

# 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 to provide rich functionalities through language understanding and reasoning. 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 often concern either semantic web service composition or QoS-aware web service composition. Therefore, we propose a general comprehensive quality model considering semantic matchmaking quality and QoS simultaneously that can achieve a more desirable trade-off in consideration of both sides by conducting an experiment comparing with promising QoS-aware model. Further more, we develop a PSO-based service composition approach with explicit support for the comprehensive model, and compare the PSO-based method with one recent GP-based method to show its performance in finding a more optimised solution.

## 1 Introduction

*Web service composition* pertains to a combination 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 has 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 aspect, on the other hand, demands further research. Through ontology-based semantics [13], web services can understand and better collaborate with each other. There are many ontology languages and formats for semantic service descriptions, such as OWL-S, WSMML, and SAWSDL [16], which make “machine understanding” possible through identifying and matching semantic similarities in input/output parameters of web services in heterogeneous environments. The second challenge (2) is related to finding *optimised solutions* to QoS. This problem gives birth to *QoS-aware service composition* that considers the composition of service-level agreements (SLA) [19] involving a collection of SLA rules and policies for supporting QoS-based composition.

Existing works on service composition focus mainly on addressing one of challenges above. Many works have been conducted to optimise the quality of compositions under a pre-defined abstract workflow, which is generally considered as a *semi-automated Web service composition* approach [1,15]. Meanwhile, many research works consider the possibility of generating a composition plan automatically in discovering and selecting suitable web services, which are considered to be NP-hard [12]. *Semantic web services composition* is distinguished from the syntactic service composition, which supplements well defined semantic descriptions from the concept of ontologies, instead of parameters value. In the past few years, substantial works have been done on semantic web service composition [2,3,11]. However, few works have enabled truly automatic semantic web service composition, where both QoS and quality of semantic match making will be optimised simultaneously and achieved 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 both QoS optimisation and semantic matchmaking quality optimisation in 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 in considering different matching types with corresponding concept similarities.
2. To propose a PSO-based service composition algorithm that utilises the proposed quality model. To do that we will first propose a representation of semantic service composition, which can model the quality of semantic matchmaking and QoS together.
3. To evaluate our proposed approach, we conduct experiments to compare our comprehensive quality model with one widely used QoS evaluation model.

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 [7,18,10,23,24,25]. However, few researchers address both semantic matchmaking quality and QoS requirements at the same time 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) for enabling the interoperability of web services, where the number of web services or length of the graph is minimised to reach the optimised graph-based composition solutions. However, this evaluation approach does not guarantee optimised qualities of composition solutions for QoS.

QoS-aware web service composition is studied using traditional approaches or Evolutionary Computation (EC) techniques. Qi et al. [18] propose a heuristic service composition 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 for QoS. but 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 problems. Gupta et al. [7] employ an Genetic Algorithm using a set of binary strings as individuals, 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 is introduced in [10] to generate solutions with functional correctness. In particular, the greedy search is used to generate directed acyclic graphs (DAGs) as composition solutions, which is further transferred to the tree structure for initialisation and mutation using unfolding techniques. 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 as a composition solution. These QoS-aware approaches does not consider semantic matchmaking quality, which could lead to over-general outputs to be produced by selected services, The finding is evidently supported by our first comparison experiment. In fact, customers' perspectives, application domains and ontology granularity all could have significant impact on the outputs requested by users. In some scenarios, the output is too broad to bear any specific meaning for the

customers, even though those web services selected leads to a very good overall QoS.

Few works [4,9,17] employ EC technique for considering both semantic match-making quality and QoS simultaneously. Lecue et al. [9] employ Genetic Algorithm for semi-automated web service composition considering semantic match-making quality and QoS, where the semantic matchmaking quality is measured by the quality of semantic links that requires formal definition of ontology in Description Logic. This evaluation takes huge cost and time for the domain experts to establish required ontology. Another GA-based approach work [4] employs a sequence of fitness functions are used in the binary selection of chromosomes, where semantic matchmaking quality is evaluate in two fitness functions considering concept similarity and parameter similarity respectively, but different semantic matchmaking types are ignored in the evaluation model. The work [17] as an immune-inspired web service composition approach employ indirect representation that a binary alphabet to encode a planning graph, where the semantic matchmaking quality is measured by the similarity using information retrieval technique.

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 general approaches that achieve a more desirable balance in consideration of both semantic matchmaking quality and QoS. Therefore, we proposed a general comprehensive quality model to fill the gaps, and then we proposed a POS-based method with explicit support for the comprehensive model to find optimal compositions.

### 3 Motivation and Problem Description

#### 3.1 Motivation

The aim of web service composition considered in this paper is to automatically select the optimal solutions in considering both QoS and semantic matchmaking quality by a user's request. The request is to provide inputs and obtain outputs, the composition results that should meet the optimal requirements above. A motivating example of this problem is shown in Fig 1. The figure is a DAG represent 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 from scholars who seeks for booking services of airplanes, buses and hotels reservation, and also the tourist map for the conference city. The graph also contains edges that marked with connecting outputs and inputs, which represent valid semantic matches where 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 chain web services to reach users' needs with different web services

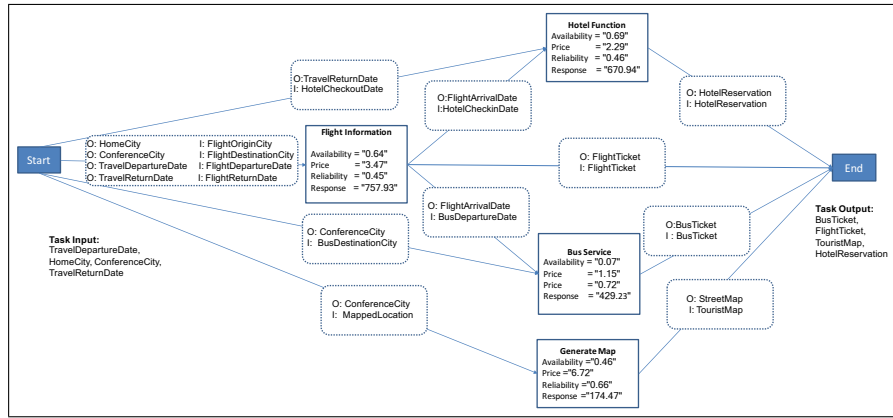


Fig. 1: An example of a web service composition

in a service repository. These web services have different QoS (such as availability, price, reliability and response time). Moreover, the valid semantic matches between two web services could be associated with different semantic match-making quality. For example, Generate Map service produce StreetMap which is considered to be a valid semantic match to TouristMap required by users. Fig. 2 represents a taxonomy of all the inputs and the outputs described in Fig. 1 for a travel planning domain, where the concept Map and its sub-concept urbanMap are assigned with a touristMap instance and streetMap instance respectively. The valid semantic match is considered that an instance of urban Map must be an instance of Map, not vice versa. If a web service *GenerateTouristMap* found in the service repository that produces TouristMap as an output. I consider its semantic matchmaking quality is higher than that of *GenerateMap*. However, *GenerateTouristMap* might have a very bad QoS.

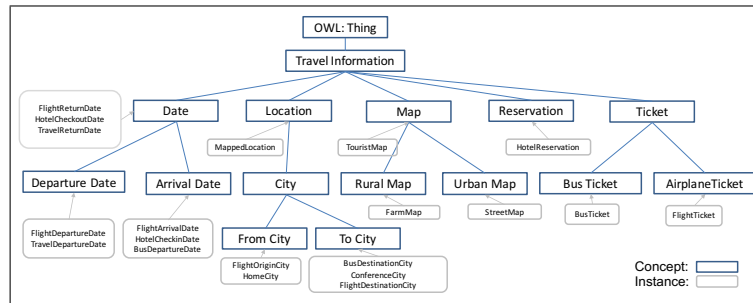


Fig. 2: A taxonomy of a travel planning domain

### 3.2 Problem Formulation

We herein give formal definitions of the assumptions the concepts regarding the semantic web service composition problem utilised in our approach that consists of three main concepts: a semantic web service model, a semantic matchmaking model, a weighted DAG composition model, and QoS calculation model.

**Semantic Web Service Model.** The semantic web service model represent web services with functional descriptions, and how interoperability of web services enable matchmaking to automatically generate 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  $O = \{C, Taxonomy \mid I_T^C, O_T^C \subseteq C\}$ .

**Definition 2.** A semantic web 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  $I_S^C, O_S^C \subseteq C$ .  $QoS_S = \{t_s, c_s, r_s, a_s, \dots\}$  is a set of non-functional attributes of  $S$ , it typically refers to response time  $t_s$ , cost  $c_s$ , reliability  $r_s$ , and availability  $a_s$  [26].  $\mathcal{SR}$  is a set of  $S$ , where all the  $S$  are registered.

**Semantic Matchmaking Model.** The mechanism to compose web services relays on the semantic inputs and outputs, which enable inputs to be matched by outputs. To measure the quality of these matches, we utilise different matchmaking types (*exact*, *plugin subsumes* and *fail* [14]) into consideration for exploiting the semantics of inputs and outputs.

**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 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  $type(a, b)$  to define full matchmaking between two concept-related input and output of two services.

**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.

**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  $End = \{O_T, \phi, \phi\}$ .

$E = \{e_1, e_2, \dots, e_n\}$  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  $S_{tar}$ .

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

**QoS Model in Weighted DAG**

### 3.3 Quality of Service

According to [26], four most often considered QoS parameters are response time, cost, reliability and availability described as follows:

- *Response time* ( $T$ ) measures the expected delay in seconds between the moment when a request is sent and the moment when the results are received.
- *Cost* ( $C$ ) is the amount of money that a service requester has to pay for executing the web service
- *Reliability* ( $R$ ) is the probability that a request is correctly responded within the maximum expected time frame.
- *Availability* ( $A$ ) is the probability that a web service is accessible.

The aggregation value of QoS attributes for a composite service varies with respect to different constructs, which reflects how services associated with each other in a service composition [26]. Here we consider two composition constructs: sequence and parallel constructs, in building the composition web service. The QoS calculation models are described as follows:

**Sequence construct** all involved atomic services associated with a sequence construct are executed in a definite sequence order. The aggregation value for total time ( $T$ ) and total cost ( $C$ ) is as the sum of time and cost of web services involved respectively. The overall availability and reliability in a sequence construct are calculated by multiplying their corresponding availability and reliability of each web service in probability theory. This construct is shown in Fig. 3.

**Parallel construct** Web services in a parallel construct are executed concurrently. The QoS aggregation value for total cost, availability and reliability are the same as these in sequence construct while the Total time ( $T$ ) is determined by the most time-consuming path in the composition flow of the solution. This construct is presented in Fig. 4.

## 4 Comprehensive Quality Model

In this section, we propose a general comprehensive quality-aware service model in consideration of both semantic matchmaking quality and QoS for semantic web service composition. This model overcome the disadvantages of current prevailing QoS-aware quality model that ignores quality of semantic matchmaking.

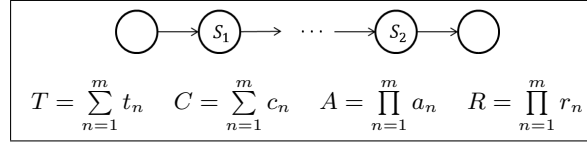


Fig. 3: Sequence construct and calculation of its QoS properties [25].

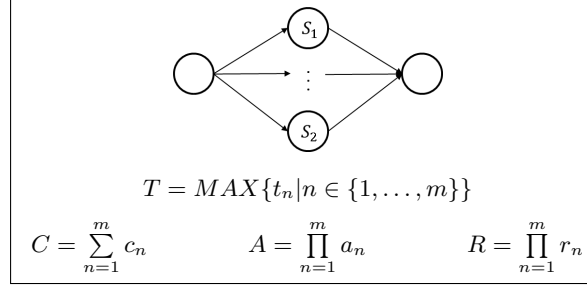


Fig. 4: Parallel construct and calculation of its QoS properties [25].

#### 4.1 Comprehensive Quality Model

**Semantic web Service link.** In Fig. 5,  $S_1$  and  $S_2$  are linked together if exists any parameter matchmaking between Output of  $S_1$  and Input of  $S_2$ . Therefore, measuring semantic matchmaking quality ( $SM$ ) is to measure the overall quality of semantic web service link ( $sm_L$ ), which is aggregated from the quality of parameter matchmaking ( $sm_P$ ). This semantic web service link is adopt from [9].

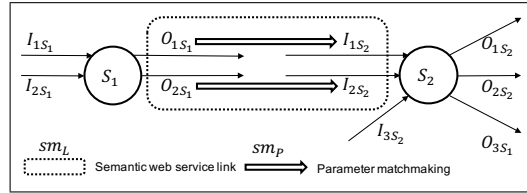


Fig. 5: An example of semantic web service link

**Semantic web matchmaking quality model.** Due to that the discretisational characteristics of different match types and values assigned to matching types are driven by the cost of data integration and manipulation [9], partial ordering match types are considered to be one factor for quality of parameter matchmaking, denoted as  $mt_p$ . For example, Exact matching type demands less time for computation compared to that of Plugin match. Another factor in our proposed model is concept similarity, which could be evaluated based on the



edge counting method defined in [20]. This formula (1) is used to estimate the concept similarity between matching parameters, which is denoted as  $s_p$ .  $N_1$  and  $N_2$  measure the distances from an output-related concept and an input-related concept to the top node of a taxonomy respectively, and  $N$  measure the distances from the closest common ancestor of these two matched input-related and output-related concepts.  $\lambda$  is set to 0 if we do not measure the similarities of neighbourhood concepts that is not the matching levels considered in this paper. Consequently, given the quality of parameter matchmaking type and quality of parameter concept similarity, the quality of parameter matchmaking is defined by Formula (2).

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

$$sm_p \doteq (mt_p, s_p) \quad (2)$$

Further more, since semantic web service link represent services connections, where quality of semantic web service link ( $sm_L$ ) is considered to be semantic matching quality on service level by the aggregation of parameter matchmaking quality. Therefore,  $sm_L$  is defined in formula (3), where  $sm_L$  and  $s_L$  are the average value of all involved  $mt_p$  and  $s_p$  respectively.

$$sm_L \doteq (mt_L, s_L) \quad (3)$$

**Comprehensive quality model.** The comprehensive quality model consider both semantic matchmaking quality and QoS. The semantic matchmaking quality refers to  $sm_L$ . Consequently, the comprehensive quality model is defined in Formula (4), which could be further broken down into Formula (5).

$$cq \doteq (sm_L, QoS_s) \quad (4)$$

$$cq \doteq (mt_L, s_L, a_s, r_s, c_s, t_s) \quad (5)$$

**Semantic matchmaking quality aggregation.** The quality aggregation is defined based on the constructs of composition web services in consideration of semantic matchmaking quality and QoS. The quality on composition solution is further calculated by following the formula (6)(7), and QoS aggregation are discussed in Sect. 3.3

$$MT = \prod_{n=1}^m mt_{L_n} \quad (6)$$

$$S = (\sum_{n=1}^m s_{L_n}) / m \quad (7)$$

## 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.

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

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

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

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

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

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

## 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 [23]. Fig. 6 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 shrunk 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 that mapped to the particle's location, interleaving with semantic matchmaking process utilising ontology-based index cache. In the weighted graph, graph

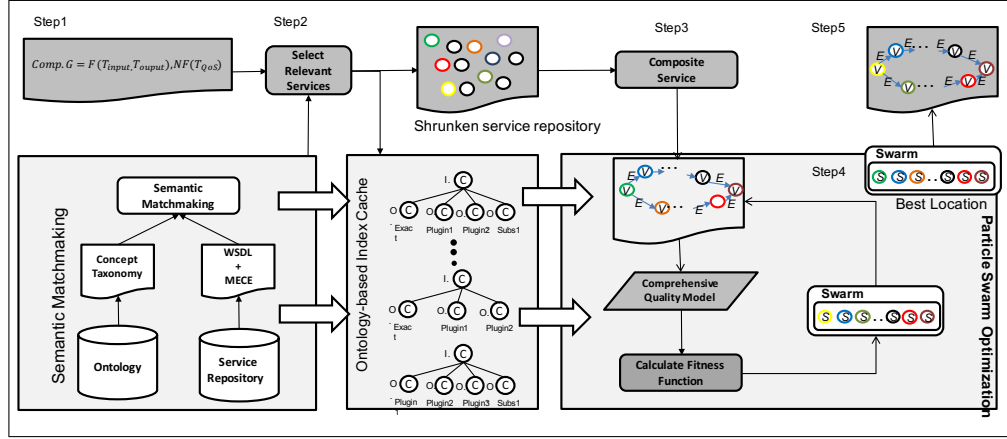


Fig. 6: Overview of POS-based approach to QoS-aware automated semantic web service composition.

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 [23], 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 shrunk 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.

---

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: 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**  $OPTWG$ ;

---



---

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:  $WG \leftarrow null$ ;
- 2:  $WG \leftarrow new endNode(), new startNode()$ ;
- 3:  $OutputSet \leftarrow \{I\}$ ;
- 4: **while** all  $O \notin OutputSet$  and  $queue \neq null$  **do**
- 5:   **foreach**  $ws in queue$  **do**
- 6:     **if**  $ws.inputs \in OutputSet$  **then**
- 7:       **foreach**  $I in ws.inputs$  **do**
- 8:           $mt_P, s_P \leftarrow query IndexCache$ ;
- 9:           $mt_L \leftarrow aggregation(mt_P)$ ;
- 10:          $s_L \leftarrow aggregation(s_P)$ ;
- 11:          $sm_L \leftarrow \{mt_L, s_L\}$ ;
- 12:          $WG.edge \leftarrow sm_L$ ;
- 13:          $WG \leftarrow new wsNode()$ ;
- 14:          $OutputSet add \{ws.outputs\}$ ;
- 15:          $queue.remove ws$ ;
- 16:   remove  $dangling nodes$ ;
- 17:   remove  $dangling edges$ ;
- 18: **return**  $WG$ ;

---

### 5.3 QoS-aware Semantic Web Service Composition Algorithm

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 the comprehensive quality model in automated semantic web service composition approach; and (2) compare solutions generated by a recent QoS-ware evaluation approach [10,22] 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 [21] 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 with QoS Evaluation Model

QoS-aware web service composition [23,24,25,10] do consider two semantic match-making types (Exact and Plugin) for discovering desired web service, but there is no measurement of matchmaking types and concept similarity, so the weights for  $MT$  and  $S$  are considered to be 0. Consequently, our evaluation mode become a widely used fitness function [10,23],  $Fitness = w_1\hat{A} + w_2\hat{R} + w_3(1 - \hat{T}) + w_4(1 - \hat{C})$  when  $\sum_{i=1}^4 w_i = 1$  for QoS-aware web service composition. To compare the composition solution generated by our approach with QoS-aware approach. We look at mean value of  $MT$ ,  $S$  and  $QoS$  at optimum at the 100th generation for two approaches. In QoS-aware approach,  $MT$  and  $S$  are recorded, and  $QoS$  is normalised from 0 to 0.5 to make it comparable, besides that, all weights in QoS-aware fitness function are set equally to 0.25.

We observe an interesting pattern from Table 2 using statistic analysis: mean value of  $MT$  and  $S$  at optimum by our approach is consistently higher than those by QoS-aware approach. Meanwhile,  $QoS$  generated by QoS-aware approach can achieve a more desirable tradeoff in both semantic matchmaking quality and QoS.

To better understand that our evaluation approach could generate solutions having better semantic matchmaking quality with a slight trade-off in  $QoS$ , we demonstrate an example solution to Task 3. Fig. 7 (1) and (2) are two weighted DAGs as best solutions found employing QoS-aware approach and Comprehensive quality-aware method respectively. Two approaches generate exactly the same service workflow structure while some service vertices and edges denoted in red are different. We list the semantic matchmaking quality represented by all edges ( $e_1$  to  $e_4$ ), overall semantic matchmaking quality and QoS in Fig. 7 (3). To demonstrate how different the semantic matchmaking quality is, we analyse  $e_4$  that has the smallest  $\Delta Q$ , which reveals the gain (positive  $\Delta Q$ ) or loss (negative  $\Delta Q$ ) for the listed qualities using our approach. The  $e_4$  has two different source service vertices ( $Ser1640238160$  and  $Ser1947554374$ ) and the same *end* vertices in two solutions. The outputs of two different source service are marked on the related taxonomy in Fig. 7 (4),  $Ser1640238160$  and  $Ser1947554374$  are service with output concept-related parameters  $Inst795998200$  and  $Inst582785907$  corresponding to two concepts  $Con103314376$  and  $Con2037585750$  respectively, and  $Inst658772240$  are the required parameter for the *end* vertice, which is related to concept  $Con2113572083$ . There exist  $Inst795998200 \in Con103314376 \sqsubseteq_2 Inst658772240 \in Con2113572083$  and  $Inst582785907 \in Con2037585750 \sqsubseteq_3$

Table 2: Mean Quality for comprehensive quality-aware methods and QoS-aware approach

| WSC09 |            | QoS-aware<br>Evaluation            | Comprehensive<br>Quality Evaluation |
|-------|------------|------------------------------------|-------------------------------------|
| Task1 | <i>MT</i>  | 0.189787 $\pm$ 0.039278686         | 0.221862 $\pm$ 0.009582 $\uparrow$  |
|       | <i>S</i>   | 0.884962 $\pm$ 0.014140            | 0.894082 $\pm$ 0.009206 $\uparrow$  |
|       | <i>QoS</i> | 0.278730 $\pm$ 0.007786            | 0.280222 $\pm$ 0.008212             |
| Task2 | <i>MT</i>  | 0.001795 $\pm$ 0.000719            | 0.001977 $\pm$ 0.001566 $\uparrow$  |
|       | <i>S</i>   | 0.906971 $\pm$ 0.005855            | 0.923970 $\pm$ 0.006898 $\uparrow$  |
|       | <i>QoS</i> | 0.239979 $\pm$ 0.000578 $\uparrow$ | 0.238596 $\pm$ 0.001264             |
| Task3 | <i>MT</i>  | 0.158526 $\pm$ 0.014028            | 0.245830 $\pm$ 0.007761 $\uparrow$  |
|       | <i>S</i>   | 0.949109 $\pm$ 0.002331            | 0.972765 $\pm$ 0.002980 $\uparrow$  |
|       | <i>QoS</i> | 0.247002 $\pm$ 0.000661 $\uparrow$ | 0.245631 $\pm$ 0.000431             |
| Task4 | <i>MT</i>  | 0.000000 $\pm$ 0.000000            | 0.000004 $\pm$ 0.000002 $\uparrow$  |
|       | <i>S</i>   | 0.879514 $\pm$ 0.007456            | 0.920733 $\pm$ 0.000001 $\uparrow$  |
|       | <i>QoS</i> | 0.242297 $\pm$ 0.000507 $\uparrow$ | 0.236677 $\pm$ 0.002211             |
| Task5 | <i>MT</i>  | 0.000042 $\pm$ 0.000030            | 0.000078 $\pm$ 0.000020 $\uparrow$  |
|       | <i>S</i>   | 0.915933 $\pm$ 0.012888            | 0.927678 $\pm$ 0.002578 $\uparrow$  |
|       | <i>QoS</i> | 0.238189 $\pm$ 0.000240 $\uparrow$ | 0.237485 $\pm$ 0.000328             |

*Inst658772240*  $\in$  *Con2113572083*. It is obvious that our approach selects the service providing *Inst795998200* that are closer to the users' required outputs.

## 7.2 Comparison Test with GP-based approach

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

The Table 3 shows the mean fitness values 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).

At last, We also compare the average fitness value with optimum through the evolutionary process for both two approaches over 30 independent runs. Fig. 8 is an example of convergence rate for Task 3, where the behaviour of PSO-based approach presents a very clear evidence of fast convergence and reaches a better-optimised solution, while the performance of GP is barely satisfactory, as GP-based method improves its fitness value gradually.

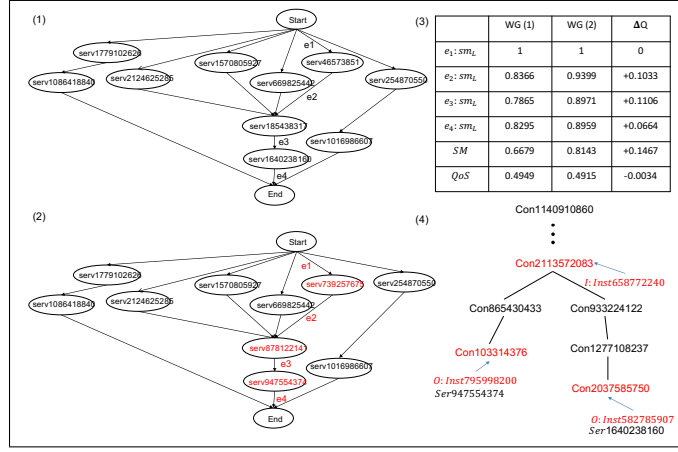


Fig. 7: Example Comparison of solutions to Task 3 using two different evaluation model.

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

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

## 8 Conclusion

This work introduces a general comprehensive evaluation model for considering semantic matchmaking quality and QoS. We proposed a PSO-based service composition approach utilising our proposed quality mode that can achieve a desired balance of both quality aspects. Also, we compare one GP-approach with our PSO-based method to show our performance results in finding more optimised solution. 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)



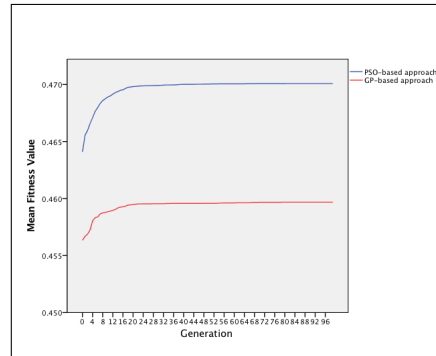


Fig. 8: An example of comparing convergency rate of PSO and GP approaches for task 3

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. Sahai, A., Machiraju, V., Sayal, M., Van Moorsel, A., Casati, F.: Automated sla monitoring for web services. In: *International Workshop on Distributed Systems: Operations and Management*. pp. 28–41. Springer (2002)
20. Shet, K., Acharya, U.D., et al.: A new similarity measure for taxonomy based on edge counting. *arXiv preprint arXiv:1211.4709* (2012)
21. 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)
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)