Comprehensive Quality Awareness Automated Semantic Web Service Composition

Chen Wang, Hui Ma, and Aaron Chen

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

[5]. *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 [2,3,11]. 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 [22]. 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 its decoded representations. These representations are the solutions for semantic web service composition.
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 [3,2,11] or QoS-aware web service composition [7,18,10,22,23,24], However, only a few researchers address both semantic matchmaking quality and QoS requirements for web service composition problems. To the best of our knowledge, [4,9,17] reported some attempts on service composition that considers both aspects.

Semantic web service composition [3,2,11] 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 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 [24] 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 is 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 [23], 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 [22]. An optimised queue is used as an indirect representation that is decoded into a DAG-based solution. These QoS-aware approaches [7,18,10,22,23,24] 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 [4,9,17] 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 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 [4] 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. 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

The aim of web service composition considered in this paper is to automatically compose optimal solutions in considering both QoS and semantic matchmaking quality based on a user's request. The request is to provide inputs and obtain outputs. The resulting composition of this request is a composition solution with an optimal comprehensive quality in terms of semantic matchmaking quality and QoS. A motivating example of this problem is shown in Fig 1. The figure shows a DAG that represents a solution with four web services involved for a request R as a composition task, where inputs of the task are {TravelDepartureDate, HomeCity, ConferenceCity, TravelReturnDate } and outputs are {BusTicket, FlightTicket, TouristMap, HotelReservation }. This is a classic example of travel planning problem for people who seek for booking services of airplanes, buses and hotels reservation, and also generating tourist maps for the conference city. The graph also contains edges that marked with connected outputs and inputs of two connected atomic services, which represent valid semantic matches that the outputs of a web service can be passed as the inputs of its successors.

Although obtaining a correct combination of web service is essential, there are many ways to connect multiple web services chosen from a service repository. These web services provide different QoS in terms of availability, price, reliability and response time. Moreover, valid semantic matches between two connected web services could also have different matchmaking quality. For example, $GenerateMap$ service produce $StreetMap$ which is considered to be a valid semantic match to $TouristMap$, while $BusService$ produce $BusTicket$ which is considered to be a better valid semantic match to $BusTicket$. The descriptions of these matched resources are captured in a taxonomy depicted in Fig

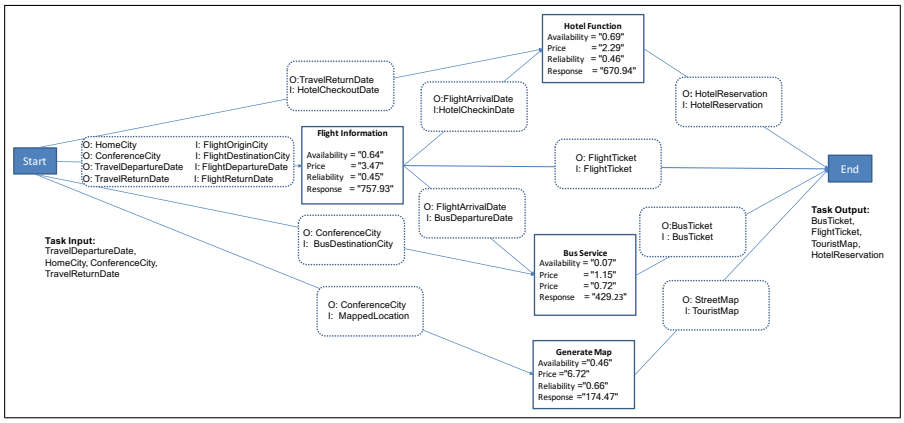


Fig. 1: An example of a web service composition

.2, where all the inputs and the outputs related to the web services are described for a travel planning problem. For example, the concept Map and its sub-concept $urbanMap$ are assigned with a $touristMap$ instance and $streetMap$ instance respectively in Fig .2. The valid semantic match is considered that an instance of $urbanMap$ can be an instance of Map, but not vice versa. To demonstrate the motivation of proposing a comprehensive quality, we give a simple example. The service $GenerateMap$ with a output ($StreetMap$) and a price ("6.72") in the web service composition can be replace by a service $GenerateTouristMap$ with a different output ($TouristMap$) and a different price ("16.87") available in the service repository. This composition solution results in a better semantic matchmaking quality, but price negatively contributes to the QoS. Therefore, to reach the optimal solutions, we need to measure these changes to the solutions considering matchmaking quality and QoS simultaneously for automatically generated solutions.

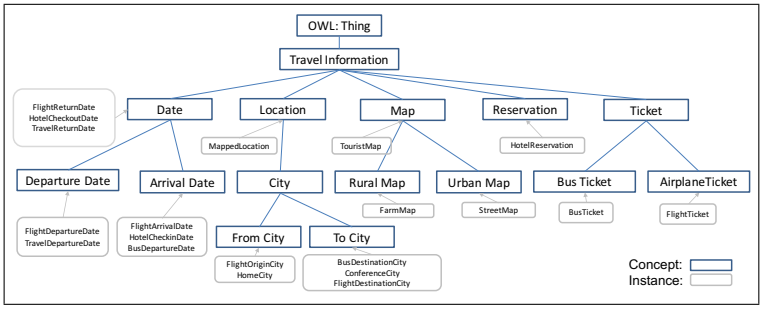


Fig. 2: A taxonomy of a travel planning domain

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: a semantic web service model, a semantic matchmaking model, a weighted DAG composition model, and QoS calculation model, and composition problem model.

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.QoS-aware web service composition [22,23,24,10]

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 the 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 [25]. $\mathcal{SR}$ 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 the returns of $type(a, b)$ 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 [25] are response time, cost, reliability and availability described as follows:*Response time* (t_s) measures the expected delay in seconds between the mo-

ment 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 [23] as follows:

A, R, C, T is aggregated a_s, r_s, c_s and t_n respectively 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 $T = MAX\{timePath_n | n \in \{1, \ldots, m\}\}$.

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 in concept similarity of two matched concepts. For example, two plugin types with different concept similarities: $publication \sqsubseteq printedmaterial$ and $romanticnovel \sqsubseteq printedmaterial$.

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

The $q_{sim}(a, b)$ 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 if we do not measure the similarities of neighbourhood concepts, which is not the matching type considered in this paper.

$$s_p = \frac{2N \cdot e^{-\lambda L/D}}{N_1 + N_2} \tag{1}$$

Definition 5. Given $fullmatch(a \Rightarrow b)$, $sm(a,b)$ is a function that returns a pair of values consisitng of $type(a,b)$ and $q_{sim}(a,b)$ defined in Formula (2).

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

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. Naturally, all these concepts can be captured 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 $I_{S_{src}}$ 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 e. Therefore, semantic matchmaking quality of one e, 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 futher 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 related to the composition task. 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. Semantic matchmaking types (Exact and Plugin) are considered for discovering desired web service in QoS-aware web service composition [10,22,23], but there are no measurements of matchmaking types and concept similarities. Consequently, weights for MT and S are considered to be 0 in our propopsed comprehensive evaluation model, it turns to a widely used QoS evaluation model in [10,22,23].

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

5.1 PSO-based Approach to QoS-Aware Semantic Web Service Composition

PSO has shown its efficiency in solving combinatorial optimisation problems [6]. Therefore, we will employ a PSO-based approach, which is considered to be easier to maintain the correctness of solutions compared to GP-based approaches that often require repairing or penalising the solutions [22]. Fig. 3 shows the overview of our approach with five steps. Step 1: The composition process is triggered by a composition goal defined in Subsection 3, which describes customers requirements both semantic matchmaking quality and QoS. Step 2: This composition goal are used to discover all relevant web services, which lead to a shrunken service repository that is subsequently used by PSO as a searching space. Step 3: A weighted graph representation is randomly built up from an initial service queue

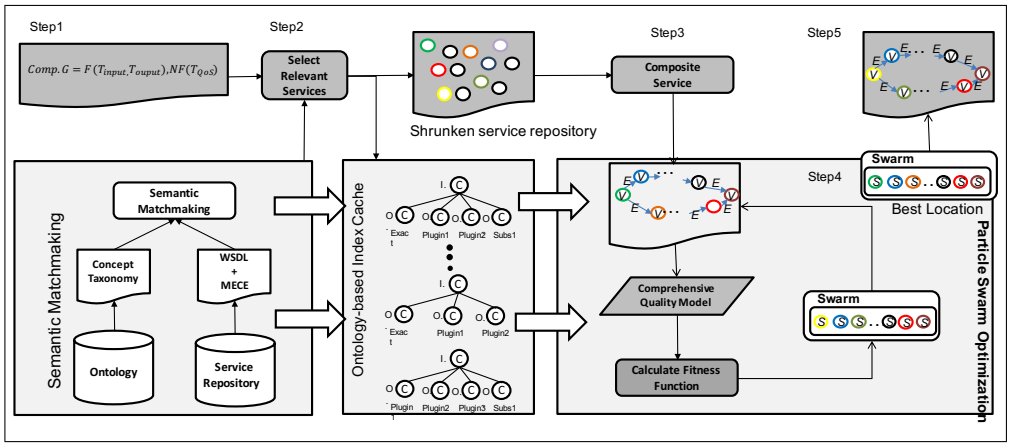


Fig. 3: Overview of POS-based approach to QoS-aware automated semantic web service composition.

that mapped to the particle's location, interleaving with semantic matchmaking process utilising ontology-based index cache. In the weighted graph, graph edges are assigned with semantic matchmaking quality as weights. Step 4: The fitness value of the weighted graph is evaluated to update the position of particle under PSO algorithm in Sect. 5.3, where the position is mapped to the index of service queue. later on, the updated service queue is used to decode a new weighted graph as the composition solution. Step 5. Lastly, the best position found in the searching space is selected and decode into the final optimised solution. This PSO-based approach is similar to [22], but we employ weighted graphs as a different solution presentation with explicit support for the proposed quality model.

5.2 Ontology-based Index Cached Optimisation

our PSO-based approach demands decoding processes from optimised queues to weighted DAGs, the bottlenecks of efficiently generating weighted DAGs lie in building edges and nodes, which are related to the cost of semantic quality calculation and the size of service repository respectively. To effectively construct weighted DAGs, we pre-calculate semantic matchmaking quality. The key idea of the index is to create a map using a pair of keys, output-related concepts and potentially matched input-related concepts with considering different levels of match types, and the map values stored mt_p and mt_s. Meanwhile, this index size could also be reduced by only considering the concepts related in the shrunken service repository, which means the index cache is filtered by those task-relevant web services. This optimised cache also contributes to less and constant time for weighted graphs building through the whole evolutionary process.

5.3 QoS-aware Semantic Web Service Composition Algorithm

ALGORITHM 1. Steps of the PSO-based Web service composition technique.

Input : relevant Web services rws
Output: Optimised service queue $queue$,Optimised Weighted Graph $OPTWG$

1: Map each relevant service to an index in the particle's position vector;
2: https://www.facebook.com/ Randomly initialise each particle in the swarm;
3: **while** *max. iterations not met* **do**
4: **foreach** *particles in the swarm* **do**
5: $queue \leftarrow$ map service in a descending order by particle's position vector;
6: $WG \leftarrow generateWeightedGraph()$;
7: Calculate the WG 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;
15: **return** $OPTWG$;

The overall algorithm investigated here is made up of the PSO-based web service composition algorithm 1 and the decoding algorithm 2. In algorithm 1, the idea is to translate the particle location produced by PSO into a service queue as an indirect representation, such that finding the best fitness of the weighted graph is to discover the optimised location of the particle in the search space. In PSO, the dimension of each particle equals to the number of relevant web services. The index of each services is mapped to a separate location component in a particle. Then services queue is sorted by the particles' position vector in the ascending order, from which we decode a weighted graph using Algorithm 2. It is a simple forward graph building algorithm, and this method can lead to more services and edges connected to the graph that must be removed. Also, semantic quality value are assigned to all the edges, which is calculated from quality aggregation function with given parameter matchmaking quality.

6 Experiment Design

In this section, a quantitative evaluation approach is adopted in our experiment design. The objectives of the evaluation are to (1) measure the effectiveness of

ALGORITHM 2. Create a composition weighted graph from a queue.

```
Procedure generateWeightedGraph()
    Input  : Task inputs I, task outputs O, Optimised service queue queue,
             IndexCache IndexCache
    Output: Weighted Graph WG
1:  WG ← null;
2:  WG ← new endNode(), new startNode();
3:  OutputSet ← {I};
4:  while all O ∉ OutputSet and queue! = null do
5:      foreach ws in queue do
6:          if ws.inputs ∈ OutputSet then
7:              foreach I in ws.inputs do
8:                  mt_P, s_P ← query IndexCache;
9:                  mt_L ← aggregation(mt_p);
10:                 s_L ← aggregation(s_p);
11:             sm_L ← { mt_L,s_L };
12:             WG.edge ← sm_L;
13:             WG ← new wsNode();
14:             OutputSet add {ws.outputs};
15:             queue.remove ws;
16:     remove danglingnodes;
17:     remove danglingedges;
18:     return WG;
```

the comprehensive quality model in automated semantic web service composition approach; and (2) compare solutions generated by a recent QoS-ware evaluation approach [10,21] with our evaluation method proposed in the paper; and (3) compare our PSP-based method with one existing GP-based approach [10].

We utilise benchmark dataset web service challenge 2009 (WSC09) [8] to perform the evaluation. WSC09 provides problems with five tasks corresponding to variable number of services, and ontologies. Therefore, it is a challenge dataset for measuring the scalability of our quality evaluation model. Table 1 presents the features of the WSC09 dataset. The number of concepts, individuals in the ontology and services in each data set is shown in the second, third, fourth column respectively. Also, we extend all the datasets with QoS attributes from service providers to enable our evaluation.

Table 1: Features of the WSC09 datasets

Dataset	No.Concept	No.Individual	No.Service
WSC09 01	1578	3102	572
WSC09 02	12388	24815	4129
WSC09 03	18573	37316	8138
WSC09 04	18673	37324	8301
WSC09 05	31044	62132	15211

The parameters were chosen based on general 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 accordingly. The mt_p is set to 1 (Exact match) and 0.75 (Plugin). In general, weight settings and parameter match type quality are decided by users' preferences.

7 Results and Analysis

7.1 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 [10,22,23]. 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.

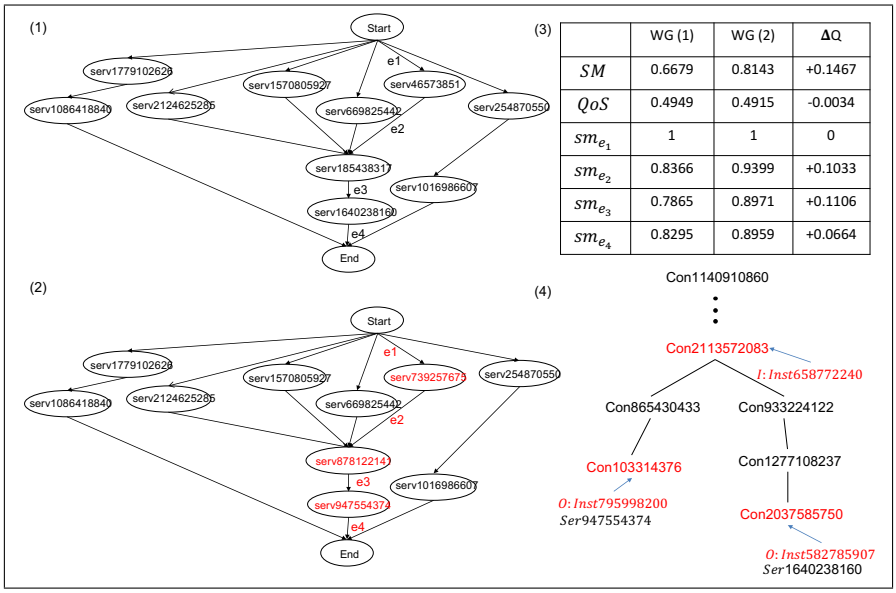
We observe an interesting pattern from Table 2. The mean values of QoS using QoS evaluation model are significantly higher than those using comprehensive quality evaluation model for Task 2, 3, 4 and 5, but the mean value of SM using comprehensive quality evaluation model are consistently significantly 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 Evaluation Model
Task1	SM	0.537375 ± 0.026709	0.557972 ± 0.009394 ↑
	QoS	0.557460 ± 0.015572	0.560444 ± 0.016424
Task2	SM	0.454383 ± 0.003287	0.462974 ± 0.004232 ↑
	QoS	0.479958 ± 0.001156 ↑	0.477192 ± 0.002528
Task3	SM	0.553818 ± 0.008180	0.609298 ± 0.005371 ↑
	QoS	0.494004 ± 0.001322 ↑	0.491262 ± 0.000862
Task4	SM	0.439757 ± 0.003728	0.460369 ± 0.000002 ↑
	QoS	0.484594 ± 0.001014 ↑	0.473354 ± 0.004422
Task5	SM	0.457988 ± 0.006459	0.463878 ± 0.001299 ↑
	QoS	0.476378 ± 0.000480 ↑	0.474970 ± 0.000656

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. 4 (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. 4 (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. 4 (4). In addtion, $Inst658772240$ is a required pa-

rameter of the *end* vertice as one of composition task outputs, which is related to concept $Con2113572083$. There exist $Inst582785907 \in Con2037585750 \sqsubseteq_3 Inst658772240 \in Con2113572083$ for e_4 in $WG(1)$ and $Inst795998200 \in Con103314376 \sqsubseteq_2 Inst658772240 \in Con2113572083$ for e_4 in $WG(2)$. Therefore, 2-level Subsumes is closer than 3-level Subsumes, $Inst795998200$ is closer to user's required output $Inst658772240$ compared with $Inst582785907$.



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

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

To evaluate the performance of our proposed PSO-based approach, we compare one GP-based approach [10] with our PSO-based method. we consider the links between child nodes and parent nodes as semantic web service links for measuring semantic matchmaking quality. Therefore, we evaluate both semantic matchmaking quality and QoS simultaneously using our proposed comprehensive quality model. To make a fair comparison with our PSO-based, we consider the same number of evaluations (3000 times) used in our PSO-based approach, we set the parameters settings for the GP-based approach using 30 individuals and 100 generations.

The Table 3 shows the mean fitness values of 30 independent runs accomplished by two methods. We employ statistical analysis to test the significant

differences in mean fitness value. The results show that the PSO-based approach performs better in four of five tasks (all the p-values are consistently smaller than 0.01).

Table 3: Mean fitness results for comparing GP-based approach

Dataset	PSO-based approach	GP-based approach
WSC09 01	0.559207 ± 0.012780 ↑	0.520659 ± 0.020758
WSC09 02	0.470083 ± 0.001106 ↑	0.459681 ± 0.002874
WSC09 03	0.550396 ± 0.012780	0.567915 ± 0.023447 ↑
WSC09 04	0.468942 ± 0.001670 ↑	0.431704 ± 0.009774
WSC09 05	0.469424 ± 0.000800 ↑	0.245222 ± 0.036885

8 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. 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)
4. 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)
5. Fensel, D., Facca, F.M., Simperl, E., Toma, I.: Semantic web services. Springer Science & Business Media (2011)
6. Fukuyama, Y.: Fundamentals of particle swarm optimization techniques. Modern heuristic optimization techniques: theory and applications to power systems pp. 71–87 (2008)
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.: Genetic programming for qos-aware web service composition and selection. Soft Computing pp. 1–17 (2016)
22. 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)
23. 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)
24. 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)
25. 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)