. Communications between the Peers and their web services are via the SOAP protocol. Each Peer appointed for a task by the coordinator in a group can invoke the required web service from service providers, and there is no difference between coordinator and Peer after the assigning of tasks by the coordinator, so they are able to cooperate actively in the P2P network to execute the composition and invocation of web services in the same manner.

2.2 Problem Formulation

The P2P-based service selection problem's basic features will be defined mathematically as discussed in the following paragraph.

Let all Peer candidates be in a set as: $P=\{P_1, P_2, \ldots, P_Z\}$, and all atomic services of a request be in set: $S=\{S_1, S_2, \ldots, S_R\}$; Z is the total number of all Peer candidates, and R is the total number of all atomic services for a request. Different Peer has different capabilities to conduct some of those atomic services. For example, a Peer (P_i) has capability of hosting m types of atomic services as: $C^{P_i} = \{S_1^{P_i}, S_2^{P_i}, \ldots, S_m^{P_i}\}$. Furthermore, a Peer ($P_i$) has different qualities when performing an atomic service ($S_j^{P_i} \in C^{P_i}$), in terms of Response Time(RT), Cost(CT), Availability(AV) and Reputation(RP), and the qualities of Peer (P_i) for service ($S_j^{P_i}$) can be set like: $Q_{S_j}^{P_i} = \{RT, CT, AV, RP\}^{S_j^{P_i}}$, where RT, CT, AV, RP are actual values for Response Time, Cost, Availability and Reputation.

For service composition, the main objective is to find a set of feasible Peers which can provide the best or better quality of service. For those QoS factors, therefore, four objectives can be like $min(\Sigma RT)$, $min(\Sigma CT)$, $max(\Pi AV)$ and $max(\Pi RP)$:

$$O_1 = \min(\sum_{\substack{k=1 \\ P_i \in P}}^{R} Q_{S_k}^{P_i}(RT)); \quad O_2 = \min(\sum_{\substack{k=1 \\ P_i \in P}}^{R} Q_{S_k}^{P_i}(CT));$$

$$O_3 = \max(\prod_{\substack{k=1 \\ P_i \in P}}^{R} Q_{S_k}^{P_i}(AV)); \quad O_4 = \max(\prod_{\substack{k=1 \\ P_i \in P}}^{R} Q_{S_k}^{P_i}(RP));$$

In order to consider the four objectives as a whole, we need a proper function (f) to normalise them respectively and then can be set as:

$$F = \min(\frac{f(O_1) + f(O_2)}{f(O_3) + f(O_4)});$$

Subject to:

$$S \equiv \bigcup_{P_i \in P} C^{P_i} \; ; \; Z > 0, R > 0, m > 0.$$

This formulation does not only consider linear aggregation function (e.g. Response Time) but also takes into account non-liner function (e.g. Availability). Therefore, to obtain the optimal solution of such a QoS–aware composition problem is going to be NP–hard.

2.3 Related Work

In recent years, a lot of related works regarding the QoS in Web services have focused on the development of QoS ontology languages and vocabularies, as well as the identification of various QoS metrics and their measurements with respect to semantic Web services. For example, papers [7] and [8] emphasised a definition of QoS aspects and metrics. In [7], all of the possible quality requirements were introduced and divided into several categories, including runtime related, transaction support related, configuration management and cost related, and security related QoS. Both the papers present their definitions and possible determinants. Our previous work [15] presented a first sketch of service description approach, paying more attention on the extraction of the ontological description of services and design of the selection process with OWL-S. With regard to the selection process, the previous prototype has the limitation in terms of dealing with multi-specifications, and it only considers "ResponseTime" as the selection criteria. Selection is not quite realistic for performing effective services composition. On the other hand, this paper extends the description of non-functional properties via modelling-driven WSMO specification, and also presents ACO approach for the coordinator to assemble close-optimal Peers. Moreover, we demonstrate our implementation at different service stages, and prove that our method can reasonably improve the efficiency and adaptation of autonomous P2P-based business process.

QoS–aware Peer and service selection for composition attracts many interesting applications and trials of various methods and search strategies, mostly based on operation research (OR) or artificial intelligence (AI). In general, most of the accepted approaches for effective service composition focus on how to find the optimal/close-optimal combinations with the minimum cost (e.g. time, computation resource constraints), and they are usually dependent upon Integer Programming [1, 16] or Genetic Algorithms [2, 3], or mixed up with some other optimisation strategies [5]. The Integer programming (IP) solutions with regard to dynamically finding the best service combination have been proposed in some recent papers ([1, 16]). These works consider linearity of the constraints and the objective functions, and find the best combination of the concrete services. From the QoS point of view, they're conducted at run time. Zeng et al. [16] essentially focus on the cost, response time, availability and reliability attributes, where logarithmic reductions are used for the multiplicative aggregation functions, and the model is claimed to be extensible with respect to other similarly behaving attributes. On the other hand, Genetic Algorithms (GA) is viewed as a good heuristic method for seeking a close-optimal combination. The work conducted by Cao et al. [2] is a recent development on the application of GA to the service selection problem in the context of Web service composition. Chockalingam et al. [3] described a randomised heuristic method based on the general principles of genetic search strategy, so as to solve the mapping problem, in which parallel tasks are assigned to a multiprocessor to minimise the execution time. As another optimisation approach, ACO has been widely used for the problems such as allocating jobs in robot networks and searching the shortest path, but has rarely been considered for the service selection.

3. Approach Description

Prior to deploying ACO to find optimal or close-optimal solutions for service composition, it is necessary to have a modeling structure (i.e. WSMO). This will help to describe and model non-functional properties of services for selection process, as the effectiveness of the selection relies on explicit information about services. Furthermore, a well-defined service profile model can not only benefit service selection and composition, but also can facilitate service discovery, particularly in a complicated service network.

3.1 WSMO Description for Peer's Non-functional Properties

WSMO [9] aims to create an ontology which can semantically describe a variety of perspectives of the Web services, so as to solve the integration problem. Essentially, WSMO defines four high-level notions which relate to semantic Web services, namely Ontologies, Goals, Mediators and Web services. In this paper, our QoS extension is for non-functional properties in "Web services". In other words, we mainly focus on the consideration of QoS, such as performance, availability, cost of distributed services, etc. The incorporated QoS properties could also be used in parallel with the existing non-functional attributes proposed by other WSMO elements. Thus, it is consistent to consider QoS parameters as more general non-functional properties.

Based on [13], we define an extensible class QoSProperty which aims to extend nonFunctionalProperties class in WSMO for the P2P-based service selection.

Class nonFunctionalProperties
...other existing properties...
hasQoSProperty **type** QoSProperty

Class QoSProperty **sub-Class** nonFunctionalProperties
hasPropertyName **type** string
hasPropertyValue **type** {int, float, long, others}
hasPreferedValueType **type** {low, high}
hasWeight **type** float

Each QoS Property is generally described by PropertyName and PropertyValue. For the purpose of QoS-based selection, there are two additional attributes defined, namely: "hasPreferedValueType" and "hasWeight". The "hasPreferedValueType" is an object property representing the expected tendency of the value for the ideal attribute. The "hasWeight" is a value denoting the weight of the property, especially when synthetically measuring several different property metrics. In this context we define the weight value within the range [0, 1], while different end users may have different weight values for their service requirements.

For example, Response Time (*RT*) of an atomic service (S_i) on Peer (P_i) can be described in Web service profiles as following:

dc _"http://purl.org/dc/elements/1.1#",
webService _http://example.org/ LoanApprove
nonFunctionalProperties
dc#title **hasValue** "Peer (P_i)"
dc#description **hasValue** "ResponseTime of Atomic Service (S_i) on Peer (P_i)"
……
hasPropertyName **hasValue** _string ("RT")
hasPropertyValue **hasValue** _int ("500")
hasPreferedValueType **hasValue** _string ("low")
hasWeight **hasValue** _float ("0.7")
endNonFunctionalProperties

Likewise, Cost (*CT*), Availability (*AV*) and Reputation (*RP*) can also be modeled in the similar way in WSMO.

3.2 Searching for Composition Solution with Ant Colony Optimisation

Ant Colony Optimisation (ACO) is a well-established optimisation technique based on the principle that real ants are able to find the shortest paths between their nest and a food source [4, 12]. This mechanism works on the basis of pheromones, a kind of biochemical scent, which is left behind by the ants. Other ants are attracted by these pheromones and always walk in the direction with the highest pheromone concentration. ACO is able to provide close-optimal solution to NP-hard combinatorial problems. In this work an artificial ant is an agent which moves from node to node on a composition graph. The following are the three heuristic hints from natural ant behaviours that we have translated to our artificial ant colony: (i) the preference for paths with a high pheromone level, (ii) the higher rate of growth of the amount of pheromone on ideal paths, and (iii) the path-mediated communication among ants.

Table 1. Normalising QoS Attributions [12]:

$RT(l) = \frac{ReponseTime(l)}{\sum_{i=1}^{m} ReponseTime(l)}$; where $ReponseTime(l) = \sum_{i=1}^{n} ReponseTime(ws_i)$;
$CT(l) = \frac{Cost(l)}{\sum_{i=1}^{m} Cost(l)}$; where $Cost(l) = \sum_{i=1}^{n} Cost(ws_i)$;
$AV(l) = \frac{Availability(l)}{\sum_{i=1}^{m} Availability(l)}$; where $Availability(l) = \prod_{i=1}^{n} Availability(ws_i)$;
$RP(l) = \frac{Reputation(l)}{\sum_{i=1}^{m} Reputation(l)}$; where $Reputation(l) = \prod_{i=1}^{n} Reputation(ws_i)$.

For the purpose of selection processes, ResponseTime, Cost, Availability and Reputation are utilised often as QoS factors, and they can be described as a Peer's non-functional properties with WSMO. Assume a composite service "*l*", we define that *RT(l)*, *CT(l)*, *AV(l)* and *RT(l)* are the respective normalised QoS factors (ResponseTime, Cost, Availability and Reputation) in the interval [0,1]. These normalised QoS attributions can be computed by applying the rules described in Table 1, which is based on a simple way of normalisation. The composite service "*l*" contains "*R*" atomic Web services, and at each iteration each ant can find a combination of possible candidate Peers for composite service "*l*").

Therefore, we can define a cost function for evaluating QoS of a service composition *l* as follows:

$$Q(l) = \frac{w_1 \cdot RT(l) + w_2 \cdot CT(l)}{w_3 \cdot AV(l) + w_4 \cdot RP(l)} \qquad (1)$$

Where: w_1, w_2, w_3 and w_4 are real and positive weights which indicate the importance of the QoS factors for service integrator (or user). *Q(l)* represents cost function of composite service *l*, and the objective is to minimise the cost function value of composite service "*l*". The reason why we define the function in this way is that we need to differentiate the preferences of QoS properties, such as Response Time (*RT*) (is preferred as low as possible) and Availability (*AV*) (is preferred as high as possible).

In Figure 3, let 'Z' candidate Peers be in the set: $P=\{P_1, P_2, \ldots, P_Z\}$ and 'R' atomic services be in the set: $S=\{S_1, S_2, \ldots, S_R\}$, and all ants be in the set $I=\{I_1, I_2, \ldots, I_u\}$. The goal is to find suitable Peers for each atomic service. Hence, by applying the ACO based approach, *u* artificial ants are initially placed among Atomic Service$_1$'s nodes (from P_1 to P_Z). At each step they move to newer possible nodes until it reaches the end node. When all the ants have completed a path, the ant that made the path with the lowest cost function value *Q* would modify the links belonging to its path by adding an amount of pheromone trail. In each iteration, each ant generates a feasible composite service by

choosing the nodes according to a probabilistic state transition rule.

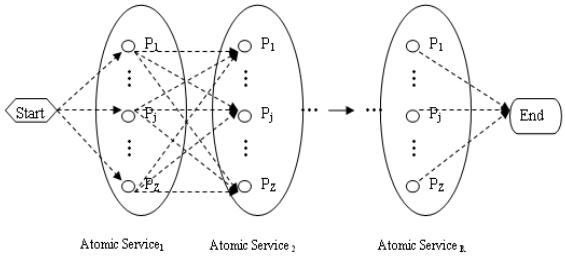

Figure 3: Composition graph for services workflow

For the selection of a node, ant uses heuristic factor as well as the pheromone factor. The heuristic factor denoted by η is a heuristic desirability about an ant moving from a current node to another, and the pheromone factor denoted by τ is an indication of how many ants have visited the link. The probability of selecting a next node (from Atomic Service_i to Atomic Service_j) is given by:

$$P_{ij}^{k} = \begin{cases} \dfrac{\tau_{ij}^{\alpha} \cdot \eta_{ij}^{\beta}}{\sum_{h \in H} \tau_{ih}^{\alpha} \cdot \eta_{ih}^{\beta}} & if \ \rightarrow \ i \in H \\ 0 \rightarrow Otherwise \end{cases} \quad (2)$$

where P^k_{ij} is the probability with which ant k chooses a next node to go. H is the set of nodes that have not been passed by any ant, and h is a node which has not been visited yet by ant k. The parameters α and β control the relative importance of the pheromone versus the heuristic information η_{ij}, which is the heuristic desirability of the ant on node i moves to node j, is given by:

$$\eta_{ij} = \frac{1}{\mathrm{N}j} \quad (3)$$

where N_j is the number of Web services between Atomic Service_j and Atomic Service_n. Actually, for service Peer composition, η can be viewed as a heuristic stimulus which can keep ants moving from source towards the destination covering all atomic services.

The whole path's pheromone update is applied at the end of each iteration by only one ant, which can be the iteration-best or the best-so-far, and the update formula is defined as follows:

$$\tau_{ij} = \begin{cases} (1-\rho) \cdot \tau_{ij} + \rho \cdot \Delta\tau, if \ (i, j) \in L \\ \tau_{ij} \rightarrow Otherwise \end{cases} \quad (4)$$

where $\rho \in (0, 1)$ is a parameter that controls the speed of evaporation of pheromone, and $\Delta\tau = 1/Q(L)$, here L is the current best path with the lowest Q value.

In ACO method, each atomic service on a Peer is regarded as a virtual node of a potential path that might be found by ants. After each iteration, a possible path would be found by ants, and then the combination QoS is calculated and recorded, afterwards the visited nodes' pheromones would be updated. Finally, the nodes with more pheromones would be more possibly lead to an eventual path, i.e. a close-optimal combination would be generated.

3.3 QoS–aware Composition with Integer Programming

As mentioned in related work in Section 2, one of the most widely adopted approaches to solve QoS-aware composition problem is linear integer programming. In this paper, our scope is not to introduce an analytic description of this method, for details readers may refer to Zeng et al. [16]. However, it is necessary to present a high level mathematical formulation on using Integer Programming in service composition, before performing the performance comparison in Section 4.

Given the atomic service set $S=\{S_1, S_2, \ldots, S_R\}$ are required to be invoked in a composite service by a set of Peers $P=\{P_1, P_2, \ldots, P_Z\}$, and suppose that a atomic service ($S_i \in S$) can be executed on some Peers ($P^{S_i} \subseteq P$), and $K^P_{S_i}$ is the total number of possible Peers (P^{S_i}) for atomic service (S_i). Then, we can define our integer programming problem in terms of $\sum_{i=1}^{R} K^P_{S_i}$ integer variables y_{ij} where:

$$\sum_{j=1}^{K^P_{S_i}} y_{ij} = 1; \quad y_{ij} \in \{0,1\}$$

Variable y_{ij} denotes whether an atomic service (S_i) is executable on the Peer candidate ($P_j \in P^{S_i}$). For example, assume that there are 100 potential Peers ($K^P_{S_i} = 100$) that can execute atomic service (S_i). Since only one of them will be selected to execute atomic service (S_i), we have $\sum_{j=1}^{100} y_{ij} = 1$. Therefore, the number of required variable tends to explode with the number of service invoked and with the total number of Peer candidates available. On the contrary, for ACO the path length (i.e. the number of nodes for a path) is bound to the number total number (R) of atomic services. The number of possible Peer candidates only augments the search space.

Furthermore, the integer programming is not suitable for non-linear aggregation functions for the QoS attributes, such as taking into account Availability and Reputation as QoS attributes. Therefore, any user-defined QoS attribute

needs to have a linear (or at least linearised) aggregation function. However, integer programming is often faster than ACO for composition of a limited number of services. Thus, when the workflow size and the number of Peer candidates are limited, and there is no need to use non–linear aggregation functions, integer programming is preferable.

4. Experiments

In this section, we assess performance and feasibility of the proposed ACO method. To verify the proposed approach, experiments were conducted based on different scenarios. Essentially, we conducted few experiments to prove the proposed method: 1. by applying ACO in a general composite workflow, and analyse ACO reliability, in terms of violations; 2. compare the performances between ACO and liner integer programming.

In simulation, we established the environment with Matlab, and all experiments were conducted on a PC with Intel Pentium4 3.0GHz, 2MB; 800 MHz FSB; DDR2 1GB @ 667MHz; and MATLAB 2008a. All QoS data (i.e. ResponseTime, Cost, Availability and Reputation) of service Peers are randomly generated in Matlab, and we simply set all weights same, as we do not need to consider the importance of different QoS attributes in experiments. Through out the duration of the most regular experiments, the parameters of the ACO were set as: α, β are 1 and 5, respectively, ρ (speed of evaporation) is 0.1, and 20 iterations (as few changes occur after 20 runs in most tests) for each trial, and the number of ants are equal to the number of Peer candidates. We also examined how α and β values will change the performance of the ACO algorithm. It's also noteworthy that in deal situations, the bigger the number of ants' amount is larger, the better performance we will get.

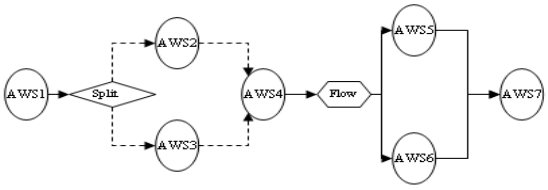

Figure 4: A general composition workflow

As shown in Figure 4, a general workflow, which contains structures of split and flow for service composition, are depicted, with 7 atomic Web services (AWSs) involved. As for as the Peer selection process is concerned, we do not consider differences between simple sequential composition pattern and other mixed patterns, as our emphasis is on selecting an appropriate Peer for each atomic service amongst service Peers. Therefore, the proposed selection method can be extensible for service selection in all sorts of composition patterns, even though different composition patterns may impact the calculation of cost functions.

The experiment assumes that there are 10 service candidate Peers, and utilise the proposed method to select appropriate service Peers which are able to conduct those 7 atomic Web services and provide better overall service quality. The major goal of this method is to identify better Peers with pheromones and probabilities for atomic services. In the selection process, there is little difference between a purely sequential workflow and the mixed one. While using ACO method to search for proper Peer candidates, ants will exploit their own heuristics at join or split nodes.

Table 2: Calculated Q Values of Peers for Atomic Services without ACO Approach

	AWS 1	AWS 2	AWS 3	AWS 4	AWS 5	AWS 6	AWS 7
Peer 1	0.873	0.398	0.780	**0.008**	0.455	0.950	**0.144**
Peer 2	**0.142**	0.413	0.554	Inf	0.869	0.630	0.394
Peer 3	0.682	0.963	Inf	0.677	0.219	0.378	0.259
Peer 4	Inf	**0.075**	0.751	0.217	0.466	0.471	0.338
Peer 5	0.781	0.839	0.532	0.054	0.387	0.887	0.853
Peer 6	0.946	0.248	**0.009**	0.029	Inf	**0.130**	Inf
Peer 7	0.212	0.729	0.683	0.236	0.437	0.442	0.145
Peer 8	Inf	0.695	0.040	0.292	0.372	0.658	0.798
Peer 9	0.990	0.229	0.153	0.229	0.449	Inf	0.162
Peer 10	0.458	0.333	0.830	Inf	**0.193**	0.419	0.465

Table 2 shows each service Peer's Q value that is calculated based on the formulas provided in Table 1, without ACO approach. Each data represents the relationship (Q value) between a candidate Peer and an atomic service rather than a composition of service. The infinite (*Inf*) value of a Peer means that the service Peer can not fulfill the corresponding atomic service, and Q values of selected Peers are calculated based on a set of random data which represents relevant Peers' non-functional properties. When using ACO method, the combination results are exactly the same selections with the best Q value for atomic services in Table 2. It suggests that ACO based approach can also generate the results without completely knowing Q values of each Peer. Actually, to exhaustively calculate all Peers' Q values using this process (e.g. Table 2) is a simple optimisation method to select service Peers for service composition. This approach can easily find the best Peers combination after knowing each Peer's Q values for atomic services. Hence, the major cost of computation by the optimisation method is the calculation based on all possible combinations, and when the number of atomic services and Peers increases, the time cost would greatly increase as well. Given the computation cost, ACO based approach does not need to calculate all service candidates' Q values beforehand, since the calculation is only needed to conduct when a service Peer is chosen with a certain chosen probability. In other words, ACO approach can save more computation time than the optimisation method, particularly for a large number of possible available combinations.

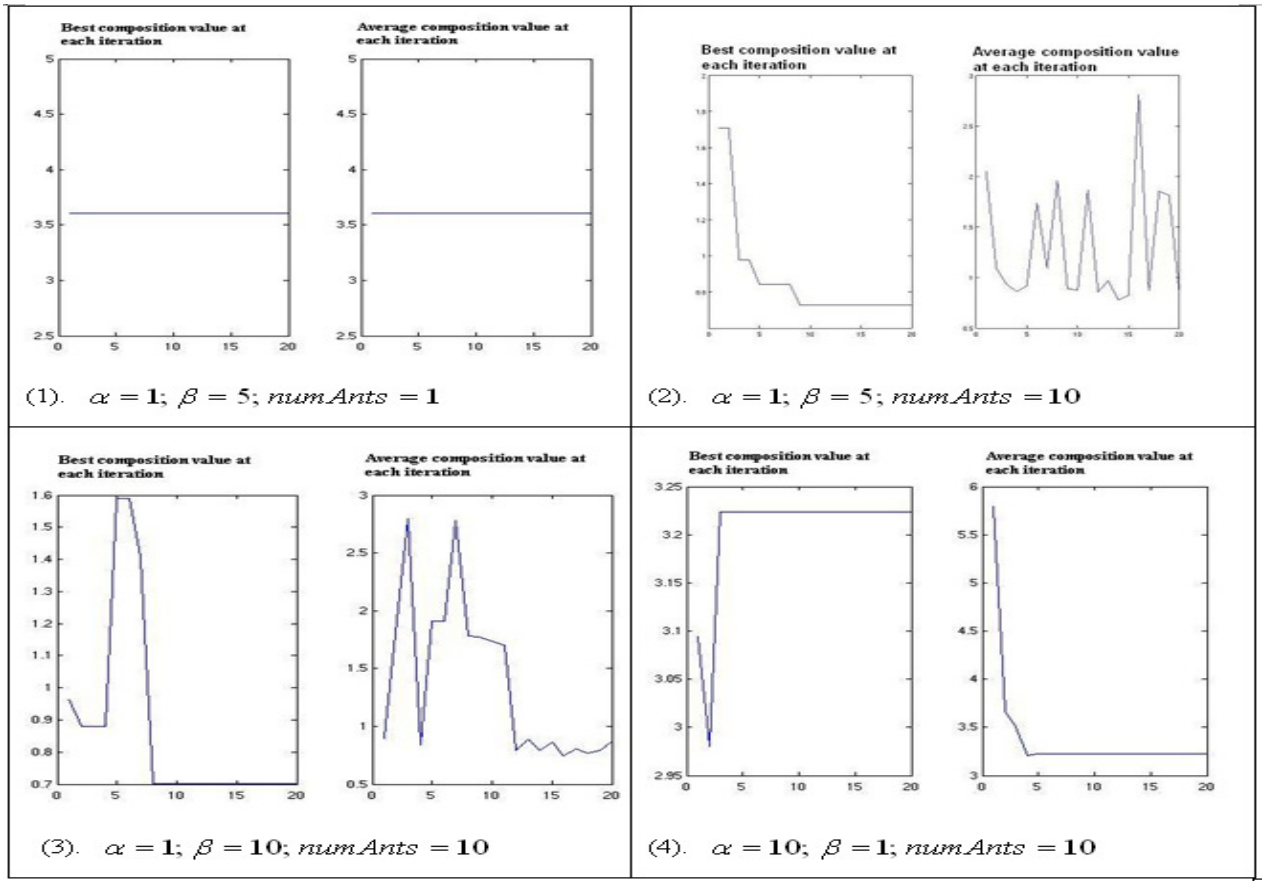

Figure 5: The Q values in ACO iterations with different configurations

Figure 5 shows the nature of ACO regarding best composition value and average value at each iteration with different configurations. This experiment involves comparing four different settings by shifting the key parameters: α, β and the number of ants. In Figure 5, all these composition values are based on calculated *Q* values. At each iteration, the change of best value indicates that ants found a better path of composition than the previous iteration. In Figure 5(1), when there is only one ant, the situation at most time is a constant result for the entire process. As shown in left parts of Figure 5(2), the best composition values were getting better and better from the beginning to the 9th iteration, and afterwards there were no changes happening since ants could not find better solution. In the right part of Figure 5(2), the situation of average composition values of *Q* changed during all the listed 20 iterations. Here, the average value is the mean value of all possible compositions found by ants at each iteration. These changes in average value are caused by the probabilities of ants' choices and the amount of pheromones. However, the trend of the best composition value is led well by heuristic information, as shown in the left part of Figure 5(2). In Figure 5(3), β is changed to 10, which means the heuristic feature is emphasized. However, $\alpha = 1$ means that pheromone factor is relatively weak, the result shows the searching process is not stable in this to a unique direction. There is jump at the 5th iteration and then fell down again and then sticks to the best possible results. In Figure 5(4), α and β are changed into $\alpha = 10$, $\beta = 1$, which means that pheromone feature is stressed in the search process compared to heuristic factor, then ants are attracted to a high pheromone path and the process quickly (at the 4th iteration) focused on a constant result.

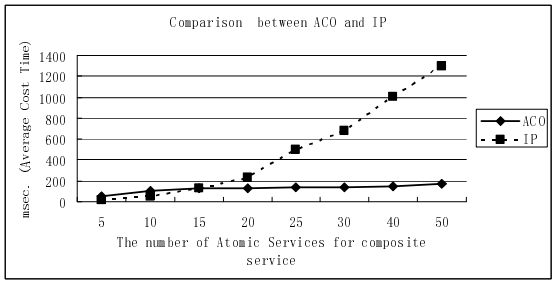

Figure 6: Comparison between ACO and IP

Compared to Integer Programming, as shown in Figure 6 regarding computation time, when the number of atomic services is small, say, less than 15, IP outperforms ACO.

After about 20 atomic services the performances of ACO and IP tend to be the same. Then with more atomic services, while the ACO is able to keep its computation time performance almost constant, this is not the case with IP based optimisation method, where we see an exponential growth due to the corresponding increment of the number of variables needed to represent the problem. In this experiment, we assumed a set of workflows consisting of 5 to 50 atomic services, and the same number of service Peer candidates are involved in the selection for each atomic service.

In general, ACO is viewed as a close-optimal approach for finding suitable combinations. In order to investigate the deviation of ACO from optimal solution, another test is conducted as shown in Figure 7. In this experiment, we ran the selection for 50 atomic services of a composite service by giving three different numbers of iterations: 10-Iteration, 20-Iteration and 50-Iteration. All the comparison values are based on the calculated Q values.

For a composite service, it usually can be viewed as a path, and the way of calculating Q value is based on the formula (1) and Table 1. In Figure 7, it represents deviations between close-optimal paths' Q values and the optimal paths' Q values. The deviations in ACO algorithms increase with the increasing number of atomic services. This is based on the reason that there is a higher probability that ants will visit the next node/service candidate. Hence, sometimes the combination found by ACO is not satisfying if the number of iterations is small or insufficient. However, with the increasing number of iterations, the deviation can be reduced as much as possible. Figure 7 shows that 50-Iteration has relatively less deviation than 10 and 20 Iterations respectively. One final note, ACO can not provide the absolute optimal guarantee for service selection and composition, but it can provide reasonably close-optimal solution that may benefit many practical applications.

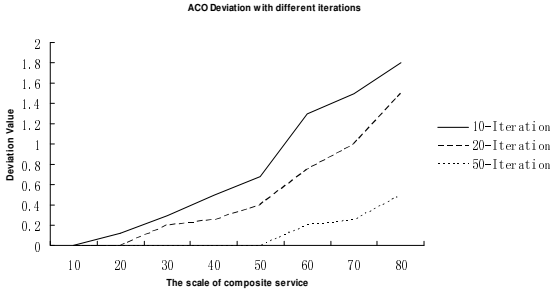

Figure 7: Deviation of ACO

5. Conclusions and Future Work

In this paper, based on the WSMO description for non-functional properties, an ACO-based approach is proposed to tackle the QoS-aware Peers' composition problem, in terms of determining a set of Peers to be bound to atomic services. In the future, we will investigate few probabilistic models, e.g. Partially Observable Markov Decision Process, to cope with more realistic uncertainties within the actual service compositions and executions, and to see how it can be applied to real applications.

6. References

[1] R. Aggarwal, K. Verma, J. Miller and W. Milnor. "Constraint driven Web service composition in METEOR-S". In Proceedings of the 2004 IEEE International Conference on Services Computing, Published by IEEE Computer Society, pp. 23-30, 2004.

[2] L. Cao, M. Li, and J. Cao. "Using genetic algorithm to implement cost-driven Web service selection". Multiagent and Grid Systems, 3(1), pp. 9-17, 2007.

[3] T. Chockalingam and S. Arunkumar. "Genetic algorithm based heuristics for the mapping problem". Computers and Operations Research, 22(1), pp. 55–64, 1995.

[4] M. Dorigo, V. Maniezzo, and A. Colorni. "The Ant System: Optimization by a Colony of Cooperating Agents". IEEE Transactions on Systems, Man and Cybernetics - Part B, 26(1), pp. 1-13, 1996

[5] I. Grossmann. "Review of nonlinear mixed-integer and disjunctive programming techniques". Optimization and Engineering, 3(3), pp. 227-252, September 2002.

[6] JXTA: JXTA v2.0 Protocols Specification. Available at: http://jxta-spec.dev.java.net /JXTAProtocols.pdf

[7] K. Lee, J. Jeon, W. Lee, S. Jeong and S. Park, "QoS for Web services: Requirements and Possible Approaches". W3C Working Group Note 25, 2003. Available at: http://www.w3c.or.kr/kr-office/TR/2003/ws-qos/.

[8] S. Ran, "A model for Web services Discovery with QoS". ACM SIGecom Exchanges, 4(1): 1-10.

[9] D. Roman, U. Keller, H. Lausen, et. al., "Web Service Modelling Ontology". Applied Ontology, 1(1): 77-106, 2005.

[10] J. Shen, J. Yan and Y. Yang: SwinDeW-S: extending P2P workflow systems for adaptive composite Web services. Australian Software Engineering Conference (ASWEC 2006), pp.61-69.

[11] J. Shen, Y. Yang and J. Yan: A P2P based Service Flow System with Advanced Ontology-based Service Profiles. Advanced Engineering Informatics, 21(2): 221-229, 2007.

[12] J. Shen and S. Yuan: QoS-Aware Peer Services Selection Using Ant Colony Optimisation, The 1st Workshop on Service Discovery and Selection in SOA Ecosystems (SDS-SOA 2009) in conjunction with 12th International Conference on Business Information Systems (BIS 2009), Poznan, Poland, published by Springer (LNBIP), April, 2009, pp.362-374.

[13] D.T. Tsesmetzis, I.G. Roussaki, I.V. Papaioannou and M.E. Anagnostou: QoS awareness support in Web-Service semantics, Proceedings of the Advanced International Conference on Telecommunications and International Conference on Internet and Web Applications and Services (AICT/ICIW 2006), p.128.

[14] Y. Vanrompay, P. Rigole, Y. Berbers. "Genetic algorithm-based optimization of service composition and deployment". In Proceedings of the 3rd international workshop on Services integration in pervasive environments, pp. 13-17, 2008.

[15] S. Yuan and J. Shen, "Mining E-Services in P2P-based Workflow Enactments", special issue "Web Mining Applications in E-commerce and E-services" of Online Information Review, Emerald Group Publishing. 32(2), pp: 163-178. 2008.

[16] L. Zeng, B. Benatallah, A. H. H. Ngu, M. Dumas, J. Kalagnanam, and H. Chang. "QoS-aware middleware for Web services composition". IEEE Transactions on Software Engineering, 30(5), pp. 311-327, May 2004