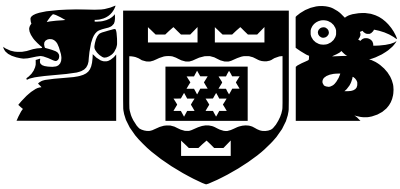
VICTORIA UNIVERSITY OF WELLINGTON
Te Whare Wānanga o te Ūpoko o te Ika a Māui



School of Engineering and Computer Science
Te Kura Mātai Pūkaha, Pūrorohiko

PO Box 600
Wellington
New Zealand

Tel: +64 4 463 5341
Fax: +64 4 463 5045
Internet: office@ecs.vuw.ac.nz

Comprehensive Quality-Aware Semantic Web Service Composition

Chen Wang

Supervisors: Dr. Hui Ma and Dr. Aaron Chen

July 17, 2017

Submitted in partial fulfilment of the requirements for PhD.

Abstract

.....

Contents

Figures

Chapter 1

Introduction

1.1 Problem Statement

Service-oriented computing (SOC) is a novel computing paradigm that employs services as fundamental elements to achieve the agile development of cost-efficient and integrable enterprise applications in heterogeneous environments [55, 56]. One of the primary purposes of SOC is to overcome conflicts due to diverse platforms and programming languages to make integrable and seamless communication among those existing or newly built independent services. *Service Oriented Architecture* (SOA) could abstractly realise service-oriented paradigm of computing. This accomplishment has been contributing to reuse of software components, from the concept of functions to units and from units to services during the evolution of development in SOA [9, 52]. One of the most typical implementation of SOA is *web service*, which is designated as "modular, self-describing, self-contained applications that are available on the Internet" [15]. Several standards play a significant role in registering, enquiring and grounding web services across the web, such as UDDI [14], WSDL [36] and SOAP [23]. *Web service composition* aims to loosely couple a set of web services to provide a value-added composite service that accommodates complex functional and non-functional requirements of service users.

Two most notable challenges for web service composition are ensuring interoperability of services and achieving Quality of Service (QoS) optimisation [23]. *Interoperability* of web services presents challenges in syntactic and semantic dimensions. On the one hand, the syntactic dimension is covered by the XML-based technologies [76], such as $WSDL$ and $SOAP$. In this dimension, most services are composed together merely based on the matching of input-output parameters. On the other hand, the semantic dimension enables a better collaboration through ontology-based semantics [51], in which many standards have been established, such as OWL-S [47], Web Service Modeling Ontology (WSMO) [37], SAWSDL [33], and Semantic Web Services Ontology (SWSO) [59]. This dimension brings around some other underlying functionality of services (i.e., precondition and postcondition) that could effect the execution of web services and their composition. The interoperability challenge gives birth to *Semantic web services composition*, as a different technique from traditional service composition methods (i.e., only syntactic dimension is presented in web services). Since the quality of semantic matchmaking is always used to measure the goodness of interoperability while composing services. Another challenge is related to QoS optimisation, where QoS represents non-functional attributes of service composition (e.g, cost, time, reliability and availability). Often, a global search is employed to minimise the cost and maximise the reliability of composite services. This challenge gives birth to *QoS-aware service composition* that aims to find composition solutions with optimised QoS. Furthermore, QoS-aware service composition problem is facilitated by *Service Level Agreement (SLA)* [65], i.e., binding

constraints on QoS. This results in the *SLA-aware web service composition*, e.g., constraints on cost, execution time, availability and reliability are separately specified with both lower and upper bounds.

Apart from the two notable challenges discussed above, the environment of service composition is changing in the real world, rather than *static*. E.g, QoS values of services being composed of are fluctuating over time. Services chosen at the planning stage may not available to be invoked at the runtime. New services can also be registered in the service repository from time to time. Existing techniques for *static web service composition* can not support such a changing environment. Therefore, *Dynamic web service composition* become a very demanding research field with a growing interest and a practical value. Particularly, some mechanisms are required to be developed to automatically detect the changes or recover from the faults [12]. Additionally, in context of semantic web service composition, semantics of web services can make the problem of dynamic web service composition more complicated due to the changes in the ontology.

Different approaches [17, 19, 27, 38, 45, 60, 64, 77, 70] have been proposed to solve those composition problems discussed above and they can be classified into two main categories: *semi-automated web service composition* and *fully automated web service composition*. The first composition problem requires human beings to manually create abstract workflows. Generally, researchers assume the pre-defined abstract workflow is given and provided by the users. The optimisation problem in this approach turns to selecting the concrete services with the best possible quality to each abstract service slot in a given workflow. Due to a tremendous growth in industries and enterprise applications, the number of web services has increased dramatically and unprecedentedly. The process of conducting abstract workflows manually is fraught with difficulties in effectively and efficiently solving composition problems, such as QoS-aware service composition problem. Therefore, full automation of composition process is introduced in web service composition for less human intervention, less time consumption, and high productivity [61]. The differences in fully automated approach is that an abstract workflow is not provided, but generated while service are being selected.

Generating composition plans automatically in discovering and selecting suitable web services is a NP-hard problem [50], which means the composition solution is not likely to be found with reasonable computation time in a large searching space. *Artificial Intelligence (AI) planning-based approaches*, *Evolutionary Computation (EC) techniques* and hybrid techniques have been introduced to handle this problem. AI planning problem is utilised to solve automated web service composition problems as a plan making process, from initial states to a set of actions to desired goal states-composite web services, where services are considered as actions triggered by one state (i.e., inputs) and resulted in another state (i.e., outputs). In the second approach, heuristics have been employed to generate near-optimal solutions using a variety of EC techniques have been used in this context, e.g., Genetic Algorithms (GA), Genetic Programming (GP) and Particle Swarm Optimisation (PSO). EC-based techniques have been effectively developed to solve *QoS-aware web service composition* problems with different structures for solution representations. Many representations have been carefully investigated in QoS-aware web service composition problems, since they could significantly affect the performance of fully automated service composition systems. In the third approach, a hybrid of AI planning-based approaches and EC-based approaches [17, 45] have been proposed to fulfil the correctness in constructing workflows under users' constraints, while the quality of composition solutions (e.g., QoS) are also optimised according to users' requirements. From the literature, hybrid approaches generally outperform EC-based methods for finding more optimal solutions in the domain of automated QoS-aware web service composition. It becomes a very promising approach to investigate. As summaried here, a

few previously addressed challenges for fully automated service compositions lies in jointly optimising QoS and semantic matchmaking quality, dealing with multi-objective optimisation, handling environment changes in dynamic service composition, and composing semantic web services. Those challenges are addressed with key limitations in Section 1.2.

The overall goal of this thesis is to propose hybrid approaches to comprehensive-quality aware automated web service composition. This comprehensive quality aims to jointly optimise semantic matchmaking quality and QoS. Meanwhile, this new approach also tackles several service composition problems, such as multi-objective optimisation, dynamic web service composition and semantic web service composition based on preconditions and postconditions.

1.2 Motivations

The motivations of this proposed research lies in the requirements from five key aspects that simultaneously account for. 1. *Various techniques of hybridisation*. 2. *Hybridisation of quality of semantic matchmaking and quality of service*. 3. *Muti-objective optimisation*. 4. *Dynamic semantic service composition*. 5. *Service composition based on preconditions and effects*. Herein these requirements are explicitly discussed below.

Various Techniques of Hybridisation

Various techniques have been utilised to solve service composition problems, such as AI planning, local searching and EC-based techniques [22, 57, 60, 70]. AI planning is a prominent technique for handling web service composition problems while always ensuring the correctness of the solution (i.e., all the inputs of involved services are satisfied) [70]. Local search is a exhaustive search technique for solving optimisation problems. In local search, solutions keep moving to the neighbour solutions, driven by some local maximisation criterion until near-optimal solution found [57]. However, this technique has the shortage of being trapped in local optimal. On the other hand, EC-based techniques are outstanding at solving combinatorial optimisation problems for finding the globally optimised solutions in large searching spaces. To take the benefits from various techniques, various techniques of hybridisation allows escaping local optima easily and improving the rate of convergence rate [63].

Traditionally, various techniques are employed to handle service composition problem independently in existing works. Many researchers investigated AI planning techniques for service composition problems using classical planning algorithm, where inputs, outputs, preconditions and effects are well defined along with the actions (i.e, services) [46, 58]. On the one hand, AI planning ensures both the correctness of functionality and satisfactory of constrains, but it is always considered to be less efficient and less scalable, incapable of solving complicated optimisation problems [57]. On the other hand, some researchers combined AI planning and local search to handle optimisation problems, e.g., a combination of Graphplan [8] and Dijkstras algorithm is proposed by [22] to achieve a correct solution with optimised QoS. Yet, many EC techniques have been utilised to handle service composition problems for global optimal solutions. A few researchers also combine both local search and EC-based techniques for efficiently finding composition solution with optimised QoS [57]. From the techniques discussed above, they are problem-specific for either optimising QoS or number of services. In this thesis, more complicated and realistic service composition problems are addressed. To with cope this composition problems featured in all the motivations discussed below (i.e comprehensive quality-aware service composition, multi-objective optimisation, dynamic service composition, and composing semantic web services based on

preconditions and postconditions), new hybrid methods must be proposed to adapt in the problems specified in the thesis. For example, a hybrid method for jointly optimising both semantic matchmaking quality and QoS.

Hybridisation of Quality of Semantic Matchmaking and Quality of Service

Web service compositions are optimised by the well known non-functional attributes (i.e, QoS), when services are composed together based on outputs provided by one service and inputs required by another service. In the domain of semantic web services, often, the provided information (i.e., outputs) does not perfectly match the required information (i.e., inputs) according to their semantic descriptions [40]. The quality of the matches (i.e., quality of semantic matchmaking) are one part of the goal for achieving service compositions [38]. Therefore, a hybridisation of QoS and semantic matchmaking quality becomes a combinatorial optimisation problem in web service composition. One motivated example from the practical perspective is explained here: many different service compositions can meet a user request but differ significantly in terms of QoS and semantic matchmaking quality. For example, in the classical travel planning context, some component service must be employed to obtain a travel map. Suppose that two services can be considered for this purpose. One service S can provide a street map at a price of 6.72. The other service S' can provide a tourist map at a price of 16.87. Because in our context a tourist map is more desirable than a street map, S' clearly enjoys better semantic matchmaking quality than S but will have negative impact on the QoS of the service composition (i.e., the price is much higher). One can easily imagine that similar challenges frequently occur when looking for service compositions. Hence, a good balance between QoS and semantic matchmaking quality is called for.

Existing works on service composition focus mainly on addressing only one quality aspect discussed above. For the semantic matchmaking quality, it is mainly addressed in works that focus on the discovery of atomic services, i.e., one-to-one matching of user requirements to a single service. Some works [6, 10, 49] on semantic service composition utilise semantic descriptions of web services (e.g., description logic) to ensure the interoperability of web services, but the goal of the composition is often to minimise the number of services or the size of a graph representation for a web service composition. These approaches do not guarantee an optimised QoS of service compositions. On the other hand, huge efforts have been devoted to studying QoS-aware web service composition [16, 19, 27, 45, 60, 77]. Some of these works do consider the semantic matchmaking while composing solutions, but do not recognise the importance of semantic matchmaking quality for optimising service composition. Therefore, it is not sufficient to only consider one quality aspect for optimising service composition. For these reasons, there is a need to device an comprehensive quality model for jointly optimising the two quality aspects. Apart from that, existing solutions representations may not effectively and efficiently handle this new optimisation problem, so new solution representations need to be proposed to maintain the required information for the optimisation of the two quality aspects using hybrid methods.

Multi-Objective Composition Optimisation

EC-based approaches for handling web service composition problems fall into two groups, depending on the different goals of optimisation for either a single objective or multiple objectives. In single-objective service compositions, one composition solution is always returned by a composition task, where the preferences of each quality component within the single objective (e.g., a weighted sum of different quality criteria) is known by users. However, users do not always have clear preferences when many quality criteria are presented.

Therefore, multi-objective is a natural requirements from users to provide a set of trade-off solutions that concern about the conflicting and independent quality criteria. E.g., Premium users do not care cost as much as price-sensitive users do, so premium users usually may prefer a composition solution with lowest execution time, rather than one with a relatively lower execution time without exceeding a budget. Therefore, a multi-objective fully automated service composition approach is very demanding for providing a set of solutions due to the following two reasons: first, the preferences of different quality is not clear and hard to determine in advance; second, single-objective optimisation using weighted sum method can not reach solutions in the non-convex regions [31].

Existing research on the automated web service composition mainly concentrates on single objective problems for QoS-aware web service compositions. I.e., there is only one solution promoted by a unified QoS ranking score to the users. However, in multi-objective context, some works [43, 68, 73, 74] on service composition problems are only approached by semi-automated methods to handle the conflicting QoS attributes independently, where the workflow structure is assumed to be pre-existing. Meanwhile, constraints on SLA are also employed to some of these approaches to reach the solutions with desirable level. These constrains raised the complexity of absolute Preto priority relation [26]. From above discussion, these is a lack of fully automated approaches to multi-objective web service composition problems for QoS-aware web service composition abiding by constrains on SLA. Moreover, the insufficiency of handling only non-functional attributes (i.e., QoS) has given rise to adding semantic matchmaking quality into simultaneous consideration.

Dynamic Semantic Web Service Composition

In a dynamic environment, QoS of the atomic services in service repository is fluctuating over time. Static service composition solution is no longer enough, and requisite actions must be taken if the original composition solution changes in QoS or is not be executable due to any service involved goes offline. Apart from that, newly registered services could also could impact composition plan as it could significantly contribute to the overall QoS or quality of semantic matchmaking. Therefore, dynamic web service composition is proposed to effectively and efficiently monitor and update composition solutions when they are outdated [42].

The major techniques endeavoured to update outdated or incorrect compositions allow for dynamic adaptation of the solutions based on implementing variability constructs at the language level. This approach is difficult to manage, and error-prone. Based on and by extending the previous approaches, variability model [4] is proposed to support the adaption. However, most of the EC-based approaches to web service composition have been studied in static scenarios, rather than dynamic ones. Although a lack of research in this field, they have been showing its confidence in its behaviour for handling dynamic web service composition for two reasons as below: a proper amount of population stored could be used for retrieving an alternative composition solution in the case of failure. Also, The stored population could be further evolved while taking changes of QoS into account. This "self-heal" process supports the adaption of a dynamic environment. Presented those benefits above, it is very advisable to study the effectiveness of EC approaches in a dynamic composition context.

Automated Web Service Composition Based on Preconditions and Effects

Apart from considering the satisfactory inputs and production of outputs, some conditional constraints also determine the execution of services. These conditional constraints lead to

multiple possible paths for execution when services are composed together, since inputs and outputs are not everything required for service execution. E.g., In the scenario of an online book shopping system adapted from [70], services are composed to provide an operation for book shopping. Users expect purchasing outcome (e.g. receipt) returned If book and customer details (e.g. title, author, customer id) are given. In this case, the users may have specific constraints. If the customer has enough money to pay for the book in full amount, then they would like to do so. Otherwise, the customer would like to pay by instalments. Therefore, the constraints are on their current account balance needs to be handled during the execution of the service composition.

Most of the approaches to automated web service composition are approached through services represented by only inputs and outputs, which are simply utilised to achieve service composition. However, the underlying functional knowledge of services (i.e., prerequisites for execution, and result in some changes, often know as precondition and effects) is not included [53]. On the one hand, many promising approaches [11] been explored to achieve compositions that consider precondition and effects using AI planning, since AI planning ensures both the correctness of functionality and satisfactory of constrains. Meanwhile, Exhaustive methods are always utilised with AI planning for optimisation problem in web service composition based on preconditions and effects. These methods always present less efficiency, poor scalability, and intensive computation. On the other hand, EC techniques (i.e., heuristic methods) are considered to be more flexible and more efficient. Given the benefits from both AI planning and EC-based techniques, they are motivated to be fully explored for automated web service composition based on precondition and effects.

1.3 Research Goals

The overall goal of this thesis is to develop new and effective hybrid approaches to comprehensive quality-aware automated semantic web service composition. More specifically, the focus will be on developing hybrid approaches that could explicitly support a proposed comprehensive quality for jointly optimising QoS and semantic matchmaking quality using single-objective method, developing multi-objective approaches for optimising the quality criteria that involved in the decision making of composition solutions selection, and developing hybrid composition techniques to dynamic service composition for handling changes of composition environment, and developing approaches for semantic web service composition, particularly, considering precondition and effects. This research aim to develop a hybridisation of various composition techniques for effectively handling the several service composition problems discussed above. The research goal described above can be achieved by completing the following set of objectives:

1. **Hybrid approaches to comprehensive quality-aware automated web service composition that simultaneously optimises both QoS and semantic matchmaking quality**. Particularly, we extend existing works on QoS-aware service composition by considering jointly optimising the both quality aspects, which is proposed as a comprehensive quality model. On the other hand, representations of the composition solutions are the key aspect of the approaches, and they must maintain all the required information for the evaluation. Therefore, we will investigate the following sub-objectives to handle this objective.

   (a) *Propose a comprehensive quality model that addresses QoS and semantic matchmaking quality simultaneously with a desirable balance on both sides.* We aim to establish a quality model with a simple calculation and good performance for the evaluation

of our proposed comprehensive quality. Meanwhile, to enable a better evaluation on our approaches, it must support most of existing benchmark datasets, e.g., Web Service Challenge 2009 (WSC09)[32] and OWLS-TC [35].

(b) *Propose direct and indirect solution representations for comprehensive quality-aware web service composition.* Graph-based and tree-based representations are widely used for directly representing service composition solutions. Graph-based representations are capable of presenting all the semantic matchmaking relationships as edges, but hardly presenting some composition constructs (e.g. loop and choice). Tree-based representations could be more ideal for practical use, since they can present all composition constructs as inner nodes of trees. However, they could hardly maintain all the edge-related relationships supported by graphs. To take advantage of the benefits from both graph-based and tree-based representations, we aim to propose a tree-like representation representation. The *indirect representations* do not present the final service composition solutions, they must be decode to executable service composition. Previous studies have shown their better performances in searching optimal solution for QoS-aware web service composition [18, 19]. However, the decoding process could increase the computation time. Apart from that, the indirect representation potentially increases the searching space, due to the changes of the indirect representation may result in the same solutions. To overcome these disadvantages, it is advisable to propose more efficient indirect representations.

(c) *Propose hybrid methods to effectively and efficiently handle the problem for comprehensive quality-aware automated web service composition.* The reasons of utilising hybrid techniques are briefly discussed in the first motivation. Herein, hybrid approaches are suggested to be developed for supporting both the proposed indirect and indirect representations, as well as the comprehensive quality model. In particularly, we aim to propose hybrid heuristics strategies to provide fast convergence of fitness value and avoid being trapped by the local optimal.

2. **Develop multi-objective approaches to optimising the comprehensive quality of service composition**. In practice, many quality criteria proposed in our comprehensive quality are often simultaneously desired and normally conflicting. Existing works [13, 72, 74, 43, 75, 80] mainly concentrate on semi-automated QoS-aware web service composition. Therefore, a study needs to be carried out for not only a better understanding the trade-offs between different objectives (e.g., quality of semantic matchmaking and QoS are naturally considered as two conflicting objectives), but also a good investigation on fully automated approaches utilising those algorithms (e.g, NSGA-II [21], SPEA2 [83] and MOEA/D [79]). These algorithms are needed for finding a Pareto front of evolved solutions that satisfies users' interests. Meanwhile, different representations may not perform equally effective, so a study on improving the performances of different representations with different fully automated approaches also arouses researchers' interest. Apart from that, SLA consideration needs to be taken into account, as well as customised matchmaking levels that needs to be proposed to bring the flexibility in meeting different requirements of segmented users. The development of this approach can be divided into the following three sub-objectives:

   (a) *Develop a fundamental EC-based approaches to multi-objective fully automated comprehensive quality semantic web service composition*.
   Here we develop a fundamental muti-objective optimisation approach using some multi-objective EC-based algorithm. (e.g, NSGA-II [21], SPEA2 [83] and MOEA/D

[79]), where different representations and modified multi-objective EC algorithms are simultaneously taken into account for studying both their effectiveness and efficiency. This sub-objective is also established for mainly studying each independent quality criteria from our proposed comprehensive quality model in Objective 1. In particular, both quality of semantic matchmaking and QoS must be optimised independently, since they may represent conflicting interests. It would be interesting to examine different tradeoffs among the service composition solutions with respect to the different quality criterion. Apart from that, fully automated approaches are also developed to overcome the limitation (i.e., semi-automated approaches) in existing works discussed above.

(b) *Develop hybrid approaches to multi-objective fully automated comprehensive quality semantic web service composition*. Since we have achieved the sub-objective 2a, the effectiveness and efficiency are the next focus. The fundamental approaches should be extended by introducing some local search. In particular, some efforts could be made for simultaneously considering the improvements on representations themselves and the combinations with a fast local search. For example, an exhaustive search for the neighbourhood of best the individual within the current population is performed with a relatively higher priority for service selection.

(c) *Develop hybrid approaches to multi-objective fully automated comprehensive quality semantic web service composition abiding by constraints on SLA and customised matchmaking level*. In the real world scenario, satisfactory on given SLA constraints are required to be meet other than only optimising QoS. Therefore, this sub-objective should be further extended to considering some additional constraints on QoS (i.e., multilevel constraints with lower and upper bound for different individual QoS criterion [74]). Meanwhile, to satisfy the customised different semantic matchmaking levels (e.g., exact matchmaking level and less strict matchmaking level), extensive methods are also required to copy with the constraints on the different accepted matchmaking level.

3. **Develop hybrid techniques to handle dynamic semantic web service composition effectively**. The objectives 1 and 2 are proposed assuming the environment of service composition is static. In our context, environments refer to the registered services in the service repository and their non-functional attributes remain constant, as well as the ontology utilised for describing the resources of web services. Since existing approaches are only executed once to generate a composition plan from a given composition task, some factors could significantly impact the generated solution. E.g., QoS values of services being composed of are fluctuating over time, service chosen at the planning stage may not available to be invoked at the runtime, or newly registered service may need to be considered for reconstructing a better plan. To effectively handle these changes, three studies are performed as three sub-objectives as follows:

   (a) *Develop fundamental techniques to re-optimise solution candidates for changes in QoS and Ontology.* Traditionally, initial population is created with solution candidates that are further evolved for searching optimal solutions. During the evolutionary computation process, most of service candidates are discarded excluding the best service candidate found. Those discarded solutions candidates may be promising, since some of them could turn to be alternative best due to the changes in services or ontology. Therefore, instead of discarding the solution candidates, it would be very motivated to keep the discard solution candidates where both diversity and elitism are preserved. We aim to propose an effective approach to

re-optimise these maintained candidates for further use.

(b) *Develop hybrid techniques to re-optimise solution candidates for changes in QoS and Ontology.* Once the fundamental techniques to re-optimise solution candidates for changes in QoS and Ontology are achieved, it should be further studied in developing more efficient and effective approach to better handling this problem. We aim to propose an adaptive and hybrid approach to this dynamic problem. Our initial idea is to assign a higher priority to a group of services with changes and a lower priority to a group of services without changes, respectively, for considering of service selection based on a local search during the evolutionary process. The higher priority must be adaptively handled with a proper decreasing rate with respect to each service in the first group.

(c) *Develop hybrid techniques for handling service failure and newly registration using updated candidates in the population.* Apart from the changes in the QoS and Ontology. A dynamic web service composition also tackle the issue of occasionally service fail or new service registration. For the case of service fail, some methods must be proposed to either handle those un-invokable services for replacement. We aim to propose some approaches using direct representations, where we could either efficiently mutate the solutions candidates partially on un-invokable atomic services, or its involved parent composition components, or effectively re-generate whole solutions using invokable service in the service repository. For the case of new service registration, giving a priority for newly registered services should be properly considered for selecting services. We could discard parts (e.g 50 percetage) of the current population, and then top up the current population from updated services repository.

4. **Develop hybrid approaches to semantic web service compositions based on preconditions and effects. (Optional)** Particularly, we extend most service composition approaches (i.e., satisfactory on inputs and outputs) to include preconditions and effects. These conditional constrains also raise the study of various of composition constructs in the field of automated semantic web service composition, e.g., loop and choice. Therefore, two sub-objectives are proposed to target as follow.

   (a) *Develop fundamental techniques to semantic web service composition based on preconditions and effects*. In the problem stated above, inputs and outputs are everything of web services for handling some web services. Remodelling service composition problem by additionally considering the preconditions and effects are the initial tasks required to be completed. In particular, we need to establish a fundamental mechanism of satisfactory on preconditions and effects involved in the consideration of only sequence and parallel composition constructs for automated semantic web service composition problem. We aim to develop fundamental approaches to effectively this problem. In particular, representations are needed to be proposed for coping with the newly modelled problem.

   (b) *Develop hybrid techniques to semantic web service composition based on preconditions and effects*. Once the fundamental techniques to semantic web service composition based on preconditions and effects are proposed. It followed that more effective and efficient works. In particular, we aim to utilise the motivated hybridisation of various techniques for improving the performances of existing proposed methods.

   (c) *Develop hybrid techniques to semantic web service composition based on preconditions and effects for supporting loops and choice*. We initially extend the fundamental

mechanism of satisfactory on preconditions and effects to include loops and choice. To extensively cope with these two constructs, new effective representations are required to be studied. Apart from that, hybrid approaches are aimed to developed to efficiently solving fully automated comprehensive quality-aware semantic web service composition problem.

1.4 Published Papers

During the initial stage of this research, the preliminary work was carried out on establishing the comprehensive quality model. Afterwards, some studies on the direct and indirect representations are completed for the first objective of this research, but the earlier works focus on static web service composition using single-objective optimisation technique, the results show indirect representation shows its promise in solving our service composition problem. The following are the publications made from the preliminary studies:

- Wang, C., Ma, H., Chen, A., Hartmann, S.: Comprehensive quality-aware automated semantic web service composition. In: Australasian Joint Conference on Artificial Intelligence (To appear)

1.5 Organisation of Proposal

The remainder of the proposal is organised as follows: Chapter 2 provides a fundamental definition of the web service composition problem and performs a literature review covering a range of works in this field; Chapter **??** discusses the preliminary work explores direct and indirect representations for comprehensive quality-aware semantic web service composition using the hybridisation of AI planning techniques and EC-based techniques; Chapter **??** presents a plan detailing this project's intended contributions, a project timeline, and a thesis outline.

Chapter 2

Literature Review

In this chapter, we first introduce the background of web service composition in Section 2.2. Followed that Section **??** discuss the single-objective service composition for both EC and non-EC based approaches. Section **??** reviews existing works in muti-objective approaches and many-objective approaches. Dynamic web service composition is covered in Section **??**. Section **??** discussed most AI planning-based approaches to semantic service composition. Lastly, Section **??** summarises some critical points discussed in this chapter as well as some inadequacies or limitations in the literature review.

2.1 Web Service

Web services are self-describing modules offering functionalities over the internet. The functionalities of web services are often specified by their functional attributes, which satisfy users' functional requirements and provide mechanisms to allow users to search desired service. Web services are classified into two groups based on their functionalities: *information-providing services* and *world-altering services* [48]. The first type of services expect some data returned by giving inputs or nothing. For example, a service for air velocity transducer reads the wind speed and return the velocity at the time. This service does not require any inputs. On the other side, a service for city weather requires given city name to return the weather information for that specific city. Information-providing services do not produce any side effect to the world. The functionalities of these services are only inputs and outputs. The second type of services not only provide data information but also alters the status of the world by producing side effects. For example, a PayPal service will cause a deduction in the balance of users' bank account. *In this proposal, we mainly focus the first type of services for first three objective. Later on, an extensive study is optimally carried on the second type of services*

In realistic scenarios, the non-functional are also important. For example, users may not prefer a service with higher cost with the same functionality provided by another one. As demonstrated above, the functional attributes determine what service really does, while the non-functional attributes often refers to some quality criteria, which is considered for raking services [3]. We first explain web services using a formal model from [2] below, where both the functional and non-functional attributes are captured uniformly. Further, A Labelled Transition System [3] is addressed with its abstract and updated model for demonstrating the behaviours of web services in Subsection 2.1.1, which emphasises on side of the functional attributes. After demonstrating these models, we will discuss about the nonfunctional properties of web services in Subsection 2.1.2.

A Formal Model of Web Service. Given a finite set $\wp$ of property types of web services and a finite set ϑ of values sets. Each property type is associated with a value set. We view a

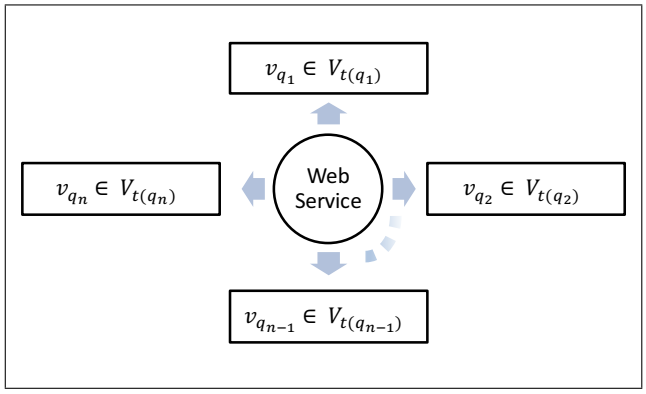


Figure 2.1: Property-Based Web Service Formal Model [2].

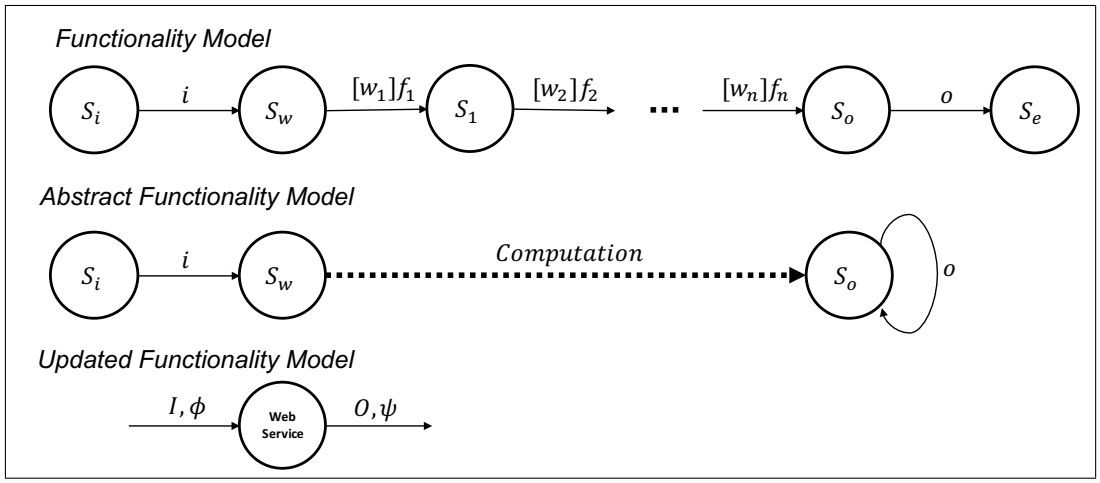


Figure 2.2: The functionality of a Web Service [2].

Web service as a finite set Q of property instances with each property instance $q \in Q$ being of a property type$t(q) \in \wp$ that is associated with a value $v_q \in \vartheta_{t(q)}$. See Fig. 2.1

2.1.1 Functional Properties of Web services

A Functionality Model. This mode is represented as a Labelled Transition System (LTS): $L = (S, T, \rightarrow)$ comprises a set S of states, a set T of transition labels and a labeled transition relation $\rightarrow \subseteq S \times T \times S$. This transition system is established to present the actual behaviour of web services, which consists of a series of states. See Fig. 2.2.

- S_i start state includes knowledge available to web service before the web service is invoked by inputs;
- S_w state includes the inputs additional to the knowledge in 1;
- A series of state S_1 to S_n proceed with corresponding actions f_n that only occurred if each related condition $[w_i]$ is approved to be true.
- S_o state contains all the outputs and all the changes performed in the knowledge base.
- S_e is the end state, which is equivalent to S_o, since the knowledge base is not changed.

An Abstract Functionality Model. The first functionality model presented above is not completely demonstrated for the internal actions, as the services provider does not want reveal all the internal functions, and it is not feasible to list a global set of property name. Therefore, an abstract functionality of a web service is modelled by eliminating all the intermediate properties. In the abstract model of a web service, the functional properties of the web service could be identified as inputs i, pre-state S_i, outputs o and post-state S_o. These four properties are mapped to a set of input I, preconditions ϕ, a set of outputs O and postcondition φ respectively in third updated functionality model demonstrated below.

A Updated functionality Model. In the updated model, a set of inputs I is required by a service and a set of output O is returned after the successful execution. Apart from that, the precondition ϕ must be hold in the knowledge base before service is invoked by passing the input I. To enable the interoperability of the functional properties, ontology reasoner is employed to reason about the properties of web services. To distinguish the changes between before and after service execution, these changes are modelled as property instances. For example, inputs and outputs assigned as variable names and further referenced in preconditions and postconditions, which can be distinguished as different instances from the knowledge base. In particular, preconditions is assigned to the description of requirements of inputs using logic formula. Herein the description of the formula considers the following cases that are demonstrated using Planning Domain Description Language (PDDL [25]):

- Conditions on the type of inputs, e.g., the payment of an online shopping website is made by Visa or Cash: $\phi : (Format(payment) = Visa \cup Format(payment) = Cash)$.
- Relationships among inputs, e.g., authoried users are required for an online shopping: $\phi : (Authoried?Useraccount)$.
- Conditions on the value of inputs, e.g., saving account balance has more than 100 dollars: $\phi : (\geq (amounts, savingaccount), 100)$

These preconditions must be hold in the state consistently while inputs are being passed to services. Similarly, the postcondition is restricted to the description of constraints on returned outputs, relationships between inputs and outputs, and changes caused by the service in the knowledge bases.

2.1.2 Nonfunctional Properties of Web services

Apart from the functional properties of web services discussed above, the non-functional properties of web services play an important part in composing services. For example, customers prefer lowest execution cost with highest response time and reliability. According to [78], four most often considered QoS parameters are 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.

2.1.3 Web Service Discovery

To generate service compositions, web service must provide mechanisms for discovery required services. Therefore, service discovery must be one fundamental technique to be considered in all service composition approaches. [1] discussed three mechanisms of semantic web service discovery: classification-based approach, functionality-based approach and hybrid approach. Those service discovery techniques are further demonstrated below.

The first service discovery technique makes use of the classes provided by service semantic annotation in WSMO-Lite language. Therefore, service requesters can use class names to express a goal offering a straightforward discovery from a set of classes. However, classes without clear meaning definition could lead to the issue of incomprehensibility of web service discovery. For example, several classes may declared in either different terms for the describing the same content or same terms for describing different content.

The second service discovery technique does not take classes into account, but consider functional properties of web service to include pre-conditions and post-conditions. In particular, a desired functionality description is defined. A discovery algorithm must be developed to handle a matching for different input, output, precondition and postcondition with associated concepts and relations in the provided domain knowledge base. The key idea of the matchmaking is to check whether services accept all the desired inputs provided bu users and whether the desired outputs is delivered by services. In addition, the matchmaking algorithm also checks for the satisfiability of implications that actual precondition and actual postcondition must imply the desired precondition and desired postcondition respectively. The strength of the second is that it potentially meet the demands of all the comprehensible discovery, while the weakness is a lack of efficiency and scalability.

The third service discovery technique is based on a hybridisation of classification and functionality-based discovery. Classification hierarchy is proposed to achieve automatic semantic reasoning in hierarchical functionality. For example, a functionality class is associated with super classes and sub classes for more general functionality class and more specific functionality class respectively. However, to make a consistency of classification hierarchy, the inputs, outputs, precondition and postcondition of a functionality class must satisfy the conditions that contains all the inputs, outputs, precondition and postcondition of all the classes it is subsumed by. The advantage of this approach is to achieve better performance combining strengths of the previous two pure classifications based and functionality based approaches. While the classification hierarchy needs to be kept consistent when a new web service is published or updated.

The first and third approach is considered either less effective or demands to build up a consistent ontology for classes and their functional attributes. That is not the focus of our research. *In the proposal, we use the second service discovery technique for meeting the comprehensible discovery. That is, different types of ontology reasoning are utilised to approach the matchmaking as a fundamental part of service composition algorithm.*

2.2 Web Service Composition

Since one atomic web service could not satisfy or fully satisfy users' complex requirements, web service composition is approached by composing web services together with more sophisticated functionalities to meet the demand. Due to fully human intervention in manual service composition, so it is very time-consuming and less productive. Therefore, many approaches have been developed to achieve semi-automated or fully automated service composition. The semi-automated service composition is inspired by the business process that required prior knowledge to build up the abstract workflow. This workflow can be decom-

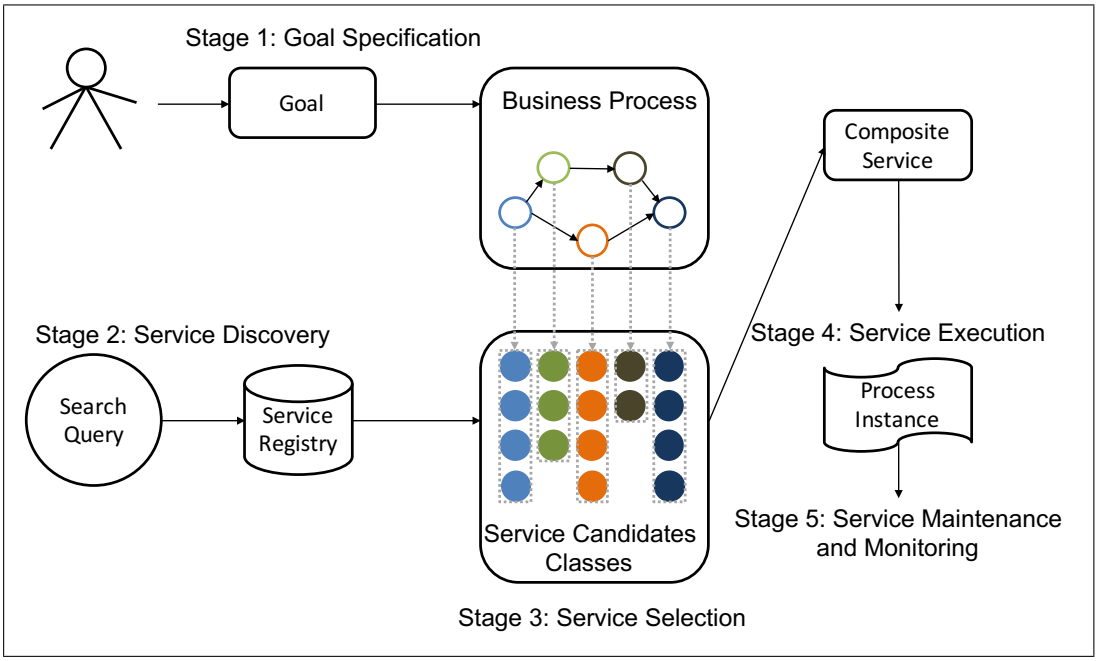


Figure 2.3: Web service composition lifecycle [50].

posed into several functional services slots for proper services being fitted. These steps are further discussed in Subsection 2.2.1. On the other hand, when we are composing services, both semi-automated and fully automated web service composition holds the challenge in the interoperability of services discussed previously. In particular, several problems are simultaneously taken into account. That is I/O matchmaking (i.e., the mechanisms of services for ensuring the interoperability), discovery relevant services to the optimising the quality of service composition, e.g., overall QoS. Consequently, the following concerns are required to consider in generating composition solution.

2.2.1 Web Service Composition Lifecycle

Typical steps in a workflow-based automated Web service composition solution are shown in Fig. 2.3. The detail of the service lifecycle is discussed as follows:

1. *Goal specification:* The first step in service composition is to collect users' requirements for composition goal that comprises of the functional (i.e., correct data flow) and non-functional side (i.e., QoS). This step is achieved by building up an abstract workflow including a series of tasks with clearly defined functionality. Those tasks could be completed by selecting proper concrete services to reach desired QoS.

2. *Service discovery:* Once the goal is clearly specified, concrete web services are to be selected for each task regarding its functional requirement. Often, more than one concrete web service is likely to be found to match the one task. However, those matched web services are always different in QoS. Therefore, web services are classified by the functionality of each task, i.e., inputs and outputs.

3. *Service selection:* At this stage, many techniques have been studied to select web services to best match each task for the satisfying functional requirement of each task and overall business process. Therefore, a plan of service composition is created ahead of execution.

4. *Service execution:* the process instance is monitored for any changes or services failures during service execution. In this stage, some actions are to be taken for adapting the changes.

The web service lifecycle discussed above is a typical *semi-automated approach*. There is a distinction between semi-automated and fully automated approaches. On the one hand, during the goal specification stage of semi-automated approach, the abstract work is already provided. On the other hand, an abstract workflow is not provided at the stage of goal specification for *fully automated service composition*. Often, fully automated service composition rely on some algorithms (e.g., Graphplan algorithm [8]) to achieve service composition, during which service workflow is gradually built up along with the service discovery and service selection at the same time. In this proposal, I concentrate on developing methods for fully automated service composition since it has been shown the flexibility. Also, herein service discovery and service selection are considered as interrelated tasks that are interleaved with the composition algorithm.

2.2.2 Functional Properties of Web Service Composition

Substantial work [6, 38, 39, 41, 62] on semantic web service composition utilises Description logic (DL) reasoning between input and output parameters of web services for matchmaking. OWL and OWL-S are the most common semantic specifications used currently [59], and they enable automatic selection, composition, and interoperation of Web services to implement complicated composition tasks [47]. However, some matchmaking types (discussed below) may penalise the matchmaking quality and are less preferred by users. Therefore, exploring a effective mechanism for measuring the quality of semantic matchmaking in service composition is a very demanding research area.

Given two concepts a, b in ontology $\mathcal{O}$, four commonly used *Matchmaking types* are often used to describe the level of a match between outputs and inputs [54]:

- the *matchmaking* returns *exact* if a and b are equivalent ($a \equiv b$),
- the *matchmaking* returns $plugin$ if a is a sub-concept of b ($a \sqsubseteq b$),
- the *matchmaking* returns *subsume* if a is a super-concept of b ($a \sqsupseteq b$),
- the *matchmaking* returns $fail$ if none of previous matchmaking types is returned.

Often, the similarity of two instances of two knowledge representations encoded in the same ontology is also utilised to measure the quality of matchmaking regarding the four matchmaking types discussed above. The work [39] additionally consider *interaction* matchmaking type ($a \sqcap b$), i.e., if the intersection of a and b is satisfiable. In their work, a causal link link $sl_{i,j} \doteq \langle S_i, Sim_T(Out\_s_i, In\_s_j), S_j \rangle$ is created between two functional property instances for a input and a output. In particular, both *exact* match and $plugin$ match are presented as robust causal links, while both *subsume* match and *intersection* match are presented as valid casual links. However, valid casual links are not specific enough to be utilised as the input of another web service. Thus the output requires Extra Description Equation 2.1 to enable proper service composition. As a result, Subsume and Intersection matching type is transferred to be Exact and PlugIn respectively to formulate a robust link.

$$In\_s_x \setminus Out\_s_y \doteq \min_{\preceq d}\{B | B \sqcap Out\_s_y \equiv In\_s_x\}, since\ Out\_s_y \sqsupseteq In\_s_x \tag{2.1}$$

In this paper we are only interested in robust compositions where only exact and $plugin$ matches are considered, see [38]. As argued in [38] $plugin$ matches are less preferable than exact matches

due to the overheads associated with data processing. We suggest to consider the semantic similarity of concepts when comparing different plugin matches. Herein we demonstrate an example of a robust causal link by between two matched services S and S', noted as $S \rightarrow S'$, if an output a ($a \in O_S$) of S serves as the input b ($b \in O_{S'}$) of S' satisfying either $a \equiv b$ or $a \sqsubseteq b$. For concepts a, b in $\mathcal{O}$ the *semantic similarity* $sim(a, b)$ is calculated based on the edge counting method in a taxonomy like WorldNet or Ontology [66]. This method has the advantages of simple calculation and good performance [66]. Therefore, the *matchmaking type* and *semantic similarity* of a robust causal link can be defined as follow:

$$type_{link} = \begin{cases} 1 & \text{if } a \equiv b \ (exact \text{ match}) \\ p & \text{if } a \sqsubseteq b \ (plugin \text{ match}) \end{cases}, \qquad sim_{link} = sim(a, b) = \frac{2N_c}{N_a + N_b} \tag{2.2}$$

with a suitable parameter $p, 0 < p < 1$, and with N_a, N_b and N_c, which measure the distances from concept a, concept b, and the closest common ancestor c of a and b to the top concept of the ontology $\mathcal{O}$, respectively. However, if more than one pair of matched output and input exist from service S to service S', $type_e$ and sim_e will take on their average values.

The *semantic matchmaking quality* of the service composition can be obtained by aggregating over all robust causal links as follow:

$$MT = \prod_{j=1}^{m} type_{link_j}, \qquad SIM = \frac{1}{m} \sum_{j=1}^{m} sim_{link_j} \tag{2.3}$$

2.2.3 Nonfunctional Properties of Web Service Composition

The nonfunctional properties of web service composition is determined by all the QoS of involved concrete web services in the solution. The aggregation value of QoS attributes for web services composition varies with respect to different constructs, which reflects how services associated with each other in a service composition [78].

We use formal expressions as in [44] to represent service compositions. We use the constructors $\bullet$, $\|$, $+$ and $*$ to denote sequential composition, parallel composition, choice, and iteration, respectively. The set of *composite service expressions* is the smallest collection $\mathcal{SC}$ that contains all atomic services and that is closed under sequential composition, parallel composition, choice, and iteration. That is, whenever $C_0, C_1, \ldots, C_d$ are in $\mathcal{SC}$ then $\bullet(C_1, \ldots, C_d)$, $\| \ (C_1, \ldots, C_d)$, $+(C_1, \ldots, C_d)$, and $*C_0$ are in $\mathcal{SC}$, too. Let C be a composite service expression. If C denotes an atomic service S then its QoS is given by QoS_S. Otherwise the QoS for C can be obtained inductively as summarized in Table 2.1. Herein, $p_1, \ldots, p_d$ with $\sum_{k=1}^{d} p_k = 1$ denote the probabilities of the different options of the choice $+$, while ℓ denotes the average number of iterations.

2.3 Evolutionary Computation Techniques Overview

Evolution Computing (EC) techniques are founded based on the principles of Darwin natural selection. The nature evolution and selection of individual in a population are automated simulated in EC. In particular, a population of individuals is initialised for directly or indirectly presenting the solutions. Those individual candidates are evolved and evaluated using a fitness function to evaluate the degree of how good (or bad) of each individual. Therefore, it is possible to reach solution with near-optimal fitness. EC have been shown its promise in solving combinatorial optimisation problems [5]. This is due to its flexibility

Table 2.1: QoS calculation for a composite service expression C

$C =$	$r_C =$	$a_C =$	$c_C =$	$t_C =$
$\bullet(C_1, \ldots, C_d)$	$\prod_{k=1}^{d} r_{C_k}$	$\prod_{k=1}^{d} a_{C_k}$	$\sum_{k=1}^{d} c_{C_k}$	$\sum_{k=1}^{d} t_{C_k}$
$\parallel (C_1, \ldots, C_d)$	$\prod_{k=1}^{d} r_{C_k}$	$\prod_{k=1}^{d} a_{C_k}$	$\sum_{k=1}^{d} c_{C_k}$	$MAX\{t_{C_k} \mid k \in \{1, ..., d\}\}$
$+(C_1, \ldots, C_d)$	$\prod_{k=1}^{d} p_k \cdot r_{C_k}$	$\prod_{k=1}^{d} p_k \cdot a_{C_k}$	$\sum_{k=1}^{d} p_k \cdot c_{C_k}$	$\sum_{k=1}^{d} p_k \cdot t_{C_k}$
$*C_0$	${r_{C_0}}^{\ell}$	${a_{C_0}}^{\ell}$	$\ell \cdot c_{C_0}$	$\ell \cdot t_{C_0}$

in encoding the problems for the representation and its good performance in many scenarios. In particular, To manage the constrainsts in the problems, five main methods have been proposed deal with the constraints: coding, penalty functions, repair algorithm, indirect methods of representation and multiobjective optimisation [24]. In the context of service composition. Many EC techniques have been approached for handling optimisation problems for web service composition, such as Genetic Algorithms (GA) [71], Genetic Programming (GP) [34], and Particle Swarm Optimisation (PSO) [30]. These techniques are briefly introduced here.

GP is considered as a particular application of GA with a set of different encoded genes. In particular, the representation of GA is commonly represented as a linear structure. However, In GP, each individual is commonly represented as a tree structure. the tree structure has a terminal set and a function set, where variables, constants and functions are consisted of respectively. Also, the tree structure is considered be efficiently evaluated recursively. Three genetic operators consisting of reproduction, crossover, and mutation are involved in to generate next generation for both GA and GP. Reproduction operator retains the elite individual without any changes. Crossover operator replaces one node of one individual with another node of another individual. Mutation operator replaces a randomly selected node in an individual. The whole evaluation process won't stop unless an optimised solution found or a pre-defined number of generation reached.

PSO is one of swarm intelligence (SI) that based on the behaviour of decentralized, self organised system. PSO algorithm is initialised by a group of random particles, which direct or indirect present the solutions. Those particles explores for the optimisation position, which is approached by repeating the process of transferring particles position according to both their own best-known position and global best position.

2.4 Single-Objective Web Service Composition Approaches

2.4.1 Non-EC Based Approaches

2.4.2 EC-based Composition Approaches

2.5 Multi-objective, and Many-objective Composition Approaches

Maximising or minimising a single objective function is a most commonly used way to handle optimising problems in automated web service composition. That is a Simple Additive Weighting (SAW) [28] technique, which presents a utility function for all the individual quality criteria as a whole. This technique optimises and ranks each web service composition using a single value for each solution. However, the limitation of this technique lies in

not handing the conflicting quality criteria. Those conflicting quality criteria are always presented trade-offs. To overcome this limitation, a set of objectives corresponding to different independent quality criteria are optimised independently. Consequently, a set of promising solutions that present many quality criteria trade-offs are returned.

2.5.1 Multi-objective approaches

Many multi-objective techniques [43, 80, 75, 74, 72, 13] have been investigated to extensively study QoS-aware web service composition problems. A set of optimised solutions is ranked based on a set of independent objectives, i.e., different QoS attributes. In particular, solutions are compared according to their relationship for domination. Especially, figuring out solutions are clearly dominating the others. For example, given two service composition solutions that are compared based on execution cost c and execution time t, solution one, $wsc_1(c = 10, t = 1)$ and solution two, $wsc_2(c = 13, t = 1)$. In our context, wsc_1 dominate wsc_2 as wsc_1 has the same execution time and a lower execution cost. If given $wsc_3(c = 10, t = 2)$, wsc_2 is a *non-dominant* solution in the relation to wsc_3 because of its longer execution time and cheaper execution cost. Therefore, If those non-dominant solutions are globally produced among both the dominant and non-dominant solutions, i.e., they do not dominate themselves. These solutions are called a *Pareto front*, which provide a set of non-dominant solutions for users to choose.

Multi-objective techniques with GA

Many approaches to multi-objective Web service employs GA [43], but other EC algorithms are also considered. For GA, [43] employs a service composition model, called MCOOP (i.e., muti-constraint and multi-objective optimal path) as web service composition solution for only a sequence service composition considered in the paper. In this model, different paths are selected from a service composition graph that includes N service group. In each group, services present same functionality with different QoS. Apart from that, GPDSS is proposed to generate the outputs of Pareto optimal composition paths. In particular, two points crossover and mutation are applied to speed up the astringency of this algorithm. The work [68] investigates a semi-automated approach to SLA-aware web service composition problem. Each linear representation proposed here presents three service composition solutions designed for three group users' categories. The individuals are randomly initialised, evaluated and optimised with objectives from all the possible combinations of throughput, latency, cost and user category. In this work, two multi-objective genetic algorithms: E-MOGA and Extreme-E are developed. E-MOGA is proposed to search a set of solutions that equally distributed in the searching space by the means of fitness function, where the production of domination value, Manhattan distance to the worst point and sparsity (i.e, Manhattan distance to the closest neighbour individual) is assigned to the feasible individual as fitness value, and SLA violation /domination value is assigned to the infeasible solutions. On the other hand, Extrem-E provide extreme solutions by employing fitness functional, where weights use a term 1/exp(p-1), where p is the number of objectives and is assigned to the p^{th} objective.

Multi-objective techniques with PSO

The work [74] combines genetic operators and particle swarm optimisation algorithm together to tackle the multi-objective SLA-aware web service composition problems. The method proposed in the paper is considered to be more effective in considering different scare of cases. It is called as HMDPSO, i.e., hybrid multi-objective discrete particle swarm

optimisation. In particular, the updates of particle's velocity and position are achieved by the crossover operator, where both velocity and position of new individual are updated in accordance with positions of *pbest*, *gbest*, and current velocity. On the other hand, mutation strategy is introduced to increase the diversity of particle and is performed on the *gbest* particle if the proposed swarm diversity indicator is below some value. For the evaluation, the fitness values of individuals are assigned in the same way as the E-MOGA method in [68].

Multi-objective techniques with ACO

Generally, ACO simulates foraging behaviours of a group of ants for optimising the traversed foraging path, where the strength of pheromones is taken account for. The work [80] turns the service composition problem into path selection problem for the given abstract workflow with different service candidate set. It employs a different strategy of "divide and conquer" for decomposing a given workflow. That is, two or more abstract execution paths are decomposed from the workflow and have no overlapped abstract services. This decomposing strategy results in a much smaller length of the execution paths compared to those in the works [**?**]. Also, a new ACO algorithm is proposed to handle the multi-objective QoS-aware service composition problem. In particular, the phenomenon is presented as a k-tuple for k objectives, rather than a single value. Apart from that, a different phenomenon updating rule is proposed by considering an employment of a proposed utility function as a global criteron. The paper [69] introduces nonfunctional attributes of web services to include trust degree according to the execution log. Also, a novel adaptive ant colony optimisation algorithm is proposed to overcome the slow convergence presented from the traditional ant colony optimisation algorithmd. In particular, the pheromone strength coefficient is adjusted dynamically to control both the updating and evaporation of pheromone. The experiment results are analysed in an alternative way. That is, the total Pareto solutions are combined from different compared ACO algorithms, then the accurate rate of each algorithm is calculated based on the compared Pareto solutions identified in the total Pareto solutions. The results also show more Pareto solutions found compared to the traditional ACO methods. However, the experiment is only conducted for the evaluation of a small case study, where only a simple abstract workflow is studied.

2.5.2 Many-objective approaches

Herein, more than three objectives in Multi-objective problems (MOPs) are often considered as many-objective problems. Ishibuchi et al. [29] present an analysis of the multi-objective algorithm for handling optimisation problems with more than 3 objectives. However, they address that the searching ability is deteriorating while the number of objectives is increasing, since the non-dominated solution is very large, which make it harder to move solutions towards the Pareto Front.

The work [20] employs NSGA-II to deal with five different quality criteria (i.e., runtime, price, reputation, availability and reliability) for semi-automated web service composition problem. To examine the techniques to decrease the deterioration, two preference relations proposed by [7] are applied to NSGA-II: Maximum Ranking (MR) and Average Ranking methods (AR). In particular, MR is the best of all the ranking scores from all the objectives, and AR is a sum of all the ranking scores from all the objectives. Therefore, three algorithms (NSGA-II, NSGA-II with MR and NSGA-II with AR) are evaluated for studying the five different performance metrics (i.e., hypervolume [81], Generational Distance [67], Spread and Coverage [82], and pseudo Pareto front (i.e, a combination of all non-dominated solutions)), where An empirical evaluation is performed on. The experiment shows NSGA-II with AR

outperforms others in both GD and Spread (i.e., more balanced solutions). However, a certain region of Pareto Front is generated by NSGA-II, rather than a wider distribution for the solutions. NSGA-II with MR performs intermediately compared to the other two algorithms. On the whole, this work, for the first time, takes two preference relations into account for solving many-objective service composition problem, and contribute to finding better solutions with many performance metrics.

Bibliography

[1] AGARWAL, S., JUNGHANS, M., FABRE, O., TOMA, I., AND LORRE, J.-P. D5. 3.1 first service discovery prototype. *Deliverable D5 3* (2009).

[2] AGARWAL, S., JUNGHANS, M., AND NORTON, B. D5. 3.2 second service discovery prototype. *Deliverable D5 3* (2009).

[3] AGARWAL, S., LAMPARTER, S., AND STUDER, R. Making web services tradable: A policy-based approach for specifying preferences on web service properties. *Web Semantics: Science, Services and Agents on the World Wide Web 7*, 1 (2009), 11–20.

[4] ALFÉREZ, G. H., PELECHANO, V., MAZO, R., SALINESI, C., AND DIAZ, D. Dynamic adaptation of service compositions with variability models. *Journal of Systems and Software 91* (2014), 24–47.

[5] BACK, T., HAMMEL, U., AND SCHWEFEL, H.-P. Evolutionary computation: Comments on the history and current state. *IEEE transactions on Evolutionary Computation 1*, 1 (1997), 3–17.

[6] BANSAL, S., BANSAL, A., GUPTA, G., AND BLAKE, M. B. Generalized semantic web service composition. *Service Oriented Computing and Applications 10*, 2 (2016), 111–133.

[7] BENTLEY, P. J., AND WAKEFIELD, J. P. Finding acceptable solutions in the pareto-optimal range using multiobjective genetic algorithms. *Soft computing in engineering design and manufacturing 5* (1997), 231–240.

[8] BLUM, A. L., AND FURST, M. L. Fast planning through planning graph analysis. *Artificial intelligence 90*, 1 (1997), 281–300.

[9] BOOTH, D., HAAS, H., MCCABE, F., NEWCOMER, E., CHAMPION, M., FERRIS, C., AND ORCHARD, D. Web services architecture. w3c working note. *W3C Working Notes* (2004).

[10] BOUSTIL, A., MAAMRI, R., AND SAHNOUN, Z. A semantic selection approach for composite web services using owl-dl and rules. *Service Oriented Computing and Applications 8*, 3 (2014), 221–238.

[11] BOUSTIL, A., MAAMRI, R., AND SAHNOUN, Z. A semantic selection approach for composite web services using OWL-DL and rules. *Service Oriented Computing and Applications 8*, 3 (2014), 221–238.

[12] CHAN, K., BISHOP, J., STEYN, J., BARESI, L., AND GUINEA, S. A fault taxonomy for web service composition. In *Service-oriented computing-ICSOC 2007 Workshops* (2009), Springer, pp. 363–375.

[13] CHEN, Y., HUANG, J., AND LIN, C. Partial selection: An efficient approach for qos-aware web service composition. In *Web Services (ICWS), 2014 IEEE International Conference on* (2014), IEEE, pp. 1–8.

[14] CURBERA, F., DUFTLER, M., KHALAF, R., NAGY, W., MUKHI, N., AND WEERAWARANA, S. Unraveling the web services web: an introduction to soap, wsdl, and uddi. *IEEE Internet computing 6*, 2 (2002), 86–93.

[15] CURBERA, F., NAGY, W., AND WEERAWARANA, S. Web services: Why and how. In *Workshop on Object-Oriented Web Services-OOPSLA* (2001), vol. 2001.

[16] DA SILVA, A., MA, H., AND ZHANG, M. Graphevol: A graph evolution technique for web service composition. In *Database and Expert Systems Applications*, vol. 9262. Springer International Publishing, 2015, pp. 134–142.

[17] DA SILVA, A. S., MA, H., AND ZHANG, M. Genetic programming for qos-aware web service composition and selection. *Soft Computing* (2016), 1–17.

[18] DA SILVA, A. S., MEI, Y., MA, H., AND ZHANG, M. A memetic algorithm-based indirect approach to web service composition. In *Evolutionary Computation (CEC), 2016 IEEE Congress on* (2016), IEEE, pp. 3385–3392.

[19] DA SILVA, A. S., MEI, Y., MA, H., AND ZHANG, M. Particle swarm optimisation with sequence-like indirect representation for web service composition. In *European Conference on Evolutionary Computation in Combinatorial Optimization* (2016), Springer, pp. 202–218.

[20] DE CAMPOS, A., POZO, A. T., VERGILIO, S. R., AND SAVEGNAGO, T. Many-objective evolutionary algorithms in the composition of web services. In *Neural Networks (SBRN), 2010 Eleventh Brazilian Symposium on* (2010), IEEE, pp. 152–157.

[21] DEB, K., PRATAP, A., AGARWAL, S., AND MEYARIVAN, T. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation 6*, 2 (2002), 182–197.

[22] FENG, Y., NGAN, L. D., AND KANAGASABAI, R. Dynamic service composition with service-dependent qos attributes. In *Web Services (ICWS), 2013 IEEE 20th International Conference on* (2013), IEEE, pp. 10–17.

[23] FENSEL, D., FACCA, F. M., SIMPERL, E., AND TOMA, I. *Semantic web services*. Springer Science & Business Media, 2011.

[24] FLEMING, P. J., AND PURSHOUSE, R. C. Evolutionary algorithms in control systems engineering: a survey. *Control engineering practice 10*, 11 (2002), 1223–1241.

[25] FOX, M., AND LONG, D. Pddl2. 1: An extension to pddl for expressing temporal planning domains. *Journal of artificial intelligence research* (2003).

[26] GAREY, M. R., AND JOHNSON, D. S. A guide to the theory of np-completeness. *WH Freemann, New York 70* (1979).

[27] GUPTA, I. K., KUMAR, J., AND 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* (2015), IEEE, pp. 1–6.

[28] HWANG, C.-L., AND YOON, K. Lecture notes in economics and mathematical systems. *Multiple Objective Decision Making, Methods and Applications: A State-of-the-Art Survey 164* (1981).

[29] ISHIBUCHI, H., TSUKAMOTO, N., AND NOJIMA, Y. Evolutionary many-objective optimization: A short review. In *IEEE congress on evolutionary computation* (2008), pp. 2419–2426.

[30] KENNEDY, J., AND EBERHART, R. Particle swarm optimization. In *IEEE International Conference on Neural Networks* (1995), IEEE, pp. 1942–1948.

[31] KIM, I. Y., AND DE WECK, O. Adaptive weighted sum method for multiobjective optimization: a new method for pareto front generation. *Structural and multidisciplinary optimization 31*, 2 (2006), 105–116.

[32] KONA, S., BANSAL, A., BLAKE, M. B., BLEUL, S., AND WEISE, T. Wsc-2009: a quality of service-oriented web services challenge. In *2009 IEEE Conference on Commerce and Enterprise Computing* (2009), IEEE, pp. 487–490.

[33] KOPECKÝ, J., VITVAR, T., BOURNEZ, C., AND FARRELL, J. Sawsdl: Semantic annotations for wsdl and xml schema. *IEEE Internet Computing 11*, 6 (2007).

[34] KOZA, J. R. *Genetic programming: on the programming of computers by means of natural selection*, vol. 1. MIT press, 1992.

[35] KÜSTER, U., KÖNIG-RIES, B., AND KRUG, A. Opossum-an online portal to collect and share sws descriptions. In *Semantic Computing, 2008 IEEE International Conference on* (2008), IEEE, pp. 480–481.

[36] LAUSEN, H., AND FARRELL, J. Semantic annotations for wsdl and xml schema. *W3C recommendation, W3C* (2007), 749–758.

[37] LAUSEN, H., POLLERES, A., AND ROMAN, D. W3c member submission-web service modeling ontology (wsmo). *W3C. Available at¡ URL: http://www. w3. org/Submission/WSMO* (2005).

[38] LÉCUÉ, F. Optimizing qos-aware semantic web service composition. In *International Semantic Web Conference* (2009), Springer, pp. 375–391.

[39] LÉCUÉ, F., AND DELTEIL, A. Making the difference in semantic web service composition. In *Proceedings of the National Conference on Artificial Intelligence* (2007), vol. 22, Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, p. 1383.

[40] LÉCUÉ, F., DELTEIL, A., AND LÉGER, A. Optimizing causal link based web service composition. In *ECAI* (2008), pp. 45–49.

[41] LÉCUÉ, F., AND LÉGER, A. A formal model for semantic web service composition. In *International Semantic Web Conference* (2006), Springer, pp. 385–398.

[42] LI, G., LIAO, L., SONG, D., AND ZHENG, Z. A fault-tolerant framework for qos-aware web service composition via case-based reasoning. *International Journal of Web and Grid Services 10*, 1 (2014), 80–99.

[43] LIU, S., LIU, Y., JING, N., TANG, G., AND TANG, Y. A dynamic web service selection strategy with qos global optimization based on multi-objective genetic algorithm. In *International Conference on Grid and Cooperative Computing* (2005), Springer, pp. 84–89.

[44] MA, H., SCHEWE, K.-D., THALHEIM, B., AND WANG, Q. A formal model for the interoperability of service clouds. *Service Oriented Computing and Applications 6*, 3 (2012), 189–205.

[45] MA, H., WANG, A., AND 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*. Springer, 2015, pp. 180–205.

[46] MARKOU, G., AND REFANIDIS, I. Non-deterministic planning methods for automated web service composition. *Artificial Intelligence Research 5*, 1 (2015), 14.

[47] MARTIN, D., BURSTEIN, M., HOBBS, J., LASSILA, O., MCDERMOTT, D., MCILRAITH, S., NARAYANAN, S., PAOLUCCI, M., PARSIA, B., PAYNE, T., ET AL. Owl-s: Semantic markup for web services. *W3C member submission 22* (2004), 2007–04.

[48] MCILRAITH, S. A., SON, T. C., AND ZENG, H. Semantic web services. *IEEE intelligent systems 16*, 2 (2001), 46–53.

[49] MIER, P. R., PEDRINACI, C., LAMA, M., AND MUCIENTES, M. An integrated semantic web service discovery and composition framework.

[50] MOGHADDAM, M., AND DAVIS, J. G. Service selection in web service composition: A comparative review of existing approaches. In *Web Services Foundations*. Springer, 2014, pp. 321–346.

[51] O'LEARY, D. Review: Ontologies: A silver bullet for knowledge management and electronic commerce. *The Computer Journal 48*, 4 (2005), 498–498.

[52] OVERDICK, H. The resource-oriented architecture. In *Services, 2007 IEEE Congress on* (2007), IEEE, pp. 340–347.

[53] PALIWAL, A. V., SHAFIQ, B., VAIDYA, J., XIONG, H., AND ADAM, N. Semantics-based automated service discovery. *IEEE Transactions on Services Computing 5*, 2 (2012), 260–275.

[54] PAOLUCCI, M., KAWAMURA, T., PAYNE, T. R., AND SYCARA, K. Semantic matching of web services capabilities. In *International Semantic Web Conference* (2002), Springer, pp. 333–347.

[55] PAPAZOGLOU, M. P. Service-oriented computing: Concepts, characteristics and directions. In *Web Information Systems Engineering, 2003. WISE 2003. Proceedings of the Fourth International Conference on* (2003), IEEE, pp. 3–12.

[56] PAPAZOGLOU, M. T. P., dustdar, s., leymann, f.. service-oriented computing. research roadmap, 2006.

[57] PAREJO, J. A., FERNANDEZ, P., AND 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 (2008), 55–66.

[58] PEER, J. Web service composition as ai planning-a survey. *University of St. Gallen* (2005).

[59] PETRIE, C. J. *Web Service Composition*. Springer, 2016.

[60] QI, L., TANG, Y., DOU, W., AND CHEN, J. Combining local optimization and enumeration for qos-aware web service composition. In *Web Services (ICWS), 2010 IEEE International Conference on* (2010), IEEE, pp. 34–41.

[61] RAO, J., AND SU, X. A survey of automated web service composition methods. In *International Workshop on Semantic Web Services and Web Process Composition* (2004), Springer, pp. 43–54.

[62] RAO, J., AND SU, X. Semantic web services and web process composition, volume 3387 of lncs, chapter a survey of automated web service composition methods, 2005.

[63] RENDERS, J.-M., AND FLASSE, S. P. Hybrid methods using genetic algorithms for global optimization. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 26*, 2 (1996), 243–258.

[64] RODRIGUEZ-MIER, P., MUCIENTES, M., LAMA, M., AND COUTO, M. I. Composition of web services through genetic programming. *Evolutionary Intelligence 3*, 3-4 (2010), 171–186.

[65] SAHAI, A., MACHIRAJU, V., SAYAL, M., VAN MOORSEL, A., AND CASATI, F. Automated sla monitoring for web services. In *International Workshop on Distributed Systems: Operations and Management* (2002), Springer, pp. 28–41.

[66] SHET, K., ACHARYA, U. D., ET AL. A new similarity measure for taxonomy based on edge counting. *arXiv preprint arXiv:1211.4709* (2012).

[67] VAN VELDHUIZEN, D. A., AND LAMONT, G. B. On measuring multiobjective evolutionary algorithm performance. In *Evolutionary Computation, 2000. Proceedings of the 2000 Congress on* (2000), vol. 1, IEEE, pp. 204–211.

[68] WADA, H., SUZUKI, J., YAMANO, Y., AND OBA, K. E^3: A multiobjective optimization framework for sla-aware service composition. *IEEE Transactions on Services Computing 5*, 3 (2012), 358–372.

[69] WANG, D., HUANG, H., AND XIE, C. A novel adaptive web service selection algorithm based on ant colony optimization for dynamic web service composition. In *Algorithms and Architectures for Parallel Processing*. Springer, 2014, pp. 391–399.

[70] WANG, P., DING, Z., JIANG, C., AND ZHOU, M. Automated web service composition supporting conditional branch structures. *Enterprise Information Systems 8*, 1 (2014), 121–146.

[71] WHITLEY, D. A genetic algorithm tutorial. *Statistics and computing 4*, 2 (1994), 65–85.

[72] XIANG, F., HU, Y., YU, Y., AND WU, H. Qos and energy consumption aware service composition and optimal-selection based on pareto group leader algorithm in cloud manufacturing system. *Central European Journal of Operations Research 22*, 4 (2014), 663–685.

[73] YAO, Y., AND CHEN, H. Qos-aware service composition using nsga-ii 1. In *Proceedings of the 2nd International Conference on Interaction Sciences: Information Technology, Culture and Human* (2009), ACM, pp. 358–363.

[74] YIN, H., ZHANG, C., ZHANG, B., GUO, Y., AND LIU, T. A hybrid multiobjective discrete particle swarm optimization algorithm for a sla-aware service composition problem. *Mathematical Problems in Engineering 2014* (2014).

[75] YU, Q., AND BOUGUETTAYA, A. Efficient service skyline computation for composite service selection. *Knowledge and Data Engineering, IEEE Transactions on 25*, 4 (2013), 776–789.

[76] YU, Q., LIU, X., BOUGUETTAYA, A., AND MEDJAHED, B. Deploying and managing web services: issues, solutions, and directions. *The VLDB JournalThe International Journal on Very Large Data Bases 17*, 3 (2008), 537–572.

[77] YU, Y., MA, H., AND ZHANG, M. An adaptive genetic programming approach to qos-aware web services composition. In *2013 IEEE Congress on Evolutionary Computation* (2013), IEEE, pp. 1740–1747.

[78] ZENG, L., BENATALLAH, B., DUMAS, M., KALAGNANAM, J., AND SHENG, Q. Z. Quality driven web services composition. In *Proceedings of the 12th international conference on World Wide Web* (2003), ACM, pp. 411–421.

[79] ZHANG, Q., AND LI, H. Moea/d: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on evolutionary computation 11*, 6 (2007), 712–731.

[80] ZHANG, W., CHANG, C. K., FENG, T., AND JIANG, H.-Y. Qos-based dynamic web service composition with ant colony optimization. In *Computer Software and Applications Conference (COMPSAC), 2010 IEEE 34th Annual* (2010), IEEE, pp. 493–502.

[81] ZITZLER, E. Evolutionary algorithms for multiobjective optimization: Methods and applications.

[82] ZITZLER, E., DEB, K., AND THIELE, L. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary computation 8*, 2 (2000), 173–195.

[83] ZITZLER, E., LAUMANNS, M., THIELE, L., ET AL. Spea2: Improving the strength pareto evolutionary algorithm, 2001.