Expert Systems with Applications 41 (2014) 3809–3824

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Review

Cloud computing service composition: A systematic literature review

Amin Jula [a,*], Elankovan Sundararajan [b], Zalinda Othman [a]

[a] Data Mining and Optimization Research Group, Centre for Artificial Intelligence, Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia, UKM Bangi, 43600 Selangor, Malaysia
[b] Centre of Software Technology and Management, Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia, UKM Bangi, 43600 Selangor, Malaysia

ARTICLE INFO

Keywords:
Cloud computing service composition
Systematic literature review
Quality of service parameter
QoS
Research objectives
Importance percentage of quality of service parameters

ABSTRACT

The increasing tendency of network service users to use cloud computing encourages web service vendors to supply services that have different functional and nonfunctional (quality of service) features and provide them in a service pool. Based on supply and demand rules and because of the exuberant growth of the services that are offered, cloud service brokers face tough competition against each other in providing quality of service enhancements. Such competition leads to a difficult and complicated process to provide simple service selection and composition in supplying composite services in the cloud, which should be considered an NP-hard problem. How to select appropriate services from the service pool, overcome composition restrictions, determine the importance of different quality of service parameters, focus on the dynamic characteristics of the problem, and address rapid changes in the properties of the services and network appear to be among the most important issues that must be investigated and addressed. In this paper, utilizing a systematic literature review, important questions that can be raised about the research performed in addressing the above-mentioned problem have been extracted and put forth. Then, by dividing the research into four main groups based on the problem-solving approaches and identifying the investigated quality of service parameters, intended objectives, and developing environments, beneficial results and statistics are obtained that can contribute to future research.

1. Introduction

In recent years, the importance of affordable access to reliable high-performance hardware and software resources and avoiding maintenance costs and security concerns has encouraged large institution managers and stakeholders of information technology companies to migrate to cloud computing. The birth of giant trustworthy clouds has led to a dramatic reduction in apprehension toward such an approach.

There are two challenges to address from the standpoint of the significance of all of the needed service accessibilities and efficient allocation possibilities. First, anticipating all of the possible required services is extremely difficult, particularly for software services. Designing and providing simple and single fundamental services by different service providers will be considered constitutive and constructive parts of complicated required services and can be utilized in encountering this problem. The second challenge is in selecting the optimum required single services, which are provided by different service providers with different quality of service (QoS) attributes; an optimal combination for forming a complicated service must be composed. Addressing this challenge as an optimization problem is an NP-hard problem because it exposes a very large number of similar single services to different service providers in the cloud.

Service composition is one of the best approaches that has been proposed by researchers and applied by cloud providers; this approach can consider both of the mentioned challenges simultaneously. Selecting appropriate services from a service pool, addressing service composition restrictions, determining the important QoS parameters, understanding the dynamic characteristics of the problem, and having rapid changes in the properties of the services and network are some important issues that must be addressed in this approach to assure the service users' satisfaction.

In the early 2000s and in the years before applications in cloud computing, service composition was introduced and investigated for web services (Kosuga et al., 2002; Milanovic & Malek, 2004; Schmid, Chart, Sifalakis, & Scott, 2002; Singh, 2001). Different artificial and evolutionary algorithms (Ai & Tang, 2008; Canfora, Di Penta, Esposito, & Villani, 2005; Liao et al., 2011; Luo et al., 2011; Tang, Ai, & IEEE, 2010; Wang, 2009; Zhao, Ma, Wang, & Wen, 2012; Zhao et al., 2012) and classic algorithms (Gabrel, Manouvrier, Megdiche, Murat, & IEEE, 2012; Gao, Yang, Tang, Zhang, & Society, 2005; Gekas & Fasli, 2005; Liu, Li, & Wang, 2012; Liu, Wang, Shen, Luo, & Yan, 2012; Liu, Xiong, Zhao, Dong, & Yao, 2012; Liu, Zheng, Zhang, & Ren, 2011; Torkashvan & Haghighi, 2012) have been applied extensively to solve the problem. Designing workflows and frameworks for the composition of single services to achieve specific goals is another approach that has been

* Corresponding author.
E-mail addresses: amin.jula@gmail.com, aminjula@ftsm.ukm.my (A. Jula), elan@ftsm.ukm.my (E. Sundararajan), zalinda@ftsm.ukm.my (Z. Othman).

0957-4174/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.eswa.2013.12.017

observed in the field (Chen, Li, & Cao, 2006; He, Yan, Jin, & Yang, 2008; Song, Dou, & Chen, 2011; Song, Dou, & Chen, 2011).

Service composition techniques were first applied in cloud computing systems in 2009 (Kofler, ul Haq, & Schikuta, 2009; Zeng, Guo, Ou, & Han, 2009). Afterward, a substantial effort was made in this area. The number of ongoing studies in cloud computing service composition is rapidly increasing due to the increasing tendency of researchers within different areas of expertise to address the problem. A glimpse of reliable published works shows that researchers who are interested in this promising area face a tremendous number of novel ideas, mechanisms, frameworks, algorithms and approaches, and further expanding the scope of problems. Furthermore, several existing datasets, effective QoS parameters and implementation environments with different features and effects should be recognized. Hence, and due to the absence of related surveys, a systematic review on cloud computing service composition is necessary and will help facilitate future researches. A systematic review in which the most important aspects of the accomplished researchers must be investigated, and useful information and statistics must be extracted.

This paper provides a systematic literature review on state-of-the art approaches and techniques in addressing cloud computing service composition. The discussed advancements and developments of this topic provide useful information to motivate further investigations in this area. Identifying the different objectives of performing cloud computing service composition studies and the reasonable and purposeful classifications of such divergent approaches and mechanisms is a major achievement of the review. Furthermore, this study extracts all of the considered QoS parameters, introduces the most significant and the least considered parameters and calculates the importance parameter percentages, aiming to eliminate barriers to future research efforts, such as proposing comprehensive and reliable mathematical models for calculating composite service QoS values. The goals of this research also include declaring the appropriate, investigated QoS datasets, utilizing software in generating problems for evaluating methods and discussing the most widely used implementation environments.

Because the investigated subject is very extensive, it is impossible to include all relevant topics. Hence, some related subjects do not fall within the research scope of this review but will be briefly mentioned. Providing network services requires common languages and protocols and is closely related to service composition. However, services that provide details will not be considered in this work. Strong and independent studies are required to address research methodologies and experimental and statistical performance evaluation strategies. Describing the accurate significance of the parameters for real cloud customers is also beyond the scope of this report but can be achieved by conducting comprehensive studies, utilizing interviews and questionnaires and adopting appropriate statistical methods.

The structure of this paper is organized as follows. After the introduction, the main aims of the paper and research questions will be defined and described in Section 2. Briefly, cloud computing and its characteristics will be explained in Section 3. Cloud computing service composition *(CCSC)* will be defined in Section 4, including its challenges and classified applied approaches. In Section 5, an extensive discussion on the objectives of the investigated research, their approaches, and utilized datasets is provided, and effective information and statistics are extracted for future research. The final sections of the paper contain the conclusions and references.

2. Survey goals and execution

Survey goals and research questions are described in Section 2.1, and the statistics on published and presented papers in different journals and conferences are presented in Section 2.2. The authors of the present paper have profited from "Guidelines for performing Systematic Literature Reviews in Software Engineering" (Kitchenham & Charters, 2007) and have also used (Garousi & Zhi, 2013) to conduct and perform this research.

2.1. Survey goals and research questions

The present research aims at collecting and investigating all of the credible and effective studies that have examined *CCSC*. More specifically, the extraction of salient features and methods of papers will be considered, and their characteristics will be described.

To achieve the above-mentioned goals and identify the methods that have been selected by researchers for their studies and result assessment methods, case studies are covered for which new methods are proposed and datasets and benchmarks are used. Most researchers have considered QoS parameters and proposed objective functions and user trends that are important in designing these functions. The following research questions (RQs) are raised.

RQ 1. What are the main goals of the researches?
RQ 2. What is the proposed approach and what are the methods used? How have the researchers conducted the research?
RQ 3. What datasets or benchmarks are used and what case studies are considered?
RQ 4. What evaluating procedures have been used to assess the results in each paper?
RQ 5. What other research has been considered in each paper to compare the results?
RQ 6. What QoS parameters are accounted for?
RQ 7. How can the user requirements and tendencies be considered in the applied objective function?

2.2. Publication statistics

In this study, attempts have been made to examine all of the papers that have been published on *CCSC* in particular and that include novel methods or interesting ideas. To achieve this goal and to answer the research questions in Section 2.1, 34 papers that were published from 2009 to December 2013 were selected from different high-level refereed journals and prestigious international conferences and are considered in Section 4. In each study, if it was required to be familiar with some concepts and methods and to read further on the topic, other books and papers are proposed and referred to. The result of this effort is a comprehensive collection of resources that can provide an acceptable level of concepts and information about the service composition problem in cloud computing and the different views of addressing this problem that are introduced in the literature.

3. Cloud computing

3.1. Cloud definition

There are different definitions for cloud computing in the literature, many of which do not cover all of the features of the cloud. In one attempt, Vaquero et al. attempted to extract a comprehensive definition using 22 different compliments (Vaquero, Rodero-Merino, Caceres, & Lindner, 2008). Efforts have been made to standardize the definition of the cloud, in which we accept the cloud definition provided by the National Institute of Standards and Technology (NIST) (Peter Mell, 2011).

The NIST cloud computing definition: "*Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks,*

servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model is composed of five essential characteristics, three service models, and four deployment models".

The cloud cannot be considered to be a new concept or technology that arose in recent years; instead, its root can be found in what John McCarthy described as the ability to provide computational resources as a "utility" (Hamdaqa & Tahvildari, 2012). Based on standard material presented by NIST, cloud computing is composed of five main characteristics, and two other characteristics are added based on the literature, with three spanning service models and four models of deployment, which will be described in some detail in the following sections (see Fig. 1).

3.2. Cloud computing characteristics

On-demand self-service. A user can request one or more services whenever he needs them and can pay using a "pay-and-go" method without having to interact with humans using an online control panel.

Broad network access. Resources and services that are located in different vendor areas in the cloud can be available from an extensive range of locations and can be provisioned through standard mechanisms by inharmonious thin or thick clients. The terms "easy-to-access standardized mechanisms" and "global reach capability" are also used to refer to this characteristic (Hamdaqa & Tahvildari, 2012; Yakimenko et al., 2009).

Resource pooling. Providing a collection of resources simulates the behavior of a single blended resource (Wischik, Handley, & Braun, 2008). In other words, the user does not have knowledge and does not need to know about the location of the provided resources. This approach helps vendors to provide several different real or virtual resources in the cloud in a dynamic manner.

Rapid elasticity. Fundamentally, elasticity is another name for scalability; elasticity means the ability to scale up (or scale down) resources whenever required. Users can request different services and resources as much as they need at any time. This characteristic is so admirable that Amazon, as a well-known cloud service vendor, has named one of its most popular and commonly used services the Elastic Compute Cloud (EC2).

Measured service. Different aspects of the cloud should automatically be controlled, monitored, optimized, and reported at several abstract levels for the resources of both the vendors and consumers.

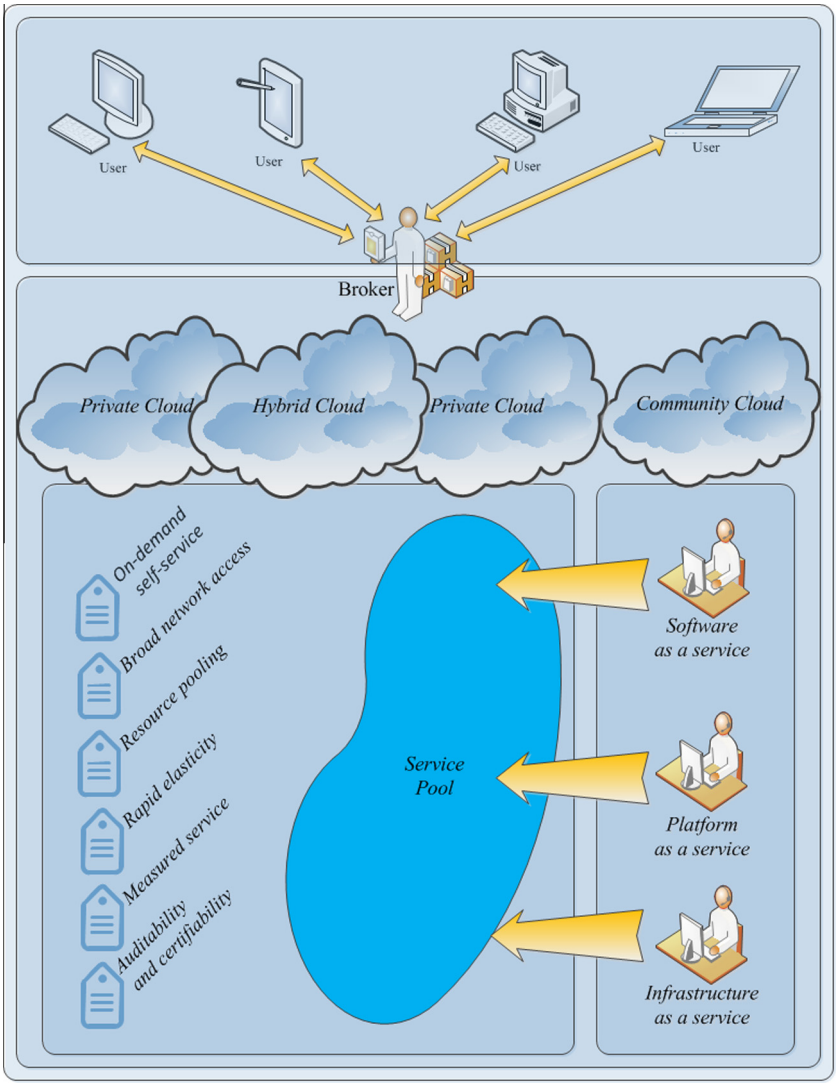

Fig. 1. Cloud computing, characteristics, deployment models, service pool, and types of services and users.

Multi-Tenacity. This concept is the fifth cloud characteristic that is suggested by the Cloud Security Alliance. Multi-tenacity means that it is essential to have models for policy-driven enforcement, segmentation, isolation, governance, service levels, and chargeback/billing for different consumer categories (Espadas et al., 2013).

Auditability and certifiability. It is important for services to prepare logs and trails to make it possible to evaluate the degree to which regulations and policies are observed (Hamdaqa & Tahvildari, 2012).

3.3. Cloud computing service models

Definition 1 (*Service*). A service is a mechanism that is capable of providing one or more functionalities, which it is possible to use in compliance with provider-defined restrictions and rules and through an interface (Ellinger, 2013).

Definition 2 (*Platform*). A platform is a fundamental computer system that includes hardware equipment, operating systems, and, in some cases, application development tools and user interfaces on which applications can be deployed and executed.

Definition 3 (*Infrastructure*). Infrastructure refers to underlying physical components that are required for a system to perform its functionalities. In information systems, these components can contain processors, storage, network equipment, and, in some cases, database management systems and operating systems.

Software as a Service (*SaaS*). A software or application that is executing on a vendor's infrastructure is recognized as a service provided that the consumer has limited permission to access; the provision is through a thin client (e.g., a web browser) or a program interface for sending data and receiving results. The consumer is unaware of the application provider's infrastructure and has limited authority to configure some settings.

Platform as a Service (PaaS). In this service model, the service vendor provides moderate basic requisites, including the operating system, network, and servers, and development tools to allow the consumer to develop acquired applications or software and manage their configuration settings.

Infrastructure as a Service (*IaaS*). The consumer has developed the required applications and needs only a basic infrastructure. In such cases, processors, networks, and storage can be provided by vendors as services with consumer provisions.

3.4. Cloud computing deployment models

Public cloud. This approach is the major model of cloud computing; here, the cloud owner provides public services in the vast majority of cases on the Internet based on predefined rules, policies, and a pricing model. Possessing a large number of widespread world resources enables providers to offer a consumer different choices to select appropriate resources while considering the QoS.

Private cloud. A private cloud is designed and established to prepare most of the benefits of a public cloud exclusively for an organization or institute. Setting up such a system because of the utilization of corporate firewalls can lead to decreased security concerns. Because the organization that implements a private cloud is responsible for all of the affairs of the system, facing abundant costs is the blind spot of establishing a private cloud.

Community cloud. Based on their similar requirements, concerns, and policies, a number of organizations establish a community and share cloud computing to be used by their community member's consumers. A third-party service provider or a series of community members can be responsible for providing the required infrastructure of the cloud computing. Lowering costs and dividing expenses between community members along with supporting high security are the most important advantages of a community cloud (Dillon, Chen, & Chang, 2010).

Hybrid cloud. A combination of two or more different public, private, or community clouds led to the creation of a different cloud model called hybrid cloud, in which constitutive infrastructures not only keep their specific properties but also require standardized or agreed functionalities to enable them to communicate with each other with respect to interoperability and portability on applications and data.

4. The cloud computing service composition problem (CCSC)

4.1. CCSC definition

Fast development in the utilization of cloud computing leads to publishing more cloud services on the worldwide service pool. Because of the presence of complex and diverse services, a single simple service cannot satisfy the existing functional requirements for many real-world cases. To complete a complex service, it is essential to have a batch of atomic simple services that work with each other. Therefore, there is a strong need to embed a service composition *(SC)* system in cloud computing.

The process of service introduction, requesting, and binding, as shown in Fig. 2, can be accounted for in such a way that service providers introduce their available services to the broker to expose them to user requests. However, users also send their service requests to the broker, who must select the best service or set of services on the basis of user requirements and tendencies; the broker wants providers to bind selected services to the users with respect to predefined rules and agreements.

Increasing the number of available services causes an increase in the number of similar-function services for different servers. These similar services are located in different places and have distinct values in terms of the QoS parameters. For this reason, *SC* applies appropriate techniques to select an atomic service among the different similar services that are located on distinct servers to allow the highest QoS to be achieved according to the end-user requirements and priorities. Because of intrinsic changes in cloud environments, available services, and end-user requirements, *SC* should be designed dynamically, with automated function capabilities.

Therefore, selecting appropriate and optimal simple services to be combined together to provide composite complex services is one of the most important problems in service composition. The *SC* problem in cloud computing can be defined as determining what atomic simple services should be selected such that the obtained complex composite service satisfies both the functional and QoS requirements based on the end-user requirements. Because of various and abundant effective parameters and a large number of simple services provided by many service providers in the cloud pool, *CCSC* is considered an NP-hard problem (Fei, Dongming, Yefa, & Zude, 2008).

In this paper, it is assumed that every composed service *(CS)* in the cloud consists of n unique services *(USs)* and has p QoS parameters. To terminate a *CS*, a combination of unique services act sequentially in an ordinal workflow (*wf*), as shown in Fig. 3.

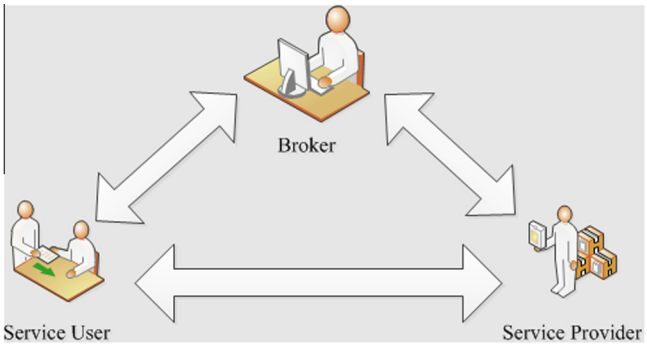

Fig. 2. Process of service introduction, requesting, and binding.

Define $q_i(US_j)$ as the value of QoS parameter i for unique service j. Accordingly, the QoS vector of unique service j is defined as Eq. (1):

$$q(US_j) = (q_1(US_j), q_2(US_j), \ldots, q_p(US_j)) \quad (1)$$

However, if wf_k is the workflow of CS_k, then it is possible to define $Q_i(wf_k)$ to obtain the value of QoS parameter i for workflow k. Then, the QoS vector of workflow k is described in Eq. (2):

$$Q(wf_k) = (Q_1(wf_k), Q_2(wf_k), \ldots, Q_p(wf_k)) \quad (2)$$

4.2. CCSC challenges

The dynamic nature of cloud environments involves occasional and consciously planned changes; these changes expose cloud computing to different challenges in the *SC*. The most remarkable challenges are the following:

Dynamically contracting service providers. The pricing policy of most service providers is determined by service fees based on supply and demand. Thus, mechanisms for updating the table of available resource characteristics must be predicted (Gutierrez-Garcia & Sim, 2013).
Addressing incomplete cloud resources. Optimal service selection by a broker depends on the availability of complete and updated information on the services. Facing several changes in the service characteristics could result in the loss of some data (Gutierrez-Garcia & Sim, 2013; Yi & Blake, 2010).
Describing and measuring QoS attributes of network services. A lack of consensus on the definition and description of network services' QoS parameters among worldwide cloud service vendors is still an important challenge for cloud developers. The absence of agreed-upon ways to measure network QoS is another problem that is not completely solved and that should be considered (Qiang, Yuhong, & Vasilakos, 2012).
Interservice dependency/conflict. Dependency or conflicts that exist among two or more services leads to a complicated service composition problem. In real-world scenarios, encountering dependency and conflict among services is quite common and should be considered in *SC* (Strunk, 2010).
Security. Designing and apprising security rules, policies, and instructions are among the basic responsibilities of cloud service vendors. However, a principled self-administered framework for supplying and provisioning services in which vendors' security concerns and policies are observed must be provided (Takabi, Joshi, & Gail-Joon, 2010; Zissis & Lekkas, 2012).

4.3. Current approaches for CCSC

The different approaches proposed in the literature can be divided into five distinct categories: including classic and graph-based algorithms (CGBAs), combinatorial algorithms (CAs), machine-based approaches (MBAs), structures (STs), and frameworks (FWs). Different studies that belong to these categories are investigated in Sections 4.3.1 to 4.3.5.

4.3.1. Classic and graph-based algorithms

CCSC is a problem in which there are many potential solutions, among which one or a limited number of solutions are optimal. Thus, *CCSC* is known as an optimization problem (Anselmi, Ardagna, & Cremonesi, 2007; Yu & Lin, 2005). Some types of classic algorithms, such as backtracking and branch-and-bound, can be used to solve optimization problems. These algorithms can guarantee that an optimal solution will be found solely by taking exponential time complexity (Neapolitan & Naimipour, 2009). Thus, using classic algorithms to solve optimization problems is possible only with some modifications and improvements for decreasing the execution time.

To achieve a feasible concrete workflow for *CCSC* with respect to the consumer QoS requirement, the problem is considered to be equivalent to a multi-dimensional multi-choice knapsack problem (MMKP) in which a parameter called happiness that is calculated based on QoS parameters is used as the utility (Kofler et al., 2009). To solve the MMKP and obtain the benefits of heterologous multi-processing environments (e.g., grid computing (Preve, 2011)), a parallel form of the branch-and-bound algorithm is proposed in which each node of the decision tree is an independent

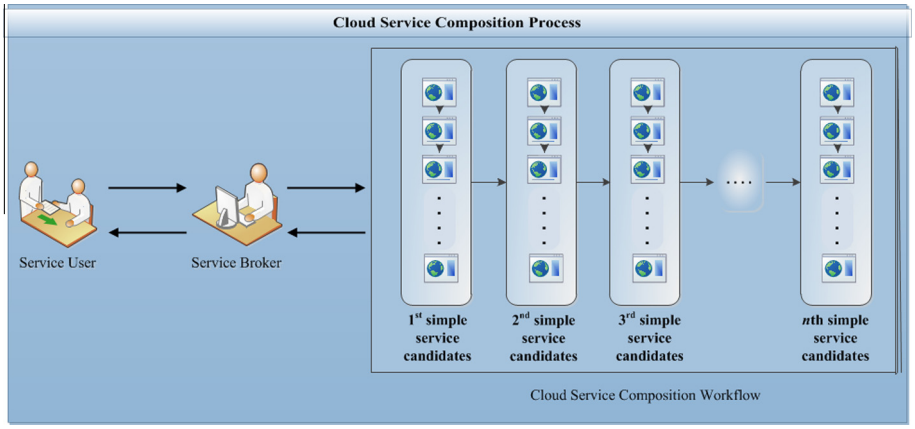

Fig. 3. Cloud service composition process.

instance of a main routine that could be assigned to separated computational nodes. To evaluate the algorithm, researchers randomly generated four different size problems; they executed the problems in serial and parallel modes and compared the results against one another. It is obvious that the most important problem that the proposed algorithm faces is the exponential complexity execution time. The performance evaluation of the algorithm could also be completed and could be more reliable if the results were compared with similar algorithms' results on real-world datasets. To rectify the high execution time problem in the method, a two-phase approach is also proposed in Kofler, Haq, and Schikuta (2010). In this approach, the first phase is similar to the previous phase, whereas for second phase, the previous existing solutions are typically reused for similar requests, and some changes are applied to correspond with the new situation.

After designing a new cloud resource data-saving method, a matching algorithm called SMA is applied in Zeng et al. (2009) to check whether the output parameters of a service and the input parameters of another service are compatible with each other. In this algorithm, the service matching problem is mapped to calculate the semantic similarity of the different input and output parameters of different services. The matching degrees of each pair of services that are stored in a table lead to the composition of a weighted directed graph in which finding all reachable paths for two nodes can yield all of the possible service composition methods. Researchers proposed an improved Fast-EP algorithm called FastB+−EP to obtain all possible paths in a shorter amount of time. However, two-step graph building and searching leads to increased execution time and decreasing algorithm performance, especially in cases when there is an increase in the size of the problem and the number of required services.

In view of all of the cloud participants (e.g., resource vendors and service consumers) as agents and simulating their activities by a colored petri net (Barzegar, Davoudpour, Meybodi, Sadeghian, & Tirandazian, 2011; Jensen, 1995), an agent-based algorithm for *CCSC* is introduced (Gutierrez-Garcia & Sim, 2010). In the proposed algorithm, the broker agent receives a consumer agent's requirements and attempts to find a service vendor agent for each single service. Thereafter, the contract net protocol (Smith, 1981) is used to select proper single services to compose the solution. Using the experimental results, it is proven that the proposed method can achieve solutions in linear time with respect to the number of service vendor agents and that it can consider parallel activities to address heterogeneous services.

A two-phase service composition approach is proposed for dynamic cloud environments in which the service performance changes in different transactions (Shangguang, Zibin, Qibo, Hua, & Fangchun, 2011). In the first phase, the cloud model (Li, Cheung, Shi, & Ng, 1998) is applied to change the qualitative value of the QoS parameters to their quantitative equivalent to calculate the uncertainty level. Thereafter, mixed integer programming (MIP) is utilized in the second phase to find appropriate services in which the binary decision vector is used to determine whether a service is selected. Another two-phase method in which MIP is also applied to focus on service performance fluctuations in the dynamic cloud environment by solving *CCSC* is developed in Zhu, Li, Luo, and Zheng (2012). To decrease the number of candidate single services, some appropriate services are selected based on the history of single services and using K-means in the first phase of the proposed method. In the second phase, MIP is used to select the best single services among the preliminary selected services. The authors proved that their two-phase approach can outperform HireSome (Lin, Dou, Luo, & Jinjun, 2011).

Applying linear programming (LP) (Korte & Vygen, 2012; Vanderbei, 2008) for optimizing virtual machine resource allocation in physical servers for video surveillance was proposed in Hossain, Hassan, Al Qurishi, and Alghamdi (2012) for the first time. Composing an optimal set of media services to prepare a composite service over virtual machine resource allocation was also considered. To reach this goal, the authors mapped the virtual machine resource allocation problem to the multi-dimensional bin-packing problem and used LP to solve it. They also suggested the use of a customized best-fit decreasing method (Kou & Markowsky, 1977) to solve the problem, which considerably increases the probability of finding appropriate results compared to LP but cannot guarantee that an optimal solution will be obtained. To evaluate two proposed methods, randomly generated service composition problems were considered, and the results are compared to fractional knapsack mapping and round-robin allocation execution results.

For generating composite services with the lowest execution cost a two-phase method is applied in Liu et al. (2012). The first phase includes utilizing a state transition matrix for the analysis of the dynamic process of composite service execution. In this phase, each composite service status was modeled on a state transition diagram, which is used to produce a state transition matrix. The execution cost of each composite service can be calculated using its state transition matrix. In phase two, Business Process Execution Language for Web Services (BPEL4WS) (Huang, Lo, Chao, & Younas, 2006) is used to find an optimal solution. Because the process of BPEL4WS, which is known as the distributed traffic model, is extremely time consuming, researchers divided it into three parts, each of which is executed by an independent computer.

To consider the user preferences and different resource characteristics bundled together, Liu et al. proposed a three-layer hierarchical structure; the first layer includes optimal service selection; the second layer is called the criterion layer and includes timeliness, stability, and security, based on which services are divided into three categories; and the third layer is designed for the representation of nine additional QoS parameters (Liu et al., 2012). Next, the phases of the proposed algorithm (SSUP) include generating and normalizing a user requirement matrix, generating a QoS weight specification using the analytic hierarchy process (AHP) (Jeonghwan, Rothrock, McDermott, & Barnes, 2010), generating and normalizing the service attribute matrix, and performing the service-demand similarity calculation. The experimental results indicate that SSUP could outperform AHP when solving the *CCSC* problem.

Worm et al. successfully addressing two antithetical aims for the same provider, namely, the aims of revenue maximization and quality assurance (Worm, Zivkovic, van den Berg, & van der Mei, 2012). The authors based their method on three decision criteria: service availability at the decision instant, executing cost, and resting time to deadline. With respect to the response time and with an emphasis on service availability, dynamic programming (DP) is used to achieve the main goals of the study, in which it is necessary to save intermediate results to select the best service among all available services. In addition, in the case of a deadline, an arrival decision rule would select services with the lowest price for the remaining tasks. The proposed algorithm is executed for static and dynamic service composition but uses real-world datasets and benchmarks, comparing the results with the previous research; calculation time and memory complexity are neglected.

Zhou and Mao proposed a cloud-based semantics approach for the composition of web services utilizing a Bayesian decision (Zhou & Mao, 2012). The authors applied a Bayesian approach to anticipate the semantics of a web service for which a discovery graph is generated based on a cloud to use for implementation. They also obtained relations that are based on the graph that could encounter a Markov chain (Song, Geng, Kusiak, & Xu, 2011). In addition, using an equation for the cosine theorem (Xiangkun & Yi, 2010), it is possible to achieve a similarity of services, which helps to

identify the cloud service interface through Web services and develop a composition system. The proposed method has been applied to Amazon services and can be compared with an existing approach (Stewart, 2009).

On the basis of variability modeling (Sinnema & Deelstra, 2007), cloud feature models (CFMs) are presented as mechanisms to explain cloud services and user requirements together to prepare a suitable ground for their cloud service selection process (CSSP) (Wittern, Kuhlenkamp, & Menzel, 2012). A CFM is represented by a directed graph in which the nodes play service roles and the edges denote the relations between services. Furthermore, the CSSP should satisfy user objectives and also respect the requirements. To satisfy these goals, the user must enter the objectives and requirements to start the process, and the CSSP uses decision support tools to generate a list of offered services based on the input model. Focusing on the dynamic characteristics of cloud computing and updating the graph throughout the execution could help the proposed method to enhance its capabilities.

A two-step procedure to satisfy users' QoS requirements is proposed (Huang, Liu, Yu, Duan, & Tanaka, 2013) in which in the first step, for each user's request, the proposed method attempts to select single services that meet the first two types of user functional requirements and that eliminate the remaining requirements. Then, a virtual network of service providers of selected services is generated and modeled by a directed acyclic graph (DAG). In the second step, in the case of a single user's QoS requirement, it is sufficient to apply an algorithm to determine the shortest path in the DAG. However, for the two QoS parameters, the problem is changed to a multi-constrained path problem (Korkmaz & Krunz, 2001) when this constructing auxiliary graph FPTAS (CAGF) is used. In the CAGF, an auxiliary graph is used in which the two-weight preliminary DAG is changed to a one-weight DAG (which can be solved as in the previous case) by considering the first QoS parameter as a weight and merging the second parameter into the connectivity among the expanded vertices. Researchers have also proposed a method for considering more than two QoS parameters. The execution time and returned path weight (the final QoS) are two parameters that are considered when making a comparison with existing DAG-based algorithms to determine the performance of the method; however, generating several graphs in the algorithm is a time-consuming activity that increases the execution time. This interesting study could also be enriched by addressing known datasets.

To obtain the best set of single services for service composition, three algorithms are proposed (Qi & Bouguettaya, 2013) for generating a service skyline that can be considered a specific set of service vendors that other possible sets cannot dominate with respect to the QoS parameters (Yu & Bouguettaya, 2010). The basic idea of these algorithms is based on reducing the search space. The first algorithm is called OPA, and it examines all of the service execution plans, one for each phase, and saves the best found solution. The researchers also applied some improvements on OPA for decreasing the processing time and the consumed space. DPA is a second algorithm; it uses a tree structure and is based on the progressive examination of service execution plans that are sorted in ascending order based on their score and progressive results output. This progressive method provides the algorithm pipeline capabilities. To overcome the problem of the repetition of nodes and thus overhead, researchers have proposed reducing the parent table data structure. For a third attempt, a linear approach is used to design a bottom-up algorithm (BUA) to address the expensive computational costs of DPA for an increasing number of services. Evaluating the proposed algorithms by using different models of problems is another advantage of this study.

HireSome-II is proposed (Dou, Zhang, Liu, & Chen, 2013) to improve the reliability of composition QoS values. With HireSome-II, the QoS history of services is investigated for evaluation instead of what the service providers claim. Thus, the context of service composition is implemented using a two-layer tree structure in which the required service is located in the root, and the nodes are located in the second layer as candidate services. In addition, the K-means clustering algorithm can be used to categorize the history of each candidate service into two peer groups. Inserting these two peer groups into the tree will provide a three-layer tree, the Task-Service-History Record tree, which identifies the best performance history for candidate services. Based on the required accuracy, additional history layers can be inserted into the tree. Based on experimental results, the performance of the proposed method proved to be superior to the authors' previous method, HireSome-I, especially in the case of facing abundant candidate services. Despite its strengths, with increasing required services, service providers and history volume, this method will face significant increases in execution time due to the use of K-means for categorizing the history of all candidates.

In (Wu, Zhang, Zheng, Lou, & Wei, 2013) a model for QoS-satisfied predictions in *CCSC* is proposed based on the hidden Markov model (HMM) (Li, Fang, & Xia, 2014; Zeng, Duan, & Wu, 2010). The proposed model was to ensure customer satisfaction of the composite service QoS, utilizing the basic form of HMM, in which QoS parameters are considered as a state-set, and distinct observation-probability functions are each defined for one of the parameters. Due to the high computational complexity of the functions, researchers also applied a two part, forward–backward structure for reducing the complexity, in which each part includes three stages, initialization, recursion and end. To evaluate the proposed model, a real cloud computing simulation system was constructed that provides more than 100 different services. The model parameters can be adjusted with a support vector machine-based algorithm. The obtained evaluation error rates were small but with respect to the complicated structure of the model, which will face many services in the real world. This model is predicted to be confronted with larger error rates and an ineligible execution time.

A novel method for QoS mapping is proposed in which a set of three rules is used to map QoS specifications and guarantees together in the cloud (Karim, Chen, & Miri, 2013). To overcome the complexity, the analytic hierarchy process (AHP) (Ishizaka & Labib, 2011; Jeonghwan et al., 2010) is utilized in which customer preferences, possible solutions and ranking the services, are conditions, search space and the goals, respectively. AHP has also led to specify QoS weights to rank and select candidate services. To evaluate the method, a case study was described, and the obtained results were discussed. Because the AHP was constructed from a graph that included many-to-many relationships, increasing the number of required services and search space led to increased method time complexity, which is unacceptable in real environments.

Two novel cloud service ranking approaches, *CloudRank1* and *CloudRank2*, are proposed in which the basic strategy ranked the services based on the prediction of their QoS (Zibin, Xinmiao, Yilei, Lyu, & Jianmin, 2013). These two algorithms are composed of three steps, including similarity computation, preference value computation and ranking prediction, and are different with respect to preference value treatment. Because of the negative effects of equal preference value treatment in the accuracy of ranking prediction in the first algorithm, the confidence value of service QoS preference values is defined and used in the treatment process of the second algorithm to remove the effects, reaching more accuracy. Researchers have also provided a new dataset called *tpds* 2012 that includes a response-time and throughput of 500 services provided by 300 users to evaluate the proposed algorithms. The evaluation results indicate that the algorithms outperformed

previous approaches; however, based on mathematical considerations, the time complexity function of the algorithms was $O(n^2 m) + O(n^2 m) + O(n^2)$, where n and m are the number of services and users, respectively. Because the number of services is typically much greater than the number of users, apparent increases in the execution time of the algorithms may occur.

4.3.2. Combinatorial algorithms

CCSC is an optimization problem, and a distinctive feature of this problem is being involved in very large search spaces to obtain optimal results. In addition, the importance of achieving an optimal solution in a shorter time frame forces researchers to seriously consider the use of combinatorial algorithms (Knuth, 2011; Abonyi et al., 2013). Thus, different efforts have been conducted using different models of combinatorial algorithms to solve the CCSC problem, and these efforts are investigated in the following studies.

To simplify the composition of services in cloud computing without priority weights, an algebra of communication process (ACP) (Fokkink, 2000) based on a QoS value aggregation algorithm is applied (Zhang & Dou, 2010) in which an artificial neural network (Haykin, 1998) is used to focus on service consumer requirements without predefined user priority parameters. A three-layer perceptron (Shrme, 2011) neural network is used in this study by employing back-propagation (Wang, Wang, Zhang, & Guo, 2011) and least-mean-square-error (Sayed, 2003) algorithms to learn the network and adjust the weights, which are initialized randomly. In the proposed neural network, the number of neurons for the input layer is equal to the number of QoS parameters considered. The number of neurons in the second layer is also equal to the number of neurons in the first layer, and the third layer has one neuron for an output. During the training process, the users' priority information is entered into the neural network to realize suitable values for the weights. Weight parameter efficiency is evaluated by applying the mean square error (Zhou & Bovik, 2009) as the objective function. In this research, datasets and benchmarks are not investigated, and the results are not compared.

By dividing the QoS parameters into three groups, including ascending, descending, and equal QoS attributes, and by using simple additive weighting to normalize the values of those parameters, a new model for calculating the QoS of composite services (Ye, Zhou, & Bouguettaya, 2011) is proposed. These authors also applied a genetic algorithm to solve the *CCSC* problem in which a roulette wheel selection algorithm is used to select chromosomes to execute a crossover operation. The proposed model uses the achievement of QoS services as the fitness value. The results obtained with the proposed method were compared with different existing algorithms, such as LP and culture algorithm (Reynolds, 1999) and has shown to be more efficient.

To achieve the goal of the representation of automatic service compositions for dynamic cloud environments, in Jungmann and Kleinjohann (2012) the problem is modeled as a Markov decision process (Arapostathis, Borkar, Fernández-Gaucherand, Ghosh, & Marcus, 1993; Chang & Yuan, 2009). A set of all possible compositions of services and a set of all possible actions for changing compositions by adding new valid services are generated, and the composite service provider is considered an agent. A reward function is also used to learn the optimal policy and optimal composition for a user request that utilizes utility values, such as previous user feedback and former execution information. The reward function results are used by the agent for subsequent decisions and updated using reinforcement learning techniques (Kaelbling, Littman, & Moore, 1996; Liu & Zeng, 2009). A judgment about this research cannot be made due to a lack of appropriate performance evaluation and comparison of results.

Because of the growing importance of networks in the QoS of cloud service composition, in Klein, Ishikawa, and Honiden (2012) an approach is suggest for considering the non-network and network QoS of services separately. To this end, they estimated the real network latency among the desired services and their applicants with low time complexity by proposing a network model that allowed services that had a lower latency to be selected. Researchers also introduced a QoS equation to calculate the network QoS, latency, and transfer rate. In the last phase of the approach, based on the genetic algorithm, a selection algorithm is designed to apply proposed models to generate composite services, and its results are compared with the Dijkstra algorithm (Neapolitan & Naimipour, 2009) and random selection. The results of this interesting study could be enriched by the use of real-world datasets.

With respect to self-adaptivity (Denaro, Pezze, & Tosi, 2007) in the service provider system, an improved genetic algorithm is proposed (Ludwig, 2012) in which a clonal selection algorithm (de Castro & Von Zuben, 2002) is used instead of tournament selection to select individuals for the crossover and mutation operators in a more discernible manner. The explanation of the proposed algorithm and the experimental results in the original paper do not go into detail regarding the researchers' work on self-adaptivity.

In Yang, Mi, and Sun (2012) game theory is used to propose a service level agreement (SLA)-based service composition algorithm. In this research, SLA is defined as quadruple, including SLA main information, service vendors and consumer information, service type and parameters, and a set of responsibilities for service vendors and consumers. To establish an SLA, *SC* is intended to be a multiple dynamic game, called a bid game, in which service vendors and service consumers are players that aim to achieve their goals. In this competitive game, every consumer should promulgate a price for each requested service based on effective parameters and other consumers' proposed prices, and vendors can assign their services according to received proposed prices with respect to the requested level of the quality of services that are agreed upon and signed in the SLA. The reliability of this method is limited due to the lack of comparison with other techniques and the lack of comparison with real-world datasets.

A parallel form of the chaos optimization algorithm (Jiang, Kwong, Chen, & Ysim, 2012) called FC-PACO-RM is proposed to solve the *CCSC* problem (Fei, Yuanjun, Lida, & Lin, 2013). The researchers attempted to dynamically modify the sequence length based on the evolutionary state of the solutions. They also utilized the roulette wheel selection algorithm before executing the chaos operator to eliminate improper randomly generated solutions and escape from their destructive consequences. Because one of the main goals of this study has been to reduce the execution time, parallelization of the proposed algorithm is also considered. To accomplish this goal, a full-connection topology has been selected because of its high searching capability and message passing interface (MPI) (Barney, 2012). Finally, a novel migration method called Way-Reflex Migration is introduced and applied to reduce communication overhead of fully connected topologies. Compared with a genetic, chaos genetic and chaos optimization algorithm, the proposed method showed better results in view of the best solution fitness and execution time.

In Wang, Sun, Zou, and Yang (2013) particle swarm optimization (PSO) with integer array coding is applied to achieve a fast method to solve the *CCSC* problem. To achieve this goal, the skyline operator with binary decision variables is used to eliminate improper services from the search space. Because it is necessary to contrast all of the services pair-wise when using the skyline operator, its execution time is not acceptable for an increasing number of services. Thus, the researchers used offline skyline services to decrease the response time (Borzsony, Kossmann, & Stocker, 2001). The QWS dataset (Al-Masri & Mahmoud, 2008) and Synthetic Generator (Wang, Sun, & Yang, 2010) are used to evaluate the

algorithms. Comparing the results of the proposed algorithm with another service composition called GOA (Ardagna & Pernici, 2007) demonstrates that the proposed method achieves positive results.

Simultaneous optimization of the response time and execution cost of the service composition process motivated Jula et al. to propose Imperialist competitive-gravitational attraction search algorithm (ICGAS) as a new memetic algorithm (Moscato & Cotta, 2003) for solving the *CCSC* problem (Jula, Othman, & Sundararajan, 2013). Because they wanted to address the enormous number of services that were provided by several service vendors, they attempted to apply the advantages of the exploration capability of an imperialist competitive algorithm (ICA) (Atashpaz-Gargari & Lucas, 2007) and the significant exploitation ability of a gravitation search algorithm (GSA) (Jula & Naseri, 2012) simultaneously. In the proposed algorithm, to balance the process of execution of the two algorithms and to increase the performance, the number of population members of GSA is determined to be equal to 20% of the number of members of the ICA algorithms. Researchers also introduced a new mathematical model for calculating the QoS of the nominated composite services based on user-defined weights for different QoS parameters. Using a very large real-world dataset to evaluate the algorithm and to compare its results with the results from executing a genetic algorithm, PSO and classic ICA have demonstrated suitable results for ICGAS.

A negative selection algorithm, which is inspired from a negative selection mechanism in biological immune recognition and can eliminate improper solutions through an execution process, is used in Zhao, Wen, and Li (2013). In the proposed algorithm, the solutions and candidate services are implemented in the form of vector and integer strings, respectively. There are two different applied models for calculating the local and global fitness. To compute the fitness of the solutions for a local search, users are responsible for defining their criteria with related weights and a set of constraints based on which quality vectors are generated for all of the services for the decision-making process. Global fitness is also computed using an equation that is designed based on local fitness. The performance capability of the proposed algorithm is proven by comparing its results with the results obtained from standard particle swarm intelligence and the clonal selection immune algorithm.

Attempting to select the best services based on overall quality of services a model is designed in which customer feedback and the objective performance analysis results are considered as two inputs, and the output is the quality of services (Lie, Yan, & Orgun, 2013). Because customer feedback is composed of linguistic variables, a mapping table is applied to change the inputs to fuzzy numbers. Then, two input values that were changed to fuzzy ratings and a fuzzy simple additive weighting system (Chou, Chang, & Shen, 2008) are utilized to obtain the model output. Researchers have also applied a filtering mechanism for removing misleading values that are given by unprofessional customers. The method was evaluated with a case study; however, the results have not been compared to other approaches. This method has also not been applied to select a set of services for assessing its effectiveness in service composition. Thus, this method cannot present proper efficiency in the case of facing many QoS parameters, customer feedbacks and requests.

4.3.3. Machine-based methods

In Bao and Dou (2012) researchers designed finite state machines (FSMs) (Koshy, 2004) to consider service correlations and rightful task executing sequences; each FSM is used to implement a group of services that have limitations on their invocation and executing sequence called CWS community and another FSM for a desired composite service called the target process. The proposed method is composed of two phases. In the first phase, a composition service tree is created on the basis of CWS community and a target process. A pruning policy is also applied to eliminate improper paths of the tree and to reduce the processing time. The QoS of all paths in the tree are calculated in the second phase based on user requirements, and the path with the highest QoS is selected as the final solution. To reach this goal, a dichotomous simple additive weighting technique (Liangzhao et al., 2004) is used in the first part, from which the scaling of paths is performed by dividing them into negative and positive criteria, and in the second part, the total QoS of all of the paths is calculated. Researchers could prove appropriate efficiency of their proposed methods by comparing its executing time with the executing time of the enumeration method (Gerede, Hull, Ibarra, & Su, 2004).

4.3.4. Structures

Based on the B+-tree (Bayer & Unterauer, 1977), an index structure has been designed by in Sundareswaran, Squicciarini, and Lin (2012) to simplify the process of information insertion and retrieval for cloud service providers. In the proposed structure, different properties, such as the service type, security, QoS, and measurement and pricing units, have specific locations to be stored and considered. To increase the speed at which the information management operators are executed and appropriate vendor queries can be found, service vendors with the same characteristics should be stored together in adjacent rows. Researchers also proposed a query algorithm based on a designed structure to search the providers' database for the best vendors; after finding k vendors close to the optimal for vendors with each desired service, a refinement procedure is designed to reduce the number of selected vendors and sort them according to their Hamming distances, which entails starting with the optimal and progressing in ascending order to facilitate the selection of better providers. The proposed method is compared with a brute-force search algorithm and has shown almost 100 times better execution speed for solving the *CCSC* problem with 10,000 service providers.

4.3.5. Frameworks

Pham, Jamjoom, Jordan, and Shae (2010) proposed a new framework for service composition in which a composition agent is responsible for receiving the request and providing service management. The agent analyzes each request and divides it into the required single services. Using a knowledge base, the service dependencies are identified, and a service recovery section extracts similar service information. A composition is successful provided that all of the required single services are available and are used to update the knowledge base. There is a packaging engine that generates a new software package by using existing composition together with the new composition and that registers it in a service catalog. Finally, a service delivery section utilizes service catalog information for service provisioning.

Chunqing, Shixing, Guopeng, Bu Sung, and Singhal (2012) proposed a systematic framework for automatic service conflict detection and supplying policies and user requirements. The first phase of the proposed framework is a conflict analysis section that includes two sub-sections called the comparator and analyzer. The comparator checks the conformity of the policies and user requirements based on their priorities, and the analyzer is responsible for uncovering the contradictions between the user requirements and their affiliate relations while applying Satisfiability Modulo Theories (SMT) (Moura & Bjørner, 2011). The filter, allocator, and solver are three parts of the second phase of the framework, which is called the solution derivation. This section finds appropriate single services and eliminates policy-violating services. It determines a set of appropriate single services for each user's needs and finally concludes with the best composite service with respect to policies

and requirements. It uses the backtracking algorithm (Neapolitan & Naimipour, 2009) for the assigned tasks.

According to the definition of trust as a conceptual probability by which a composite service is expected to execute a task as well as expected by the user, a trust-based method and a framework are proposed by Xiaona, Bixin, Rui, Cuicui, and Shanshan (2012) to solve the *CCSC* problem. Applying the trust in the process of service composition is divided into three parts, including trustworthy in service selection, guaranteeing trust in the composition processes, and trustworthy in binding the generated plan. In the designed framework, the requirement analyzer classifies the requirements of the user into their different elements, including functional and non-functional requirements and expected input–output parameters. The service retriever restores information on the services from the resource pool using query requests. Inappropriate services are eliminated from the candidate list by a service filter, and the name and type of the remaining services are identified by an WSDL analyzer. The clustering component, template generator, and binding optimizer are responsible for checking the services' composability, checking the math interfaces, and evaluating the binding plan trust, respectively.

CloudRecommender is a cloud-based three-layer structured service composition system that was proposed by Zhang, Ranjan, Nepal, Menzel, and Haller (2012). The first layer is a configuration management layer in which a cloud service ontology and cloud QoS ontology are located because of its two parts for uncovering services based on their functionality and QoS parameters; the services are mapped to a rational model and data structure. Application logic is the second layer, which is implemented to select single services in the form of SQL queries to include criteria, views, and stored procedures. The third layer is a widget that divides the user interface into four objects, including the computing resources, storage resources, network resources, and recommendation. This layer is implemented using RESTful (Richardson, 2007) and several JavaScript frameworks.

A novel framework is proposed for adaptive service selection in mobile cloud computing (Wu et al., 2013). The framework extracts the QoS preferences of the customer immediately after receiving a request. Then, based on the Euclidean distance, some of the nearest customer preference services are selected and suggested to the service adapter. Finally, the service adapter selects the best service among the suggested services to be assigned to the customer, with respect to device context matching and effectiveness of the service option. To reach the context matching service based on the input information, a fuzzy cognitive map model is also utilized in the service adapter module. The weaknesses of this method include that the proposed framework can only be used to select a single service; furthermore, this strategy has not been compared to other approaches.

5. Discussion

5.1. Objectives of the researches

To respond to the first research question RQ1 and achieve a comprehensive view of the topic, it is essential to categorize the goals of the researches. Objective scrutiny of the considered papers guarantees the existence of nine categories, RO1 to RO9, in which each paper can be placed in one or more categories, as described in Table 1 and Fig. 4(a) and (b). Based on Table 1, the largest amount of researchers' attention has been focused on RO3, and there is a large difference in terms of attention paid in the

Table 1
Desired objectives in the investigated researches.

Reference	Approach	RO1	RO2	RO3	RO4	RO5	RO6	RO7	RO8	RO9
Kofler et al. (2009)	CGBA	√		√						
Zeng et al. (2009)	CGBA			√	√					
Gutierrez-Garcia and Sim (2010)	CGBA			√						
Zhang and Dou (2010)	CA	√		√						
Pham et al. (2010)	FW								√	√
Wang et al. (2011)	CGBA		√	√						
Ye et al. (2011)	CA			√			√			
Zhu et al. (2012)	CGBA			√						
Hossain et al. (2012)	CGBA			√						
Liu et al. (2012)	CGBA	√		√						
Liu et al. (2012)	CGBA			√						
Worm et al. (2012)	CGBA			√		√		√	√	
Zhou and Mao (2012)	CGBA									
Wittern et al. (2012)	CGBA	√								
Jungmann and Kleinjohann (2012)	CA	√		√		√				
Ludwig (2012)	CA			√		√				
Yang et al. (2012)	CA			√						
Bao and Dou (2012)	MBM	√								
Sundareswaran et al. (2012)	ST				√					
Chunqing et al. (2012)	FW	√		√						√
Xiaona et al. (2012)	FW	√								√
Zhang et al. (2012)	FW				√					√
Huang et al. (2013)	CGBA	√		√						
Qi and Bouguettaya (2013)	CGBA			√						
Wang et al. (2013)	CA			√		√				
Jula et al. (2013)	CA	√		√			√			
Zhao et al. (2013)	CA	√		√						
Dou et al. (2013)	CGBA					√				√
Wu et al. (2013)	CGBA	√								√
Karim et al. (2013)	CGBA	√								√
Zibin et al. (2013)	CGBA			√		√				
Wu et al. (2013)	FW	√								√
Lie et al. (2013)	CA	√	√							
Fei Tao et al. (2013)	CA			√						

CGBA, classic and graph-based algorithms; CA, combinatorial algorithms; MBM, machine-based methods; ST, structures; FW, Frameworks.

literature between RO3 and RO1. This difference may be because decreasing the time complexity of the algorithms is a priority of researchers, and user satisfaction is the second-most important objective for researchers. The objective categories of the research are defined as follows:

1. User requirement satisfaction (RO1). Strictly abiding by customers' requirements and providing facilities for them to determine or describe their needs would make them more satisfied.
2. Qualitative to quantitative transformation of QoS parameters (RO2). When accurately considering the qualitative QoS parameters and applying them in decision making, it is important to transform them into quantitative values using reliable methods.
3. Algorithm improvements (RO3). There are many different heuristic and non-heuristic algorithms that are introduced by algorithm designers and applied to solve NP-hard problems. The following efforts to improve the current algorithms and specify them for *CCSC* to obtain the best solutions or reduce the execution time is one of the more often investigated objectives in the literature.
4. Introducing data storage and indexing structures (RO4). Possessing appropriate and well-defined data structures and databases can play an important role in the design of an efficient algorithm. Using a suitable indexing method is also useful in increasing the search speed, especially when the number of cases is very high.
5. Self-adaptability, automaticity, increasing reliability and accuracy, and quality assurance (RO5). Establishing automated and self-adaptable service brokers is unavoidable (Denaro et al., 2007) because of increasing complexity, the number of requests, the number of available services in the pool, their diversity, and the limitation of human abilities. One of the main factors that attracts customers and retains them in utilizing cloud computing is reliability. Because of the importance of providing a reliable and self-adaptable service composition organization, the most important part of a cloud is in its direct contact with customers; researchers must consider this direct contact more seriously than before.
6. Proposing an improvised QoS mathematical model (RO6). Calculating a QoS value for composite services requires a mathematical model in which all aspects, parameters, user requirements, and tendencies are investigated. To reach this goal, some researchers have attempted to present improved models that focus more on these objectives.
7. Revenue maximization (RO7). Encouraging service providers to expose their high-quality services depends on the ability to amass significant profits. Thus, revenue maximization can be noted as an important fundamental aim.
8. Optimization of the service discovery process (RO8). If the service composer policy is not to register services based on predefined requirements, then it must discover required available simple services in the network. It is critical to have the type of policy that uses optimal discovery methods.
9. Proposing new frameworks and structures (RO9). Reaching some basic goals, e.g., the definition of new roles, requires changes in the framework of the *CCSC* and in the structures. In some cases, to achieve a goal, there could be a need to design a new framework. This scenario has led some researchers to be encouraged to design new frameworks or to change the existing frameworks.

5.2. Applied approaches

To address RQ2, all of the proposed or applied approaches are divided into five distinct categories, as mentioned in Section 4.3. These categories and their statistics are summarized in Table 2 and Fig. 5(a) and (b), from which it can be inferred that the highest and lowest percentages of usage can be seen in the classic and graph-based algorithms (52%) and combinatorial algorithms (24%), respectively.

5.3. Investigated datasets

The obtained answers for the third question RQ3 are not promising. Possessing different datasets each of them can support several QoS parameters, and the predefined composition problem are useful and unavoidable for evaluating the proposed approaches and comparing their results. Unfortunately, the number of datasets that are available to all and in the research domain is very low and is limited to three datasets, QWS (Al-Masri & Mahmoud, 2009), WS-DREAM (Zibin, Yilei, & Lyu, 2010) and *tpds* 2012 (Zibin et al., 2013), and an unknown randomly generated dataset RG (Shangguang et al., 2011). Researchers have also used a synthetic generator, rarely. The datasets used in each study are listed in Table 2.

5.4. Significance of QoS parameters and their percentage

Based on the literature review presented in this paper, the priority and importance percentage of the QoS parameters and their

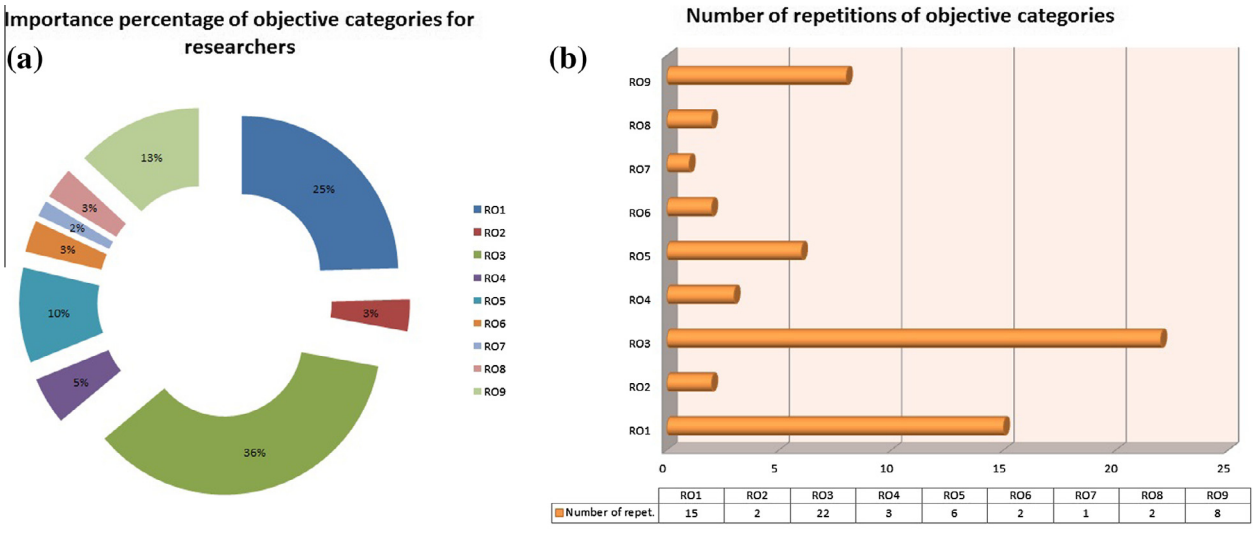

Fig. 4. (a) Importance percentage of objective categories and (b) number of repetitions of objective categories.

relevant factors have not yet been studied comprehensively. Thus, an analysis of the considered QoS parameters in the observed papers and their statistics is essential. In this section, we attempted to extract the importance of different QoS parameters and their percentage considering this field of research.

As specified in Table 2 and to answer RQ6, according to the significance and priority, most researchers have accounted for different QoS parameters, although others have neglected them. Based on the abovementioned parameters and their frequency of occurrence in the literature, it is important to obtain the most important and effective QoS parameters and their importance percentage.

To reach this goal, using Eq. (3), the number of occurrences of a parameter has been counted separately and divided by the sum of the number of occurrences of all parameters. The importance percentage of each parameter is obtained by multiplying its calculated value by 100. The result obtained is the ratio of the importance percentage of every parameter to all of the other parameters. The number of occurrences of all of the parameters and their importance percentages are shown in Fig. 6(a) and (b).

$$imp\_percentage(i) = \frac{occurr\_no(i)}{\sum_{j=1}^{param\_no} occurr\_no(j)} \quad (3)$$

where $imp\_percentage(i)$ is the importance percentage of QoS parameter i, $occur\_no(i)$ is the number of repetitions of QoS parameter i in the investigated papers, and param_no is the number of observed QoS parameters in the literature.

There could be another way to look at the importance percentage of the QoS parameters. It is possible to consider this issue in view of the ratio of the frequency of occurrences of the parameters in the papers. To increase the total number of papers, it is possible to account for all of the papers, including those that did not mention any parameters or excluded them. To reach the results, it is sufficient to divide the number of occurrences of each parameter by the total number of papers. The obtained results are shown in Fig. 7(a) and (b) when including and excluding papers that do not consider the QoS parameters, respectively.

6. Conclusion and future works

Showcasing pertinent achievements and findings with a comprehensive review generally increases the research efforts and papers in that particular scientific field. Proposing novel techniques

Table 2
Utilized tools, datasets and QoS parameters.

Reference	Tools	Dataset	QoS considered parameters
Kofler et al. (2009)	Kepler Workflow tool, CORBA, C++	–	Response time (RT), Cost
Zeng et al. (2009)	Visual C++ 6, PostgreSQL 8.4.	–	Response time (RT), Availability (Avail), Reliability (Reli)
Gutierrez-Garcia and Sim (2010)	Not mentioned	–	Not mentioned
Zhang and Dou (2010)	Not mentioned	–	Not mentioned
Pham et al. (2010)	Not mentioned	–	Not mentioned
Wang et al. (2011)	Matlab 7.6	WS-DREAM, RG	Response time (RT), Cost, Throughput (Throu), Reputation (Reput), Availability (Avail), Reliability (Reli)
Ye et al. (2011)	Not mentioned	–	Response time (RT), Cost, Availability (Avail), Reputation (Reput)
Zhu et al. (2012)	Visual C#.NET	–	Not mentioned
Hossain et al. (2012)	Not mentioned	–	Response time (RT), Cost
Liu et al. (2012)	Not mentioned	–	Cost
Liu et al. (2012)	Not mentioned	WS-DREAM	Not mentioned
Worm et al. (2012)	Not mentioned	–	Response time (RT), Cost, Availability (Avail)
Zhou and Mao (2012)	Not mentioned	–	Not mentioned
Wittern et al. (2012)	Researchers built a tool based on eclipse modeling framework	–	Not mentioned
Jungmann and Kleinjohann (2012)	Not mentioned	–	Not mentioned
Ludwig (2012)	Java	–	Response time (RT), Cost, Availability (Avail), Reliability (Reli)
Yang et al. (2012)	Not mentioned	–	Not mentioned
Bao and Dou (2012)	Not mentioned	–	Response time (RT), Cost, Reliability (Reli), Availability (Avail), Reputation (Reput)
Sundareswaran et al. (2012)	C	–	Not mentioned
Chunqing et al. (2012)	Java, Cauldron, zChaff solver	–	Cost
Xiaona et al. (2012)	Not mentioned	Seekda	Availability (Avail), Durability (Dur)
Zhang et al. (2012)	JavaScript, RESTful	–	Not mentioned
Huang et al. (2013)	Not mentioned	–	Not mentioned
Qi and Bouguettaya (2013)	Java	Using Synthetic Generator	Response time (RT), Cost
Wang et al. (2013)	Matlab 7.6, Lp-Solve 5.5	QWS, Synthetic Generator	Not mentioned
Jula et al. (2013)	Visual C#.NET	WS-DREAM	Response time (RT), Cost
Zhao et al. (2013)	Not mentioned	–	Response time (RT), Cost, Availability (Avail), Reliability (Reli)
Dou et al. (2013)	Not mentioned	–	Cost, Latency, Reputation (Reput),
Wu et al. (2013)	Not mentioned	–	Not mentioned
Karim et al. (2013)	Not mentioned	–	Cost, Response Time (RT), Security (Secur), Reputation (Reput), Availability (Avail), Reliability (Reli), Durability (Dur), Data Control (DC)
Zibin et al. (2013)	Planet-lab, Axis2	*tpds* 2012	Response Time (RT), Throughput (Thr)
Wu et al. (2013)	Not mentioned	–	Not mentioned
Lie et al. (2013)	Not mentioned	–	Not mentioned
Fei Tao et al. (2013)	Not mentioned	–	Response time (RT), Cost, Reliability Reli, Energy, Trust, Maintainability (Maint), Function similarity (FS)

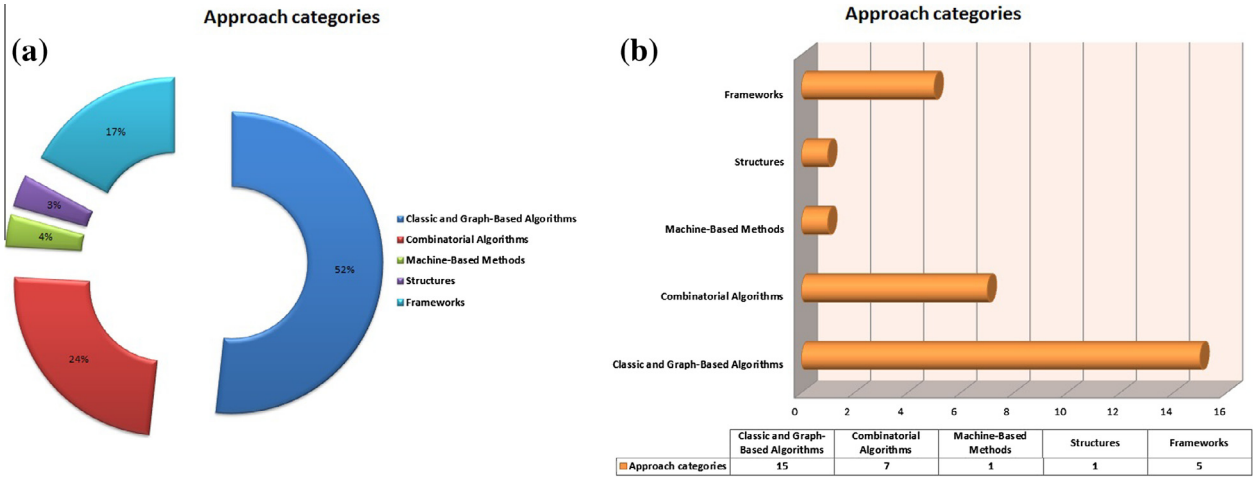

Fig. 5. (a) Approach categories and (b) number of papers in each category.

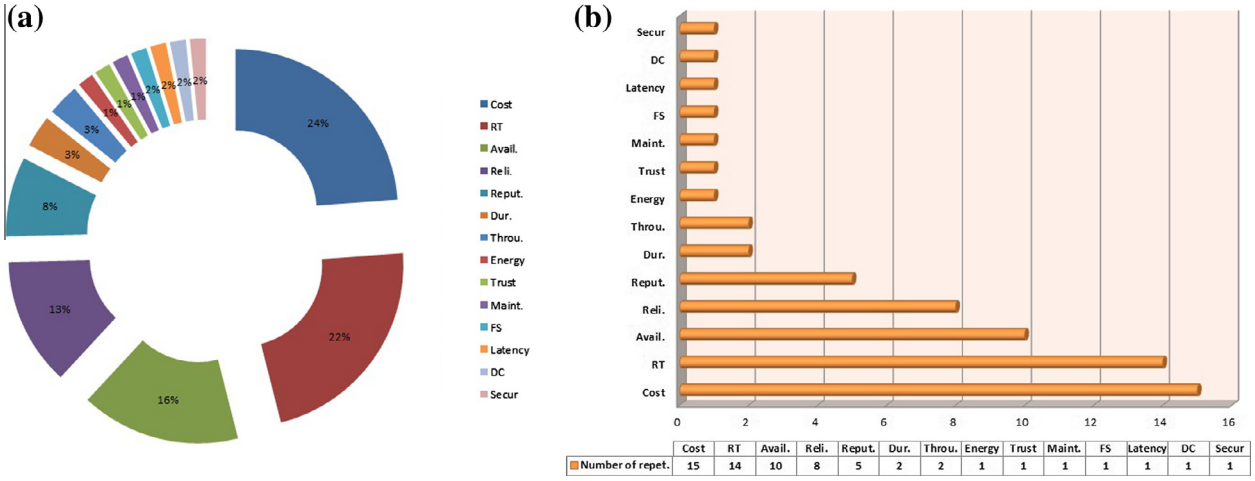

Fig. 6. (a) Percentage of repetition of QoS parameters and (b) number of repetitions of QoS parameters.

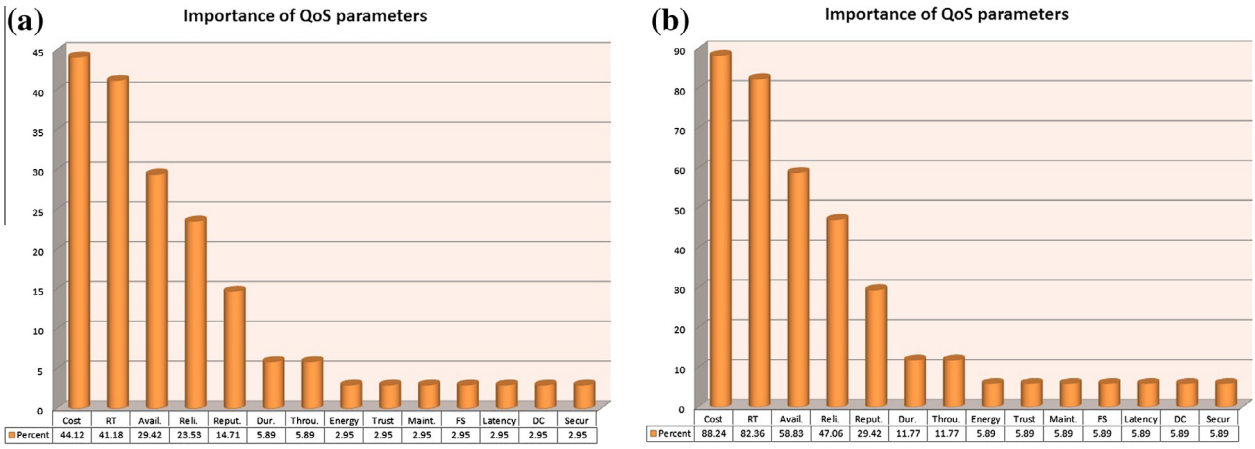

Fig. 7. (a) Percentage of repetition of QoS parameters in all investigated papers and (b) percentage of repetition of QoS parameters in all investigated papers, excluding those that did not mention any parameters.

and approaches for addressing cloud computing service composition from different aspects in addition to addressing the lack of comparable activities were the strong motivating factors for preparing this systematic literature review. This paper has provided a complete definition of the *CCSC* in combination with its associated concepts and a comprehensive analysis of the different applied algorithms, mechanisms, frameworks and techniques extracted from 34 authentic published papers, spanning

2009–2013. The achievements of this review shed light on the research grounds of *CCSC* for future studies.

This investigation demonstrates that all cloud computing service composition innovations and improvements can be categorized into the following five groups: classic and graph-based algorithms, combinatorial algorithms, machine-based methods, structures and frameworks. The most widely applied category is the *classic and graph-based* algorithms group (52%), and the least-used categories are machine-based methods and structures (3% and 4%, respectively). The objectives of these reports can also be divided into 9 categories, among which *algorithm* improvements (RO3) and user requirement satisfaction (RO1) have attracted the most attention (36% and 25%, respectively), while revenue maximization (RO7) has been the least important objective (2%).

Counting the number of QoS parameter occurrences indicated that 14 different parameters have been considered in the literature, among which service cost and response time are the most repeated ones (24% and 22%, respectively). Calculating the importance percentage of the QoS parameters also revealed that the importance percentages of two mentioned parameters after including and excluding papers that do not consider the QoS parameters were 44% and 41% and 88% and 82%, respectively. To evaluate the proposed approaches, 4 QoS datasets have been previously identified, WS-DREAMS2, QWS, tpds2012 and RG, the first three of which are real-world extracted, and the last of which is generated randomly. Synthetic generators and Seedka are also two applications that have applied for runtime generating QoS values.

With respect to the findings in this paper, achieving certain goals is of utmost importance for the planning of future work. Aiming to prepare an identical, competitive environment for comparing the proposed algorithms and approaches, it is indispensable to provide a set of differently sized, standard problems and a comprehensive QoS dataset. The dataset should include a great number of unique services and service providers and encompass cost, response time, availability, reliability and reputation as significant QoS parameters. Another essential research goal is to focus on designing comprehensive mathematical models for calculating the QoS values composite services that cover all of the involved parameters and their importance percentage. Proposing real-time algorithms that can obtain a few optimal composite services for the given requests represent a significant achievement in the field. Furthermore, the less considered objectives (e.g., *RO2*, *RO6*, *RO7*, and *RO8*) should be regarded. Finally, acknowledging the stunning growth in mobile computing, future research efforts should be directed towards this forward-looking area.

References

Ai, L. F., & Tang, M. L. (2008). *A penalty-based genetic algorithm for QoS-aware web service composition with inter-service dependencies and conflicts*. New York: IEEE.

Al-Masri, E., & Mahmoud, Q. H. (2008). Investigating web services on the world wide web. In *Proceedings of the 17th international conference on world wide web* (pp. 795–804). Beijing, China: ACM.

Al-Masri, E., & Mahmoud, Q. H. (2009). Discovering the best web service: A neural network-based solution. In *systems, man and cybernetics, 2009. SMC 2009. IEEE international conference on* (pp. 4250–4255).

Anselmi, J., Ardagna, D., & Cremonesi, P. (2007). A QoS-based selection approach of autonomic grid services. In *Proceedings of the 2007 workshop on service-oriented computing performance. Aspects, issues, and approaches* (pp. 1–8). Monterey, California, USA: ACM.

Arapostathis, A., Borkar, V. S., Fernández-Gaucherand, E., Ghosh, M. K., & Marcus, S. I. (1993). Discrete-time controlled Markov processes with average cost criterion: a survey. *SIAM Journal on Control and Optimization, 31*, 282–344.

Ardagna, D., & Pernici, B. (2007). Adaptive service composition in flexible processes. *IEEE Transactions on Software Engineering, 33*, 369–384.

Atashpaz-Gargari, E., & Lucas, C. (2007). Imperialist competitive algorithm: An algorithm for optimization inspired by imperialistic competition. In *Evolutionary Computation, 2007. CEC 2007. IEEE Congress on* (pp. 4661–4667).

Bao, H. H., & Dou, W. C. (2012). A QoS-aware service selection method for cloud service composition. In *2012 IEEE 26th international parallel and distributed processing symposium workshops & Phd Forum* (pp. 2254–2261). New York: IEEE.

Barney, B. (2012). Message Passing Interface (MPI). In Lawrence Livermore National Laboratory (Vol. 2013).

Barzegar, S., Davoudpour, M., Meybodi, M. R., Sadeghian, A., & Tirandazian, M. (2011). Formalized learning automata with adaptive fuzzy coloured petri net; an application specific to managing traffic signals. *Scientia Iranica, 18*, 554–565.

Bayer, R., & Unterauer, K. (1977). Prefix B-trees. *ACM Transaction on Database Systems, 2*, 11–26.

Borzsony, S., Kossmann, D., & Stocker, K. (2001). The Skyline operator. In *Data Engineering, 2001. Proceedings. 17th international conference on* (pp. 421–430).

Canfora, G., Di Penta, M., Esposito, R., & Villani, M. L. (2005). *An approach for QoS-aware service composition based on genetic algorithms*. New York: Assoc Computing Machinery.

Chang, W.-L., & Yuan, S.-T. (2009). A Markov-based collaborative pricing system for information goods bundling. *Expert Systems with Applications, 36*, 1660–1674.

Chen, L., Li, M. L., & Cao, J. (2006). ECA rule-based workflow modeling and implementation for service composition. *IEICE Transactions on Information and Systems, E89D*, 624–630.

Chou, S.-Y., Chang, Y.-H., & Shen, C.-Y. (2008). A fuzzy simple additive weighting system under group decision-making for facility location selection with objective/subjective attributes. *European Journal of Operational Research, 189*, 132–145.

Chunqing, C., Shixing, Y., Guopeng, Z., Bu Sung, L., & Singhal, S. (2012). A Systematic Framework Enabling Automatic Conflict Detection and Explanation in Cloud Service Selection for Enterprises. In *Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on* (pp. 883–890).

de Castro, L. N., & Von Zuben, F. J. (2002). Learning and optimization using the clonal selection principle. *IEEE Transactions on Evolutionary Computation, 6*, 239–251.

Denaro, G., Pezze, M., & Tosi, D. (2007). Designing self-adaptive service-oriented applications. In *Autonomic Computing, 2007. ICAC '07. Fourth International Conference on* (pp. 16–16).

Dillon, T., Chen, W., & Chang, E. (2010). Cloud computing: issues and challenges. In *Advanced Information Networking and Applications (AINA), 2010 24th IEEE International Conference on* (pp. 27–33).

Dou, W., Zhang, X., Liu, J., & Chen, J. (2013). HireSome-II: Towards privacy-aware cross-cloud service composition for big data applications. *IEEE Transactions on Parallel and Distributed Systems*, 1.

Ellinger, R. S. (2013). Governance in SOA Patterns (white paper). In The Northrop Grumman Corporation for Consideration by OASIS SOA Reference Architecture Team (pp. 1–11).

Espadas, J., Molina, A., Jiménez, G., Molina, M., Ramírez, R., & Concha, D. (2013). A tenant-based resource allocation model for scaling software-as-a-service applications over cloud computing infrastructures. *Future Generation Computer Systems, 29*, 273–286.

Fei, T., Dongming, Z., Yefa, H., & Zude, Z. (2008). Resource service composition and its optimal-selection based on particle swarm optimization in manufacturing grid system. *IEEE Transactions on Industrial Informatics, 4*, 315–327.

Fei, T., Yuanjun, L., Lida, X., & Lin, Z. (2013). FC-PACO-RM: a parallel method for service composition optimal-selection in cloud manufacturing system. *IEEE Transactions on Industrial Informatics, 9*, 2023–2033.

Fokkink, W. (2000). *Introduction to Process Algebra*. Springer-Verlag New York, Inc..

Gabrel, V., Manouvrier, M., Megdiche, I., & Murat, C.IEEE. (2012). *A new 0-1 linear program for QoS and transactional-aware web service composition*. New York: IEEE.

Gao, A. Q., Yang, D. Q., Tang, S. W., Zhang, M., & Society, I. C. (2005). *Web service composition using integer programming-based models*. Los Alamitos: IEEE Computer Soc.

Garousi, V., & Zhi, J. (2013). A survey of software testing practices in Canada. *Journal of Systems and Software, 86*, 1354–1376.

Gekas, J., & Fasli, M. (2005). Automatic Web service composition based on graph network analysis metrics. In R. Meersman, Z. Tari, M. S. Hacid, J. Mylopoulos, B. Pernici, O. Babaoglu, H. A. Jacobsen, J. Loyall, M. Kifer, & S. Spaccapietra (Eds.). *On the Move to Meaningful Internet Systems 2005: Coopis, Doa, and Odbase, Pt 2, Proceedings* (3761, pp. 1571–1587). Berlin: Springer-Verlag Berlin.

Gerede, E., Hull, R., Ibarra, O. H., & Su, J. (2004). Automated composition of e-services: lookaheads. In *Proceedings of the 2nd International Conference on Service Oriented Computing* (pp. 252–262). New York, NY, USA: ACM.

Gutierrez-Garcia, J. O., & Sim, K. (2013). Agent-based cloud service composition. *Applied Intelligence, 38*, 436–464.

Gutierrez-Garcia, J. O., & Sim, K. M. (2010). Agent-based service composition in cloud computing. In T. H. Kim, S. S. Yau, O. Gervasi, B. H. Kang, A. Stoica, & D. Slezak (Eds.). *Grid and Distributed Computing, Control and Automation* (121, pp. 1–10). Berlin: Springer-Verlag Berlin.

Hamdaqa, M., & Tahvildari, L. (2012). Cloud Computing Uncovered: A Research Landscape. In H. Ali & M. Atif (Eds.), *Advances in Computers* (86, pp. 41–85). Elsevier.

Haykin, S. (1998). *Neural Networks: A Comprehensive Foundation*. Prentice Hall PTR.

He, Q., Yan, J., Jin, H., & Yang, Y. (2008). Adaptation of web service composition based on workflow patterns. In A. Bouguettaya, I. Krueger, & T. Margaria (Eds.). *Service-Oriented Computing – ICSOC 2008, Proceedings* (5364, pp. 22–37). Berlin: Springer-Verlag Berlin.

Hossain, M. S., Hassan, M. M., Al Qurishi, M., & Alghamdi, A. (2012). *Resource Allocation for Service Composition in Cloud-based Video Surveillance Platform*. New York: IEEE.

Huang, C.-L., Lo, C.-C., Chao, K.-M., & Younas, M. (2006). Reaching consensus: a moderated fuzzy web services discovery method. *Information and Software Technology, 48*, 410–423.

Huang, J., Liu, Y., Yu, R., Duan, Q., & Tanaka, Y. (2013). Modeling and algorithms for QoS-aware service composition in virtualization-based cloud computing. *IEICE Transactions on Communications, E96.B*, 10–19.

Ishizaka, A., & Labib, A. (2011). Review of the main developments in the analytic hierarchy process. *Expert Systems with Applications, 38*, 14336–14345.

Jensen, K. (1995). *Coloured Petri Nets: Basic Concepts, Analysis Methods and Practical Use* (2). Springer-Verlag.

Jeonghwan, J., Rothrock, L., McDermott, P. L., & Barnes, M. (2010). Using the analytic hierarchy process to examine judgment consistency in a complex multiattribute task. *IEEE Transactions on Systems, Man, and Cybernetics Part A: Systems and Humans, 40*, 1105–1115.

Jiang, H., Kwong, C. K., Chen, Z., & Ysim, Y. C. (2012). Chaos particle swarm optimization and T-S fuzzy modeling approaches to constrained predictive control. *Expert Systems with Applications, 39*, 194–201.

Jula, A., & Naseri, N. K. (2012). A hybrid genetic algorithm-gravitational attraction search algorithm (HYGAGA) to solve grid task scheduling problem. In *In International Conference on Soft Computing and its Applications (ICSCA'2012)* (pp. 158–162). Planetary Scientific Research Center (PSRC).

Jula, A., Othman, Z., & Sundararajan, E. (2013). A hybrid imperialist competitive-gravitational attraction search algorithm to optimize cloud service composition. In *Memetic Computing (MC), 2013 IEEE Workshop on* (pp. 37–43).

Jungmann, A., & Kleinjohann, B. (2012). Towards the Application of Reinforcement Learning Techniques for Quality-Based Service Selection in Automated Service Composition. In *Services Computing (SCC), 2012 IEEE Ninth International Conference on* (pp. 701–702).

Kaelbling, L. P., Littman, M. L., & Moore, A. W. (1996). Reinforcement learning: a survey. *Journal of Artificial Intelligence Research, 4*, 237–285.

Karim, R., Chen, D., & Miri, A. (2013). An end-to-end QoS mapping approach for cloud service selection. In *Services (SERVICES), 203 IEEE Ninth World Congress on* (pp. 341–348).

Kitchenham, B., & Charters, S. (2007). *Guidelines for performing Systematic Literature Reviews in Software Engineering*. (2.3 ed., pp. 1–65). Keele University and Durham University.

Klein, A., Ishikawa, F., & Honiden, S. (2012). Towards network-aware service composition in the cloud. In the 21st international conference on World Wide Web (pp. 959-968). Lyon, France: ACM.

Knuth, D. E. (2011). The Art of Computer Programming. Volume. 4A: Combinatorial Algorithms, Part 1, Pearson Education India.

Kofler, K., Haq, I. U., & Schikuta, E. (2010). User-centric, heuristic optimization of service composition in clouds. *LNCS, 6271*, 405–417.

Kofler, K., ul Haq, I., & Schikuta, E. (2009). A parallel branch and bound algorithm for workflow QoS optimization. In *Parallel Processing, 2009. ICPP '09. International Conference on* (pp. 478–485).

Korkmaz, T., & Krunz, M. (2001). Multi-constrained optimal path selection. In *INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings* (2, pp. 834–843). IEEE.

Korte, B., & Vygen, J. (2012). *Linear Programming* (21). Berlin Heidelberg: Combinatorial Optimization Springer, pp. 51-71.

Koshy, T. (2004). Chapter 11 – formal languages and finite-state machines. In *Discrete Mathematics With Applications* (pp. 733–802). Burlington: Academic Press.

Kosuga, M., Kirimoto, N., Yamazaki, T., Nakanishi, T., Masuzaki, M., & Hasuike, K. (2002). A multimedia service composition scheme for ubiquitous networks. *Journal of Network and Computer Applications, 25*, 279–293.

Kou, L. T., & Markowsky, G. (1977). Multidimensional bin packing algorithms. *IBM Journal of Research and Development, 21*, 443–448.

Abonyi, J., Akerkar, R., Alavi, A. H., Arango, C., Aydogdu, I., Brest, J., et al. (2013). List of Contributors. In X.-S. Yang, Z. Cui, R. Xiao, A. H. Gandomi & M. Karamanoglu (Eds.), Swarm Intelligence and Bio-inspired Computation (pp. xv-xviii). Oxford: Elsevier.

Li, D., Cheung, D., Shi, X., & Ng, V. (1998). Uncertainty reasoning based on cloud models in controllers. *Computers & Mathematics with Applications, 35*, 99–123.

Li, Z., Fang, H., & Xia, L. (2014). Increasing mapping based hidden Markov model for dynamic process monitoring and diagnosis. *Expert Systems with Applications, 41*, 744–751.

Liangzhao, Z., Benatallah, B., Ngu, A. H. H., Dumas, M., Kalagnanam, J., & Chang, H. (2004). QoS-aware middleware for web services composition. *IEEE Transactions on Software Engineering, 30*, 311–327.

Liao, J. X., Liu, Y., Zhu, X. M., Xu, T., & Wang, J. Y.IEEE. (2011). Niching particle swarm optimization algorithm for service composition. In *2011 IEEE Global Telecommunications Conference*. New York: IEEE.

Lie, Q., Yan, W., & Orgun, M. A. (2013). Cloud service selection based on the aggregation of user feedback and quantitative performance assessment. In *Services Computing (SCC), 2013 IEEE International Conference on* (pp. 152–159).

Lin, W., Dou, W., Luo, X., & Jinjun, C. (2011). A history record-based service optimization method for QoS-aware service composition. In *Web Services (ICWS), 2011 IEEE International Conference on* (pp. 666–673).

Liu, F., & Zeng, G. (2009). Study of genetic algorithm with reinforcement learning to solve the TSP. *Expert Systems with Applications, 36*, 6995–7001.

Liu, H., Zheng, Z. B., Zhang, W. M., & Ren, K. J. (2011). A global graph-based approach for transaction and QoS-aware service composition. *KSII Transactions on Internet and Information Systems, 5*, 1252–1273.

Liu, M., Wang, M. R., Shen, W. M., Luo, N., & Yan, J. W. (2012). A quality of service (QoS)-aware execution plan selection approach for a service composition process. *Future Generation Computer Systems-the International Journal of Grid Computing and Escience, 28*, 1080–1089.

Liu, S., Xiong, G., Zhao, H., Dong, X., & Yao, J. (2012). Service composition execution optimization based on state transition matrix for cloud computing. In Intelligent Control and Automation (WCICA), 2012 10th World Congress on (pp. 4126-4131).

Liu, Y., Li, M., & Wang, Q. (2012). A novel user-preference-driven service selection strategy in cloud computing. *International Journal of Advancements in Computing Technology, 4*, 414–421.

Ludwig, S. A. (2012). Clonal selection based genetic algorithm for workflow service selection. In *Evolutionary Computation (CEC), 2012 IEEE Congress on* (pp. 1–7).

Luo, Y. S., Qi, Y., Hou, D., Shen, L. F., Chen, Y., & Zhong, X. (2011). A novel heuristic algorithm for QoS-aware end-to-end service composition. *Computer Communications, 34*, 1137–1144.

Milanovic, N., & Malek, M. (2004). Current solutions for web service composition. *IEEE Internet Computing, 8*, 51–59.

Moscato, P., & Cotta, C. (2003). A Gentle Introduction to Memetic Algorithms. In F. Glover & G. Kochenberger (Eds.), *Handbook of Metaheuristics* (57, pp. 105–144). US: Springer.

Moura, L. D., & Bjørner, N. (2011). Satisfiability modulo theories: introduction and applications. *Communication ACM, 54*, 69–77.

Neapolitan, R., & Naimipour, K. (2009). *Foundations of Algorithms* (4). Jones and Bartlett Publishers, Inc (December 28, 2009).

Peter Mell, T. G. (2011). The NIST Definition of Cloud Computing. In N. I. o. S. a. Technology (Ed.): U.S. Department of Commerce.

Pham, T. V., Jamjoom, H., Jordan, K., & Shae, Z.-Y. (2010). A service composition framework for market-oriented high performance computing cloud. In *Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing* (pp. 284–287). Chicago, Illinois: ACM.

Preve, N. P. (2011). *Grid Computing: Towards a Global Interconnected Infrastructure*. Springer.

Qi, Y., & Bouguettaya, A. (2013). Efficient service skyline computation for composite service selection. *IEEE Transactions on Knowledge and Data Engineering, 25*, 776–789.

Qiang, D., Yuhong, Y., & Vasilakos, A. V. (2012). A survey on service-oriented network virtualization toward convergence of networking and cloud computing. *IEEE Transactions on Network and Service Management, 9*, 373–392.

Reynolds, R. G. (1999). Cultural Algorithms: Theory and Applications. In C. David, D. Marco, G. Fred, D. Dipankar, M. Pablo, P. Riccardo, & V. P. Kenneth (Eds.), *New Ideas in Optimization* (pp. 367–378). UK: McGraw-Hill Ltd..

Richardson, L. (2007). *RESTful Web Services* (1). O'Reilly Media.

Sayed, A. H. (2003). *Fundamentals of Adaptive Filtering*. Wiley.

Schmid, S., Chart, T., Sifalakis, M., & Scott, A. (2002). Flexible, Dynamic, and Scalable Service Composition for Active Routers. In *Proceedings of the IFIP-TC6 4th International Working Conference on Active Networks* (pp. 253–266). Springer-Verlag.

Shangguang, W., Zibin, Z., Qibo, S., Hua, Z., & Fangchun, Y. (2011). Cloud model for service selection. In *Computer Communications Workshops (INFOCOM WKSHPS), 2011 IEEE Conference on* (pp. 666–671).

Shrme, A. E. (2011). Hybrid intelligent technique for automatic communication signals recognition using bees algorithm and MLP neural networks based on the efficient features. *Expert Systems with Applications, 38*, 6000–6006.

Singh, M. P. (2001). Physics of service composition. *IEEE Internet Computing, 5*, 6–7.

Sinnema, M., & Deelstra, S. (2007). Classifying variability modeling techniques. *Information and Software Technology, 49*, 717–739.

Smith, R. G. (1981). Correction to the contract net protocol: high-level communication and control in a distributed problem solver. *IEEE Transactions on Computers, C-30*, 372.

Song, X., Dou, W., & Chen, J. (2011). A workflow framework for intelligent service composition. *Future Generation Computer Systems, 27*, 627–636.

Song, X. D., Dou, W. C., & Chen, J. J. (2011). A workflow framework for intelligent service composition. *Future Generation Computer Systems-the International Journal of Grid Computing and Escience, 27*, 627–636.

Song, Z., Geng, X., Kusiak, A., & Xu, C. (2011). Mining Markov chain transition matrix from wind speed time series data. *Expert Systems with Applications, 38*, 10229–10239.

Stewart, B. J. (2009). *Leveraging Amazon Web Services for Enterprise Application Integration* (1). IBM Corporation, developerWorks, pp. 1–41.

Strunk, A. (2010). QoS-aware service composition: a survey. In *Web Services (ECOWS), 2010 IEEE 8th European Conference on* (pp. 67–74).

Sundareswaran, S., Squicciarini, A., & Lin, D. (2012). A Brokerage-Based Approach for Cloud Service Selection. In *Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on* (pp. 558–565).

Takabi, H., Joshi, J. B. D., & Gail-Joon, A. (2010). Security and privacy challenges in cloud computing environments. *IEEE Security and Privacy, 8*, 24–31.

Tang, M. L., & Ai, L. F.IEEE. (2010). *A Hybrid Genetic Algorithm for the Optimal Constrained Web Service Selection Problem in Web Service Composition*. New York: IEEE.

Torkashvan, M., & Haghighi, H. (2012). *A Greedy Approach for Service Composition*. New York: IEEE.

Vanderbei, R. J. (2008). *Linear Programming: Foundations and Extensions*. Springer London, Limited.

Vaquero, L. M., Rodero-Merino, L., Caceres, J., & Lindner, M. (2008). A break in the clouds: towards a cloud definition. *SIGCOMM Compututer Communication Review, 39*, 50–55.

Wang, J.-Z., Wang, J.-J., Zhang, Z.-G., & Guo, S.-P. (2011). Forecasting stock indices with back propagation neural network. *Expert Systems with Applications, 38*, 14346–14355.

Wang, S., Sun, Q., & Yang, F. (2010). Towards web service selection based on QoS estimation. *Internatinal Journal of Web Grid Services, 6*, 424–443.

Wang, S. G., Sun, Q. B., Zou, H., & Yang, F. C. (2013). Particle swarm optimization with skyline operator for fast cloud-based web service composition. *Mobile Networks & Applications, 18*, 116–121.

Wang, Y. W. (2009). *Application of Chaos Ant Colony Algorithm in Web Service composition based on QoS*. Los Alamitos: IEEE Computer Soc.

Wischik, D., Handley, M., & Braun, M. B. (2008). The resource pooling principle. *SIGCOMM Computer Communication Review, 38*, 47–52.

Wittern, E., Kuhlenkamp, J., & Menzel, M. (2012). Cloud service selection based on variability modeling. *LNCS, 7636*, 127–141.

Worm, D., Zivkovic, M., van den Berg, H., & van der Mei, R. (2012). Revenue maximization with quality assurance for composite web services. In *Service-Oriented Computing and Applications (SOCA), 2012 5th IEEE International Conference on* (pp. 1–9).

Wu, Q., Zhang, M., Zheng, R., Lou, Y., & Wei, W. (2013). A QoS-satisfied prediction model for cloud-service composition based on a hidden Markov model. *Mathematical Problems in Engineering, 2013*, 7.

Xiangkun, T., & Yi, G. (2010). A cosine theorem based algorithm for similarity aggregation of ontologies. In *Signal Processing Systems (ICSPS), 2010 2nd International Conference on* Vol. 2 (pp. V2–16-V12-19).

Xiaona, W., Bixin, L., Rui, S., Cuicui, L., & Shanshan, Q. (2012). Trust-Based Service Composition and Optimization. In *Software Engineering Conference (APSEC), 2012 19th Asia-Pacific* Vol. 1 (pp. 67–72).

Yakimenko, O. A., Slegers, N. J., Bourakov, E. A., Hewgley, C. W., Bordetsky, A. B., Jensen, R. P., Robinson, A. B., Malone, J. R., & Heidt, P. E. (2009). Mobile system for precise aero delivery with global reach network capability. In *Control and Automation, 2009. ICCA 2009, IEEE International Conference on* (pp. 1394–1398).

Yang, Y., Mi, Z., & Sun, J. (2012). Game theory based IaaS services composition in cloud computing environment. *Advances in Information Sciences and Service Sciences, 4*, 238–246.

Ye, Z., Zhou, X., & Bouguettaya, A. (2011). Genetic algorithm based QoS-aware service compositions in cloud computing. In J. Yu, M. Kim, & R. Unland (Eds.). *Database Systems for Advanced Applications* (6588, pp. 321–334). Berlin Heidelberg: Springer.

Yi, W., & Blake, M. B. (2010). Service-oriented computing and cloud computing: challenges and opportunities. *IEEE Internet Computing, 14*, 72–75.

Yu, Q., & Bouguettaya, A. (2010). Computing service skyline from uncertain QoWS. *IEEE Transactions on Services Computing, 3*, 16–29.

Yu, T., & Lin, K.-J. (2005). Service selection algorithms for composing complex services with multiple qos constraints. In *Proceedings of the Third International Conference on Service-Oriented Computing* (pp. 130–143). Amsterdam, The Netherlands: Springer-Verlag.

Zeng, C., Guo, X. A., Ou, W. J., & Han, D. (2009). Cloud computing service composition and search based on semantic. In M. G. Jaatun, G. Zhao, & C. Rong (Eds.). *Cloud Computing, Proceedings* (5931, pp. 290–300). Berlin: Springer-Verlag Berlin.

Zeng, J., Duan, J., & Wu, C. (2010). A new distance measure for hidden Markov models. *Expert Systems with Applications, 37*, 1550–1555.

Zhang, M., Ranjan, R., Nepal, S., Menzel, M., & Haller, A. (2012). A declarative recommender system for cloud infrastructure services selection. In *Proceedings of the 9th International Conference on Economics of Grids, Clouds, Systems, and Services* (pp. 102–113). Berlin, Germany: Springer-Verlag.

Zhang, X. Y., & Dou, W. C. (2010). Preference-aware QoS evaluation for cloud web service composition based on artificial neural networks. In F. L. Wang, Z. G. Gong, X. F. Luo, & J. S. Lei (Eds.). *Web Information Systems and Mining* (6318, pp. 410–417). Berlin: Springer-Verlag Berlin.

Zhao, S. S., Ma, L., Wang, L., & Wen, Z. P. (2012). *An improved ant colony optimization algorithm for QoS-aware dynamic web service composition*. Los Alamitos: IEEE Computer Soc..

Zhao, X., Wen, Z., & Li, X. (2013). QoS-aware web service selection with negative selection algorithm. *Knowledge and Information Systems*, 1–25.

Zhao, X. C., Song, B. Q., Huang, P. Y., Wen, Z. C., Weng, J. L., & Fan, Y. (2012). An improved discrete immune optimization algorithm based on PSO for QoS-driven web service composition. *Applied Soft Computing, 12*, 2208–2216.

Zhou, W., & Bovik, A. C. (2009). Mean squared error: love it or leave it? A new look at signal fidelity measures. *IEEE Signal Processing Magazine, 26*, 98–117.

Zhou, X., & Mao, F. (2012). A Semantics Web Service Composition Approach Based on Cloud Computing. In Computational and Information Sciences (ICCIS), 2012 Fourth International Conference on (pp. 807–810).

Zhu, Y., Li, W., Luo, J., & Zheng, X. (2012). A novel two-phase approach for QoS-aware service composition based on history records. In *Service-Oriented Computing and Applications (SOCA), 2012 5th IEEE International Conference on* (pp. 1–8).

Zibin, Z., Xinmiao, W., Yilei, Z., Lyu, M. R., & Jianmin, W. (2013). QoS ranking prediction for cloud services. *IEEE Transactions on Parallel and Distributed Systems, 24*, 1213–1222.

Zibin, Z., Yilei, Z., & Lyu, M. R. (2010). Distributed QoS Evaluation for Real-World Web Services. In *Web Services (ICWS), 2010 IEEE International Conference on* (pp. 83–90).

Zissis, D., & Lekkas, D. (2012). Addressing cloud computing security issues. *Future Generation Computer Systems, 28*, 583–592.