Comprehensive Quality Awareness Automated Semantic Web Service Composition

Chen Wang, Hui Ma, Aaron Chen and Sven Hartmann

School of Engineering and Computer Science,
Victoria University of Wellington, New Zealand
Email: {chen.wang, hui.ma, aaron.chen}@ecs.vuw.ac.nz

Abstract. Semantic web service composition has been a prevailing research area in recent years. There are two major challenges faced by researchers, semantic matchmaking and Quality of Service (QoS) optimisation. Semantic matchmaking aims to discover interoperable web services that can interact with each other by their resources described semantically. QoS optimisation aims to optimise the non-functional requirements of service users, e.g., minimum cost, maximal reliability. Many scholars have looked into QoS optimisation problems in QoS-aware web service composition, applying AI planning and Evolutionary Computation techniques. To meet users' requirements, one often needs to consider both semantic matchmaking quality and QoS simultaneously. Existing works on web service composition focus mainly on one single type of requirements. Therefore, we propose a general comprehensive quality model considering semantic matchmaking quality and QoS simultaneously with an aim of achieving a more desirable trade-off in consideration of both sides. Further more, we develop a PSO-based service composition approach with explicit support for the comprehensive model, where PSO optimise a queue of task related services that is decoded into composition solutions. We compare proposed comprehensive quality model with one promising QoS model, and also compare the effectiveness of PSO-based method with one recent GP-based method using comprehensive quality model.

1 Introduction

Web service composition pertains to a chain of multiple web services to provide a value-added composition service that accommodates customers' arbitrarily complex requirements. This application is developed by integrating interoperable and collaborative functionalities over heterogeneous systems. Due to the increasing number of large-scale enterprise applications, the number of Web services has increased substantially and unprecedentedly. Therefore, *manual and semi-automated web service compositions* are considered to be less efficient while *automated web service composition* demands less human intervention, less time consumption, and high productivity.

Two most notable challenges for web service composition are ensuring interoperability of services and achieving Quality of Service (QoS) optimisation

[6]. *Interoperability* of web services presents challenge in syntactic and semantic dimensions. The syntactic dimension is covered by the XML-based technologies (such as $WSDL$, $SOAP$). The semantic dimension enables a better understanding and collaboration through ontology-based semantics [13]. There are many ontology languages and formats for semantic service descriptions, such as OWL-S, WSML, and SAWSDL [16], which make "machine understanding" possible through identifying and matching semantic similarities in input/output parameters of web services. The second challenge is related to finding optimised solutions in QoS. This problem gives birth to *QoS-aware service composition* that aims to find composition solutions associated with a single optimised QoS value.

Existing works on service composition focus mainly on addressing only one challenge above. Among these works, huge efforts have been devoted to optimise the quality of compositions under a pre-defined abstract workflow. This is generally considered as a *semi-automated Web service composition* approach [1,15]. Meanwhile, many research works consider the NP-hard problem of generating a composition plan automatically in discovering and selecting suitable web services [12]. *Semantic web services composition* is distinguished from the syntactic service composition, the resources of semantic web services are described semantically to enable a better interoperability for chaining web services. In the past few years, substantial works have been done on semantic web service composition [2,4,11]. However, few works have enabled truely automatic semantic web service composition, where both QoS and quality of semantic match making will be optimised simultaneously to achieve a desired balance.

The overall goal of this paper is to *develop a general comprehensive approach to automated QoS-aware semantic web service composition that satisfactorily optimises both QoS and semantic matchmaking quality.* Particularly, this paper extends existing works of QoS-aware service composition by considering jointly optimise QoS and semantic matchmaking quality based on our proposed comprehensive quality model. Particle Swarm Optimisation (PSO) has shown its promise in searching for near-optimised service composition solutions [23]. We will propose a PSO-based service composition approach with explicit support for our proposed quality model. We will achieve three objectives in this work as follows:

1. To propose a general comprehensive quality model that addresses QoS and semantic matchmaking quality simultaneously to achieve a desirable balance on both sides.
2. To propose a PSO-based service composition approach that utilises the proposed comprehensive quality model. To do that we optimise a queue of web services by evaluating the comprehensive quality of its decoded representations.
3. To address a desirable balance that can be achieved using our comprehensive quality model, we conduct experiments to compare our proposed model with one widely used QoS model using PSO-based approach. In addition, to evaluate the effectiveness of our PSO-based approach, we also compare one recent GP approach [10] with our PSO-based approach using our proposed model.

Both comparisons utilise benchmark datasets from Web Services Challenge 2009 (WSC09) [8]

2 Related Work

Substantial works on web service composition focus on either semantic web service composition [4,2,11] or QoS-aware web service composition [7,18,10,23,24,25], However, only a few researchers address both semantic matchmaking quality and QoS requirements for web service composition problems. To the best of our knowledge, [5,9,17] reported some attempts on service composition that considers both aspects.

Semantic web service composition [2,4,11] captures the semantic description of web services' parameters using some kind of logic (e.g., description logic) that ensuring the interoperability of web services, where the number of web services or length of a graph representation for web service composition is minimised to reach the optimised composition solutions. However, this evaluation approach does not guarantee an optimised QoS of composition solutions.

QoS-aware web service composition is studied using traditional approaches or Evolutionary Computation (EC) techniques for finding near-optimised solutions. Qi et al. [18] propose a heuristic service composition method, named LOEM (Local Optimisation and Enumeration Method), where a small number of promising candidates related to each task are considered by local selection, and composition solutions are enumerated to reach the near-to-optimal QoS. However, there exists a scalability problem for the enumeration technique. EC techniques are used to automatically generate solutions with optimal QoS, which is considered to a NP-hard problem. Gupta et al. [7] employ a modified Genetic Algorithm (GA) using a binary string as an individual, which demands to be decoded into composition solutions. Genetic Programming (GP) are used by [25] to find optimal solutions, which is reached by penalising infeasible solutions using a fitness function. A hybrid approach employs both greedy search algorithm and GP introduced in [10] to generate solutions that are functionally correct. In particularly, greedy search techniques have been used to generate directed acyclic graphs (DAGs) as composition solutions that is further transferred to tree structures for initialisation and mutation. To eliminating the transformation process from DAGs, a promising GraphEvol is proposed in [24], where web service composition are in a form of DAGs employing Graph-based evolutionary operators like crossover and mutation. An indirect PSO-based approach was introduced in [23]. An optimised queue is used as an indirect representation that is decoded into a DAG-based solution. These QoS-aware approaches [7,18,10,23,24,25] do not consider semantic matchmaking quality, which could lead to too specific outputs produced by the selected services, and this finding is supported by our first comparison experiment.

Only a few works [5,9,17] consider both semantic matchmaking quality and QoS simultaneously. Lecue et al. [9] propose a semi-automated web service composition using GA to encode a given abstract service workflow, where the seman-

tic matchmaking quality is measured through evaluating the quality of semantic links that require a complete and formal definition of ontology in Description Logic. This evaluation model takes huge cost and time for the domain experts to establish required ontology. Another GA-based approach [5] utilise process description language to encode pre-stored cases-based workflows, where workable services are composed to complete this workflow. The work [17] is an automated immune-inspired web service composition approach, where an indirect representation that a binary alphabet is used to encode a planning graph. However, this composition method is only evaluated on a set of scenarios.

In summary, despite a large number of approaches for semantic web service composition and QoS-aware service composition approaches, there is a lack of a fully automated semantic web service composition approach to optimise semantic matchmaking quality and QoS simultaneously, and this approach must be tested utilising benchmark dataset. Therefore, we first propose a general comprehensive quality model that aims to achieve a better semantic matchmaking quality with a slight trade-off in QoS, and then we propose an automated PSO-based approach with explicit support for the comprehensive model to find optimal service compositions.

3 Motivation and Problem Description

3.1 Motivation

Our goal in this paper is to develop a PSO-based approach that can automatically produce composition services in the form of graph-based representation (e.g. DAG). The nodes in the DAG represent component services. On the other hand, the edge that connects any two component services in the DAG, e.g. $S_1 -> S_2$, indicates that the output information from service S1 will subsequently serve as the input information for service S2. Apparently, such output and input information must semantically match to ensure the correct execution of the DAG. Moreover, the overall service inputs and outputs associated with the DAG must satisfy the composition requirements provided by users.

In practice, many different DAGs can meet the given requirements but differ significantly in terms of QoS and semantic matchmaking quality. For example, in the classical travel planning context, during the process of building a DAG, a component service needs to be employed to obtain a travel map. Suppose that two optional services can be considered for this purpose. One service S1 can provide a street map at a price of 6.72. The other service S2 can provide a tourist map at a price of 16.87. Because in our DAG a tourist map is more desirable than a street map, service S2 clearly enjoys better semantic matchmaking quality but will have negative impact on the QoS of the DAG (i.e. the price is much higher). It can be easily imagined that similar problems can frequently occur while building the DAG for the travel planning problem or other web service composition problems. Due to this reason, aimed at achieving a good balance in between QoS and semantic matchmaking quality, our PSO-based service com-

position system will be further guided by a comprehensive quality model to be proposed in 4.

3.2 Problem Formulation

We herein give formal definitions of the concepts regarding the semantic web service composition problem utilised in our approach, and these concepts are captured in several models as follows.

Semantic Web Service Model. The semantic web service model represents web services with functional descriptions, and how interoperability of web services enables matchmaking for automatically generating composition solutions.

Definition 1. A request from users is defined as a composition task in a tuple: $Comp.T = \{I_T, O_T\}$ by giving a set of task inputs I_T and expecting a set of desired task outputs O_T, where I_T and O_T are associated with related concepts I_T^C and O_T^C respectively in an ontology $O = \{C \mid I_T^C, O_T^C \subseteq C\}$.

Definition 2. A semantic web service (hereafter "service") is defined as a tuple $S = \{I_S, O_S, QoS_S\} \in \mathcal{SR}$, where I_S is a set of service inputs that must be satisfied to invoke S, and O_S is a set of service outputs returned by the execution of S. I_S and O_S are concept-related instances that are linked to their corresponding concepts I_S^C and O_S^C, where $I_S^C, O_S^C \subseteq C$. $QoS_S = \{t_s, c_s, r_s, a_s, ...\}$ is a set of non-functional attributes of S, it refers to response time t_s, cost c_s, reliability r_s, and availability a_s [26]. $\mathcal{SR}$ is a service repository consisting of a set of S.

Semantic Matchmaking Model. The mechanism to compose web services relies on the semantic descriptions of inputs and outputs, which enable inputs of a service to be matched by outputs of another. To measure the quality of these matches, we utilise different matchmaking types (*exact, plugin subsumes and fail* [14]).

Definition 3. Given $a, b \in C$, $type(a, b)$ returns the matchmaking types of two concepts, and this matchmaking type is determined by the logic relationship described in the O. If both a and b are equivalent, *exact* type ($a \equiv b$) is returned. If the concept of a is a sub-concept of the concept of b type ($a \sqsubseteq b$) is returned. If the concept of a is a super-concept of the concept of b, *subsumes* type ($a \sqsupseteq b$) is returned. If none of previous matchmaking types are returned, $fail$ type ($a \perp b$) is returned.

We use $type(a, b)$ to define a full matchmaking considered in the scenario of connecting two services through concept-related output and concept-related input.

Definition 4. Given $(a \in O_{S_m}^C) \vee (b \in I_{S_n}^C) \wedge (m \neq n)$, a $fullmatch(a \Rightarrow b)$ holds if $type(a, b)$ returns *exact* or *plugin*. The operator "$\Rightarrow$" means a is passed to b, and b is fully satisfied by a. The full match is a guarantee of completely valid "$\Rightarrow$" operation, whereas, partial matches are not considered in this paper.

QoS calculation Model in a DAG Four most often considered QoS parameters [26] are response time, cost, reliability and availability described as

follows: *Response time* (t_s) measures the expected delay in seconds between the moment when a request is sent and the moment when the results are received.*Cost* (c_s) is the amount of money that a service requester has to pay for executing the web service. *Reliability* (r_s) is the probability that a request is correctly responded within the maximum expected time frame, which is expressed as a number between 1 and 0. *Availability* (a_s) is the probability that a web service is accessible, which is expressed in the same way as r_s. The aggregation value of QoS in a DAG respect to the DAG's structures that determine how services associated with each other in a service composition. The QoS calculation models are described based on QoS aggregation method [24] as follows:

A, R, C, T is aggregated a_s, r_s, c_s and t_n respectively with a given WG, where $\{S_1, S_2, ...S_n\} \subset V$, such that: $A = \prod_{k=1}^{n} a_k$; $R = \prod_{k=1}^{n} r_k$; $C = \sum_{k=1}^{n} c_k$; T is the time of most time-consumption path of WG, so $T = MAX\{timePath()_n | n \in \{1, \ldots, m\}\}$ where m is the number of WG path, and $timePath() = \sum_{k=1}^{j} t_k$, where j is the number of services involved this path.

4 Comprehensive Quality Model

In this section, we propose a general comprehensive quality model in considering semantic matchmaking quality and QoS simultaneously. This comprehensive model is fully supported by the representations of service composition discussed in this paper, as a weighted DAG model formulated in this section.

4.1 Comprehensive Quality Model

Semantic matchmaking quality model. Due to that the discretisational characteristics of different match types and values assigned to matching types driven by the cost of data manipulation [9], match types are considered to be one factor for the quality of matchmaking. For example, Exact matching type demands less time for computation compared to that of Plugin match. Another factor is concept similarity of two matched concepts. For example, two plugin types, $publication \sqsubseteq printedmaterial$ and $romanticnovel \sqsubseteq printedmaterial$, are associated with different similarities.

Definition 5. Given $fullmatch(a \Rightarrow b)$, $sim(a, b)$ is a function that given a output-related concept a of a service and a input-related concept b of another service, it returns a concept similarity value between a and b.

The $sim(a, b)$ is calculated based on the edge counting method defined in the formula (1) from [19], where N_1, N_2 and N measure the distances from an output-related concept a, an input-related concept b, a closest common ancestor of these two matched a and b to the top concept of an ontology O respectively. λ is set to 0 as we do not measure the similarities of neighbourhood concepts, which is not the matching type considered in this paper.

$$sim(a,b)=\frac{2N \cdot e^{-\lambda L/D}}{N_1+N_2} \tag{1}$$

Definition 6. Given $fullmatch(a \Rightarrow b)$, $sm(a,b)$ is a function for measuring semantic matchmaking quality between two concepts, which returns a pair of values consisting of $type(a,b)$ and $q_{sim}(a,b)$ defined in Formula (2).

$$sm(a,b) \doteq (type(a,b),\ sim(a,b)) \tag{2}$$

Weighted DAG composition model. web service composition is considered to be a data workflow, where services are chained together by their inputs and outputs using the two models defined before. All these concepts are captured and tackled in a weighted DAG as $WG = (V, E)$, where:

$V = \{Start, S1, S2...Sn, End\}$ is a set of services, where $Start$ and End are two special services defined as $Start = \{\phi, I_T, \phi\}$ and $End = \{O_T, \phi, \phi\}$.

$E = \{e_1, e_2, ...e_m\}$ is a set of edges, where each edge has a set of incoming outputs $O_{S_{src}}$ from source service S_{src} and a set of outgoing inputs $I_{S_{tar}}$ from target service S_{tar}.

$e = \{(O_{S_{src}} \cup I_{S_{src}}) \mid \forall o \in O_{S_{src}}, \exists i \in O_{S_{src}} \wedge fullmatch(o \Rightarrow i)\}$

Comprehensive quality model in a weighted DAG.

Since each e defined in E could have more than one $fullmatch(a \Rightarrow b)$ form a edge. Therefore, semantic matchmaking quality of one edge, denoted as sm_e is defined in formula (3), where $type_e$ and sim_e are the average values of all involved $type(a,b)$ and $q_{sim}(a,b)$ respectively.

$$sm_e \doteq (type_e,\ sim_e) \tag{3}$$

The semantic matchmaking quality SM is further aggregated with considering all $\{e_1, e_2, ...e_m\}$ in E, calculated by following the formula (4)(5)(6). Consequently, in Formula 7, the comprehensive quality CQ consists of the aggregated semantic matchmaking quality SM and the QoS calculation in DAG discussed before.

$$SM \doteq (MT, SIM) \tag{4}$$

$$MT=\prod_{n=1}^{m} type_{e_n} \tag{5}$$

$$SIM=(\sum_{n=1}^{m} sim_{e_n})/m \tag{6}$$

$$CQ \doteq (MT,\ SIM,\ A,\ R,\ C,\ T) \tag{7}$$

4.2 Objective Function

In real life, given a unique and optimised solution is always easier for customers to pick up directly when many quality criteria involved into decision making, rather than provided a set of solutions. It is very practical to define a single fitness as a weighted sum of all the quality related components in Formula (8). Note that weights can be adjusted according to users' preferences. The function value of 1 means the best comprehensive quality and 0 means the worst. For this purpose, MT, S, A, R, T, and C must be normalised so that the function value falls within the range from 0 to 1 using Formula (9) and (10), where the maximum and minimum value of A, R, T, and C are calculated by all web services candidates to the composition task, which are generated by a greedy search discussed in 5.1. MT and S are using a bound from 0 to 1. Therefore, the composition task is try to find maximised value of objective function associated to the solutions.

$$Fitness = w_1\hat{MT} + w_2\hat{S} + w_3\hat{A} + w_4\hat{R} + w_5(1-\hat{T}) + w_6(1-\hat{C}) \tag{8}$$

where $\sum_{i=1}^{6} w_i = 1$

$$\hat{Q_k} = \begin{cases} \frac{Q_k - Q_{k,min}}{Q_{k,max} - Q_{k,min}} & \text{if } Q_{k,max} - Q_{k,min} \neq 0. \\ 1 & \text{otherwise.} \end{cases} \tag{9}$$

where $k = 1$, 2, 3, and 4, where Q_1 is MT, Q_2 is S, Q_3 is A, and Q_4 is R.

$$\hat{Q_j} = \begin{cases} \frac{Q_{j,max} - Q_j}{Q_{j,max} - Q_{j,min}} & \text{if } Q_{j,max} - Q_{j,min} \neq 0. \\ 1 & \text{otherwise.} \end{cases} \tag{10}$$

where $j = 1$, and 2, where Q_1 is T and Q_2 is C.

Our comprehensive quality evaluation model is considered to be more general. Two semantic matchmaking types (Exact and Plugin) are considered for discovering desired web service in recent QoS-aware web service composition [10,23,24], but there are no measurements of matchmaking types and concept similarities in their approaches. Consequently, weights for MT and S are considered to be 0 in our proposed comprehensive evaluation model, it turns to be a widely used QoS evaluation model in [10,23,24].

5 PSO-based Approach to Comprehensive Quality-Aware Automated Semantic Web Service Composition

5.1 An Overview to PSO-based Method

PSO has shown promise in solving combinatorial optimisation problems, and considered an easier way to maintain the correctness of solutions compared to GP-based approaches that often require repairing the solutions [23]. Therefore, we employ a PSO-based approach to comprehensive quality-aware automated

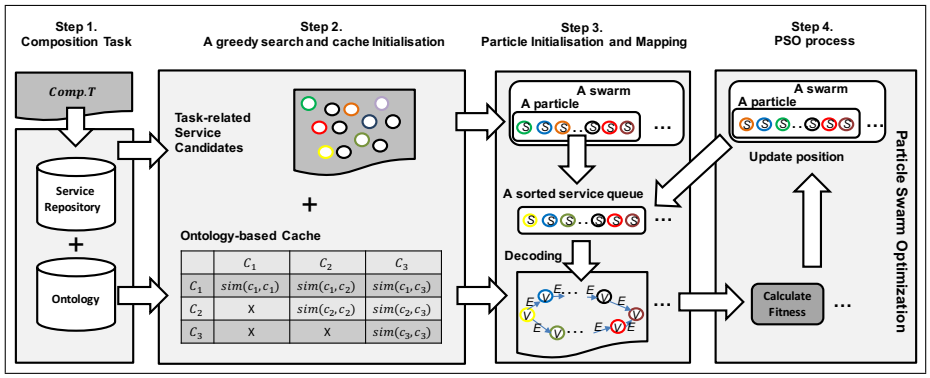


Fig. 1: An overview of POS-based approach to comprehensive quality-aware automated semantic web service composition.

semantic web service composition. Fig. 1 shows an overview of our approach consisting of four steps:

Step 1: The composition process is triggered by a composition task, which is clearly defined in 3.2.

Step 2: This composition task is used to discover all task-related service candidates using a greedy search adopted from [10]. This greedy search algorithm keeps adding outputs of the invoked services as available outputs (initialised with I_T) , and these available outputs are used to discover task-related services from a service repository and updated with the outputs of these discovered services. This operation is repeated until no service is satisfied by the available outputs. This greedy search contributes to a shrunken service repository. During the greedy search, an ontology-based cache (*cache*) is initialised that stores the concept similarities of potentially matched inputs and outputs of task-related candidates. This *cache* is also used to check whether services can be invoked by satisfying their input-related concepts with provided output-related concepts.

Step 3 and Step 4: These two steps follow the standard PSO steps [20] except for some differences in particles mapping and decoding processes. In particular, these two differences are related to sorting a created service queue using serivce-to-index mapping for a particle' position vectors and evaluating the fitness of a particle after decoding this serivce queue into a weighted DAG respectively. Those differences are further addressed in Algorithms 1 and 2 in 5.2.

5.2 PSO-based approach algorithm

The overall algorithm investigated here is made up of a PSO-based web service composition algorithm 1 and a decoding algorithm 2. In Algorithm 1, the steps 4, 5, 6 and 7 are different from those of stardard PSO: In step 4, the size of task-related service candidates generated by a greedy search determines the size of each particle's position, and each candidate in a created service candidates

queue is mapped to an index of a particles position vectors, where each vector has a weight value between 0.0 and 1.0. In step 5, all the task-related service candidates in the queue are sorted according to their corresponding weight values in descending order. In step 6, this sorted queue is used as one of the inputs of the forward decoding algorithm 2 to create a weighted DAG. In step 7, this fitness value of this weighted DAG is the fitness value of the particle calculated by the comprehensive model discussed in 4.

ALGORITHM 1. Steps of PSO-based service composition technique [23].

```
1:  Randomly initialise each particle in the swarm;
2:  while max. iterations not met do
3:      foreach particles in the swarm do
4:          Create a service candidates queue and map service candidates to a
              particle's position vectors;
5:          Sort the service queue by position vectors' weights;
6:          Create a weighted DAG from the service queue ( Algorithm2) ;
7:          Calculate the Weighted DAG fitness value;
8:          if fitness value better than pBest then
9:              Assign current fitness as new pBest;
10:         else
11:             Keep previous pBest;
12:     Assign best particle's pBest value to gBest, if better than gBest;
13:     Calculate the velocity of each particle;
14:     Update the position of each particle;
```

Algorithm 2 is a forward graph buidling algorithm based on [3]. This algorithm take one input, a sorted service queue from step 5 of Algorithm 1. If service queues are sorted resulting in different service order, it is possible to create different corresponding weighted DAGs as composition solutions. In addition. I_T, O_T and *cache* are also taken as the inputs. Firstly, *Start* and *End* are added to V of WG as an initialisation, and $OutputSet$ is also created with I_T. If all the inputs I_S of the first popped S from *queue* can be satisfied by provided outputs from $OutputSet$. This S is added to V and its outputs are added to $OutputSet$, and S is removed from *queue*. Meanwhile, e is created with its sm_e calculated based on the aggregation value of all semantic matchmaking qualities from each satisfied input of S. These steps are repeated until all O_T can be satisfied by *Outputset* or the service queue is *null*. Consequently, this forward graph building technique could lead to more services and edges connected to the WG, which should be removed before WG is returned.

ALGORITHM 2. Create a weighted DAG from a sorted queue.

```
1:  Procedure createWeightedDAG()
        Input  : I_T, O_T, queue, Cache
        Output: WG
2:      WG = (V, E);
3:      V ← {Start, End } ;
4:      OutputSet ← {I_T};
5:      while all O_T do not satisfied by OutputSet do
6:          foreach S in queue do
7:              if all I_S satisfied by OutputSet then
8:                  foreach I_S do
9:                      sm(a ∈ Outputset, b ∈ I_S) ← query Cache;
10:                 e ← calculate aggregated sm_e;
11:                 E add e;
12:                 V add S;
13:                 OutputSet add {O_S};
14:                 queue.remove S;
15:     remove danglingnodes;
16:     remove danglingedges;
17:     return WG;
```

6 Experiment Study

In this section, a quantitative evaluation approach is adopted in our experiment design. Two objectives of this evaluation are to: (1) measure the effectiveness of our PSO-based approach. To do this, we compare our PSO-based method with one existing GP-based approach [10]. (2) measure the effectiveness of our proposed comprehensive quality model. To do this, we compare a widely used QoS model in QoS-aware web service composition [10,23,24] with our comprehensive quality model using our proposed PSO-based approach.

An augmented version of Web service challenge 2009 (WSC09) used in [10,22] including QoS attributes is used for our experiment. WSC09 is a benchmark dataset consisting of five composition tasks that associated with an increasing number of services, and concepts in ontologies.

The parameters are chosen based on the settings from [20] for our PSO-based approach, In particular, PSO population size is 30 with 100 generations. We run 30 times independently for each dataset. We configure weight of fitness function to properly balance functional side and nonfunctional side. Therefore, w_1 and w_2 are set equally to 0.25, and w_3, w_4, w_5, w_6 are all set to 0.125. The returned value of $type(a, b)$ is set to 1 ($Exact$) and 0.75 ($Plugin$) according to [9]. In general, weight settings and parameter match type quality are decided by users' preferences.

6.1 Comparison Test for GP-based approach and PSO-based approach

To evaluate the effectiveness of our proposed PSO-based approach, we compare one recent GP-based approach [10] with our PSO-based method. The semantic matchmaking quality of this GP-based approach is easy to evaluated considering measuring links between parent nodes and children nodes. Therefore, we evaluate both semantic matchmaking quality and QoS simultaneously for that GP-based approach using the proposed comprehensive quality model. To make a fair comparison, we consider the same number of evaluations (3000 times) used in our PSO-based approach. We set the parameters' settings of that GP-based approach as 30 individuals and 100 generations, it is considered to be proper settings refering to [21].

The first column of Table 1 shows five tasks from WSC09 Dataset. The second and third column of Table 1 show the orginal service repository size before the greedy search and shrunk service repository size after the greedy search respectively regarding the five tasks. This greedy search helps reducing the repository size by considering only task-related service candidates, which also contributes to a significant reduced researching space. The fourth and fifth column of Table 1 show the mean fitness values of 30 independent runs accomplished by two methods. We employ independent-samples T tests to test the significant differences in mean fitness value. The results show that the PSO-based approach outperformes the existing GP-based approach in most cases except task 3 (all the p-values are consistently smaller than 0.01). In task 5, PSO-based approach performs significantly better than GP-based approach in finding optimal solutions. It may refer to that GP-based approach is stuck in the local optimal due to a very large searching space of our composition problem, but PSO-based approach requres a decoding process, and small changes in a sorted queue might lead to a varying decoded solutions. Therefore, PSO-based approach is considered to be hard to be stuck in the local optimal for our composition problem.

Table 1: Mean fitness results for comparing GP-based approach

WSC09	Original SR	Shrunken SR	PSO-based approach	GP-based approach
Task 1	572	80	0.5592 ± 0.0128 ↑	0.5207 ± 0.0208
Task 2	4129	140	0.4701 ± 0.0011 ↑	0.4597 ± 0.0029
Task 3	8138	153	0.5504 ± 0.0128	0.5679 ± 0.0234 ↑
Task 4	8301	330	0.4690 ± 0.0017 ↑	0.4317 ± 0.0097
Task 5	15211	237	0.4694 ± 0.0008 ↑	0.2452 ± 0.0369

6.2 Comparison Test for Comprehensive Quality Evaluation Model and QoS Evaluation Model

A recent widely used QoS Evaluation Model, $Fitness = w_1\hat{A}+w_2\hat{R}+w_3(1-\hat{T})+w_4(1-\hat{C})$, where $\sum_{i=1}^{4} w_i = 1$, is used for QoS-aware web service composition [10,23,24]. We employ this QoS evluation model and our proposed comprehensive quality evluation model using our proposed PSO-based approach for searching for optimal solutions. To analyse the differences in optimal solutions found by these two evaluation model, we recorded and compared the different mean values of SM (consisiting of MT and S), QoS(consisting of A, R, T and C) at the 100th generation of 30 runs. To make a sense of the comparison, all these recorded values are normalised from 0 to 1, and compared using statistical analysis in in Table 2.

We observe an interesting pattern from Table 2. The mean values of QoS using QoS evaluation model are significantly higher than those using comprehensive quality evaluation model for Task 2, 3, 4 and 5, but the mean value of SM using comprehensive quality evaluation model are consistently significantly higher than those using QoS evaluation model for all tasks with a slight trade-off in QoS.

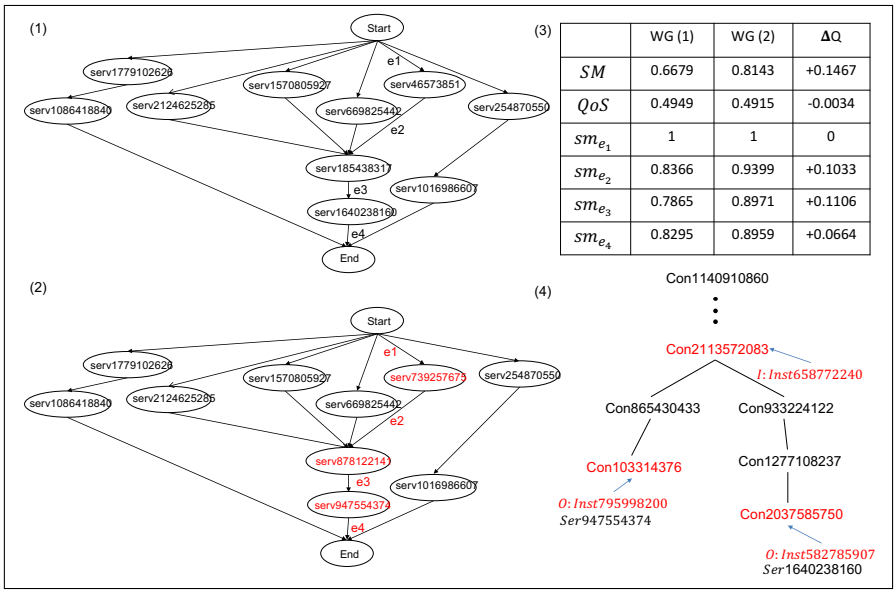
Table 2: Mean values of SM and QoS for QoS evaluation model and comprehensive quality evaluation model using PSO-based approach

WSC09		QoS Evaluation Model	Comprehensive Quality Model
Task1	SM	0.5373 ± 0.0267	0.5580 ± 0.0094 ↑
	QoS	0.5574 ± 0.0156	0.5604 ± 0.0164
Task2	SM	0.4549 ± 0.0033	0.4630 ± 0.0042 ↑
	QoS	0.4800 ± 0.0012 ↑	0.4772 ± 0.0025
Task3	SM	0.5538 ± 0.0082	0.6093 ± 0.0054 ↑
	QoS	0.4940 ± 0.0013 ↑	0.4913 ± 0.0009
Task4	SM	0.4398 ± 0.0037	0.4604 ± 0.0000 ↑
	QoS	0.4845 ± 0.0010 ↑	0.4734 ± 0.0044
Task5	SM	0.4580 ± 0.0065	0.4639 ± 0.0013 ↑
	QoS	0.4764 ± 0.0005 ↑	0.4750 ± 0.0007

6.3 Further Discussion

To better understand a better semantic matchmaking quality with a slight trade-off in the opitmal solutions using comprehensive quality evaluation model, we demonstrate the optimal solutions using task 3 as an example. Fig. 2 (1) and (2) show two weighted DAGs as best solutions found by employing QoS evaluation model and comprehensive quality evaluation model respectively. Two approaches generate exactly the same service workflow structure, but some service vertices and edges denoted in red are different. To better understand these

differences, we list the overall semantic matchmaking quality SM, overall QoS and semantic matchmaking quality associated to each edge (sm_{e_1} to sm_{e_4}) for the two weighted DAGs in Fig. 2 (3), where ΔQ reveals the gain (positive ΔQ) or a loss (negative ΔQ) for the listed qualities using our comprehensive quality evalutaion model. Therefore, an overall gain 0.1433 is calculated from a sum of a SM gain (0.1467) and QoS loss (-0.0034). Consequently, our comprehensive evaluation model acheive a desirable trade-off in considering both SM and QoS. To explain the value of SM gain, we pick up e_4 that is associated with a smallest ΔQ. The e_4 of $WG(1)$ using QoS evaluation model and $WG(2)$ using comprehensive quality model has two different source service vertices $Ser1640238160$ and $Ser947554374$ respectively, and the same *end* vertices. $Ser1640238160$ and $Ser947554374$ are services with concept-related output parameters $Inst795998200$ and $Inst582785907$ corresponds to two concepts $Con103314376$ and $Con2037585750$ respectively, which are marked on the related taxonomy in Fig. 2 (4). In addtion, $Inst658772240$ is a required parameter of the *end* vertice as one of composition task outputs, which is related to concept $Con2113572083$. Obviouly, $Inst795998200$ is closer to user's required output $Inst658772240$ compared to $Inst582785907$.



	WG (1)	WG (2)	ΔQ
SM	0.6679	0.8143	+0.1467
QoS	0.4949	0.4915	-0.0034
sm_{e_1}	1	1	0
sm_{e_2}	0.8366	0.9399	+0.1033
sm_{e_3}	0.7865	0.8971	+0.1106
sm_{e_4}	0.8295	0.8959	+0.0664

Fig. 2: An example of comparision to optimal solutions using Task 3 for QoS evaluation model and comprehensive evluation model.

7 Conclusion

This work introduces a general comprehensive evaluation model for considering semantic matchmaking quality and QoS simultaneously. We proposed a PSO-based service composition approach utilising our proposed quality mode that can achieve a desired trade-off of both quality aspects. In addtion, we compare one recent GP-approach with our PSO-based method to show our performance that results in finding more optimised solutions. Future works can investigate multi-objective EC techniques to produce a set of composition solutions for the situations when the quality preference is not known.

References

1. Bahadori, S., Kafi, S., Far, K., Khayyambashi, M.: Optimal web service composition using hybrid ga-tabu search. Journal of Theoretical and Applied Information Technology 9(1), 10–15 (2009)
2. Bansal, S., Bansal, A., Gupta, G., Blake, M.B.: Generalized semantic web service composition. Service Oriented Computing and Applications 10(2), 111–133 (2016)
3. Blum, A.L., Furst, M.L.: Fast planning through planning graph analysis. Artificial intelligence 90(1), 281–300 (1997)
4. Boustil, A., Maamri, R., Sahnoun, Z.: A semantic selection approach for composite web services using owl-dl and rules. Service Oriented Computing and Applications 8(3), 221–238 (2014)
5. FanJiang, Y.Y., Syu, Y.: Semantic-based automatic service composition with functional and non-functional requirements in design time: A genetic algorithm approach. Information and Software Technology 56(3), 352–373 (2014)
6. Fensel, D., Facca, F.M., Simperl, E., Toma, I.: Semantic web services. Springer Science & Business Media (2011)
7. Gupta, I.K., Kumar, J., Rai, P.: Optimization to quality-of-service-driven web service composition using modified genetic algorithm. In: Computer, Communication and Control (IC4), 2015 International Conference on. pp. 1–6. IEEE (2015)
8. Kona, S., Bansal, A., Blake, M.B., Bleul, S., Weise, T.: Wsc-2009: a quality of service-oriented web services challenge. In: 2009 IEEE Conference on Commerce and Enterprise Computing. pp. 487–490. IEEE (2009)
9. Lécué, F.: Optimizing qos-aware semantic web service composition. In: International Semantic Web Conference. pp. 375–391. Springer (2009)
10. Ma, H., Wang, A., Zhang, M.: A hybrid approach using genetic programming and greedy search for qos-aware web service composition. In: Transactions on Large-Scale Data-and Knowledge-Centered Systems XVIII, pp. 180–205. Springer (2015)
11. Mier, P.R., Pedrinaci, C., Lama, M., Mucientes, M.: An integrated semantic web service discovery and composition framework (2015)
12. Moghaddam, M., Davis, J.G.: Service selection in web service composition: A comparative review of existing approaches. In: Web Services Foundations, pp. 321–346. Springer (2014)
13. O'Leary, D.: Review: Ontologies: A silver bullet for knowledge management and electronic commerce. The Computer Journal 48(4), 498–498 (2005)
14. Paolucci, M., Kawamura, T., Payne, T.R., Sycara, K.: Semantic matching of web services capabilities. In: International Semantic Web Conference. pp. 333–347. Springer (2002)

15. Parejo, J.A., Fernandez, P., Cortés, A.R.: Qos-aware services composition using tabu search and hybrid genetic algorithms. Actas de los Talleres de las Jornadas de Ingeniería del Software y Bases de Datos 2(1), 55–66 (2008)
16. Petrie, C.J.: Web Service Composition. Springer (2016)
17. Pop, C.B., Chifu, V.R., Salomie, I., Dinsoreanu, M.: Immune-inspired method for selecting the optimal solution in web service composition. In: International Workshop on Resource Discovery. pp. 1–17. Springer (2009)
18. Qi, L., Tang, Y., Dou, W., Chen, J.: Combining local optimization and enumeration for qos-aware web service composition. In: Web Services (ICWS), 2010 IEEE International Conference on. pp. 34–41. IEEE (2010)
19. Shet, K., Acharya, U.D., et al.: A new similarity measure for taxonomy based on edge counting. arXiv preprint arXiv:1211.4709 (2012)
20. Shi, Y., et al.: Particle swarm optimization: developments, applications and resources. In: evolutionary computation, 2001. Proceedings of the 2001 Congress on. vol. 1, pp. 81–86. IEEE (2001)
21. da Silva, A.S., Ma, H., Zhang, M.: A gp approach to qos-aware web service composition including conditional constraints. In: Evolutionary Computation (CEC), 2015 IEEE Congress on. pp. 2113–2120. IEEE (2015)
22. da Silva, A.S., Ma, H., Zhang, M.: Genetic programming for qos-aware web service composition and selection. Soft Computing pp. 1–17 (2016)
23. da Silva, A.S., Mei, Y., Ma, H., Zhang, M.: Particle swarm optimisation with sequence-like indirect representation for web service composition. In: European Conference on Evolutionary Computation in Combinatorial Optimization. pp. 202–218. Springer (2016)
24. da Silva, A., Ma, H., Zhang, M.: Graphevol: A graph evolution technique for web service composition. In: Chen, Q., Hameurlain, A., Toumani, F., Wagner, R., Decker, H. (eds.) Database and Expert Systems Applications, Lecture Notes in Computer Science, vol. 9262, pp. 134–142. Springer International Publishing (2015)
25. Yu, Y., Ma, H., Zhang, M.: An adaptive genetic programming approach to qos-aware web services composition. In: 2013 IEEE Congress on Evolutionary Computation. pp. 1740–1747. IEEE (2013)
26. Zeng, L., Benatallah, B., Dumas, M., Kalagnanam, J., Sheng, Q.Z.: Quality driven web services composition. In: Proceedings of the 12th international conference on World Wide Web. pp. 411–421. ACM (2003)