QoS-Aware Service Composition for Converged Network–Cloud Service Provisioning

Jun Huang, Guoquan Liu
School of Commun. & Info. Eng.
Chongqing Univ. of Posts and Telecom.
Chongqing, China 400065
Email: xiaoniuadmin@gmail.com

Qiang Duan
Info. Sci. & Tech. Department
The Pennsylvania State University
Abington, Pennsylvania 19001
Email: qduan@psu.edu

Yuhong Yan
Dept. of Computer Sci. and Software Eng.
Concordia University
Montreal, Canada
Email: yuhong@encs.concordia.ca

***Abstract*—The crucial role of networking in Cloud computing calls for federated management of both computing and networking resources for end-to-end service provisioning. Application of the Service-Oriented Architecture (SOA) in both Cloud computing and networking enables a convergence of network and Cloud service provisioning. One of the key challenges to network–Cloud convergence lies in QoS-aware composition of network and Cloud services. In this paper, we propose a QoS-aware service composition method to tackle this challenging issue. We first present a system model for network–Cloud service composition and formulate the service composition problem as a variant of Multi-Constrained Optimal Path (MCOP) problem. We then develop an algorithm to solve the problem and give theoretical analysis on properties of the algorithm to show its effectiveness and efficiency for QoS-aware network-Cloud service composition. Performance of the proposed algorithm is evaluated through extensive simulation experiments and the obtained results indicate that the proposed method achieves better performance in service composition than the best currently available MCOP approach.**

***Keywords*-Cloud computing; Service-Oriented Architecture (SOA); Network-as-a-Service (NaaS); service composition; Quality-of-Service (QoS)**

I. Introduction

Networking plays a crucial role in Cloud computing. Users access Cloud services through computer networks. Services received by end users consist of not only computing functions provided by Cloud infrastructure but also communication functions offered by the Internet. Recent research has indicated that networking performance has a significant impact on the quality of Cloud services provided to end users [1]. The important role of networking in Cloud computing calls for federated management of both computing and networking resources for service provisioning in a Cloud environment.

Consider an example that a research lab generates a large amount of data to be stored and processed in Cloud. The required Cloud services include a storage service (e.g. Amazon S3) and a computing service (e.g. Amazon EC2). The service provided to the lab must also include network services for data communications between the lab and Cloud services. Suppose the total data set is 100 GB and Cloud storage/computing service can process 200 GB data per hour, then total Cloud service time is 30 minutes. However, if the lab uses a network service with 200 Mb/s throughput, then even the single-trip delay of data transmission will be more than 1 hour. This example illustrates that services provided to Cloud users are always composite network-Cloud services and network performance has a significant impact on Cloud service quality.

Recently the Service-Oriented Architecture (SOA) has also been applied in networking in order to address the challenge of rapidly developing and deploying new network functions [2]. Applying SOA in networking leads to the *Network-as-a-Service (NaaS)* paradigm, through which functionalities of various networking systems can be exposed and accessed as services in a Cloud environment. Therefore, the SOA principle employed in both networking and Cloud computing provides a promising approach to converged network–Cloud service provisioning [3].

To realize the notion of converged network–Cloud service provisioning, one of the key challenges lies in the QoS-aware service composition across the networking and computing domains; that is, composing an appropriate sequence of network and Cloud services to meet end-to-end service performance requirements while optimizing networking and computing resource utilization. For example in the above lab data processing case, if the user requires a maximum service delay (waiting time from sending data out to receiving results back), then the services for storage, computing, and networking must be selected with a holistic vision to guarantee the end-to-end service delay. A composite service for this example is illustrated in Figure 1. In this figure Cloud services 1 and 2 are respectively Amazon S3 for storage and EC2 for computing. Network services 1, 2, and 3 respectively provide data communications from the user to Cloud service 1, between Cloud services 1 and 2, and from Cloud service 2 back to the user. Selection of these Cloud and network services must consider their delay performance in order to guarantee that the total data transmission, storage, and processing delay meets user's requirement.

Although service composition has been extensively stud-

978-1-4799-5066-9/14 $31.00 © 2014 IEEE
DOI 10.1109/SCC.2014.18

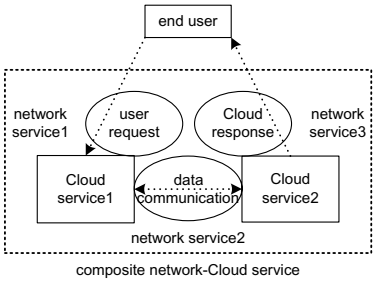


Figure 1. An example of composite network–Cloud service provisioning.

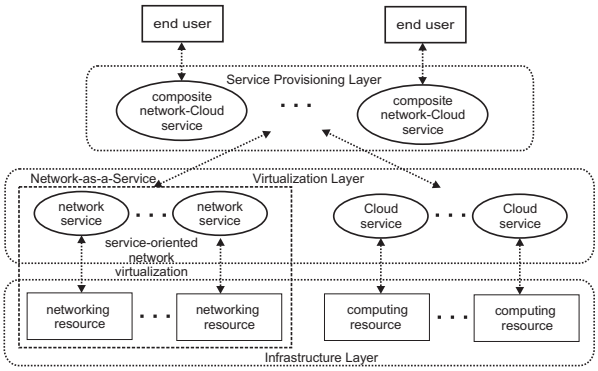


Figure 2. The layered structure of a service-oriented framework for network–Cloud service convergence.

ied, the currently available approaches mainly focus on computing services instead of converged network–Cloud services; therefore may not achieve optimal end-to-end QoS across both networking and computing domains. In addition, due to the diverse QoS parameters for network and Cloud services, composition across network and Cloud domains must be able to handle a wide variety of QoS constraints. Such a problem can be formulated as the Multi-Constrained Optimal Path (MCOP) problem, which is known to be NP-hard. Network services, due to the real-time requirements of various network protocols, typically require much faster composition algorithms to achieve shorter response time. In contrast, Web/Cloud service composition, although might have QoS requirement on composite services, typically allows best-effort response time of composition algorithms. Therefore, service composition across network and Cloud domains particularly requires more efficient and effective MCOP algorithms that can not only meet diverse QoS constraints but also achieve much shorter response time.

Toward addressing this challenging issue, this paper makes the following contributions. We propose a model for converged network–Cloud service provisioning and formulate the problem of QoS-aware network–Cloud service composition as a variant of MCOP. We develop an approximation algorithm to solve the problem of QoS-aware service composition and analyze the theoretical properties of the proposed algorithm to show it is effective and resilient. We also evaluate the proposed algorithm through numerical experiments and find that it outperforms the best currently available MCOP algorithm in terms of efficiency and accuracy for QoS-aware service selection.

The rest of this paper is organized as follows. In Section II we first introduce a service-oriented framework for network–Cloud service convergence and discuss related research progress on service composition. In Section III we present a model for network–Cloud service composition and formulate the QoS-aware service composition problem. Section IV presents an approximation algorithm to solve the service composition problem and gives theoretical analysis on the proposed algorithm. In Section V we evaluate performance of the proposed algorithm and compare it with currently available algorithms through numerical experiments. Section VI draws conclusions.

II. Convergence of Network and Cloud Service Provisioning

A. A Framework for Network–Cloud Service Convergence

A service-oriented framework for network–Cloud service convergence was first proposed in [3] and is presented here again for completeness. As shown in Figure 2, at the bottom of this framework is an *infrastructure layer* consisting of physical infrastructure for both networking and computing. Above the infrastructure layer is a *virtualization layer* in which both networking and computing resources are virtualized and abstracted as SOA-complaint services. The *service provisioning layer* discovers and selects both network and Cloud services and orchestrates them into composite network–Cloud services that match the requirements of end users. In this framework the SOA-based NaaS paradigm enables networking resources to be virtualized and exposed as services and composed with Cloud computing resources into composite network–Cloud services.

For the lab data processing example, this convergence framework enables networking systems of various providers, such as Verizon, AT&T, and Comcast, to be virtualized and offered as SOA-compliant network services through the NaaS paradigm, just like computing and storage resources in Cloud to be offered through the IaaS paradigm (e.g. Amazon EC2 and S3 services). Then appropriate network services as well as Cloud services are selected and composed for meeting the performance requirement for lab data processing.

B. Related Work on Network–Cloud Service Composition

Service composition has been extensively studied in the fields of Web services and Cloud computing. Various technologies have been developed to achieve service composition that meets functional and/or performance requirements. Most of these technologies are based on either workflow

management or AI-planning approach. Surveys of recent research on Web service composition can be found in [4] and [5]. Optimization of multiple QoS criteria in service composition has been modeled as a multi-dimension multi-choice 0-1 knapsack problem and solved by integer programming [6] or by heuristic search methods [7]. Genetic algorithms offer another approach to addressing this problem with the advantage of being able to balance the optimization of latency and other QoS metrics [8]. In [9] Cloud services are modeled as finite state machines and a simple additive weighting technique is used to select an optimal service path. A low latency service selection algorithm under various replica limitation constraints is developed in [10]. More research on QoS-aware Web service composition can be found in the survey presented in [11].

The aforementioned research results focus on Web/Cloud service composition instead of composition between network and Cloud services. The modeling and optimization approaches proposed in this paper overcome this limit by employing QoS metrics of both network and Cloud services through NaaS paradigm in the composition algorithm; thus finding a service path that achieves optimal end-to-end QoS performance. Network-aware service composition in Cloud is studied in [12]. The authors built a model for estimating network latency and developed an algorithm leveraging this model to select composite services with the minimum service delay. However QoS constraints of network and Cloud services are handled separately in [12]. In contrast, we address the service composition problem with a holistic vision of both computing and networking resources; therefore the proposed algorithm integrates network and Cloud QoS constraints and achieves optimal composite network–Cloud services.

The problem of combined service composition and network routing is studied in [13]. A decision making system was developed to solve this problem with AI-planning techniques. This work is similar to our research in that service composition and network routing are combined into one problem. However, only a single QoS metric (delay) was considered in [13]. In our work, by formulating network–Cloud service composition as a multi-constraint optimal path problem we develop an algorithm that can handle multiple QoS constraints.

Some recent research applied QoS routing techniques, typically Multi-Constraint Optimal Path (MCOP) algorithms, to tackle the service composition problem. Up to date, much progress has been made toward designing efficient MCOP algorithms for QoS routing [14], [15], [16], [17], [18]. Development of the aforementioned algorithms to enable their applications in network–Cloud service composition forms an interesting research topic. An exact MCOP algorithm developed for QoS-aware network–Cloud service composition is preliminarily reported in [19]. Since approximation algorithms are typically approaches to address NP-complexity of the MCOP problem, this paper focuses on using approximation algorithm to handle service composition. In [20] Huang and Tanaka proposed a Fully Polynomial Time Approximation Scheme (FPTAS) that has been proved to be currently one of the fastest algorithms for MCOP. However, FPTAS cannot be reused directly in the network–Cloud service composition context. In this work, we propose an approximation algorithm to resolve the problem of QoS-aware composite network–Cloud service composition by following the design principle of FPTAS.

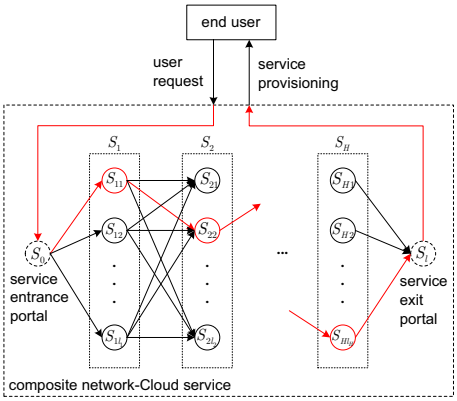


Figure 3. Modeling for network–Cloud service composition.

III. Modeling QoS-Aware Network-Cloud Service Composition

A typical end-to-end provisioning system for composite network–Cloud services consists of both computing infrastructure that offers Cloud services and network services for Cloud access and inter-Cloud communications. If the end user wants a certain level of performance guarantee from the composite service provisioning, the end user must expect a certain level QoS from each network and Cloud service. In general, such QoS expectation can be defined in Service Level Agreements (SLAs) between the user and service providers. When a service request is submitted by a user to the service provisioning system, a service broker will invoke a service composition process. The broker selects and composes a series of atomic services to form a composite service, referred in this paper as a service path, that consists of a sequence of selected services.

Figure 3 shows a model for network–Cloud service composition. The end-to-end service has H service components, S_i, $i = 1, 2, \cdots, H$. The number and sequence of service components are decided by the user's request. Each component could be a Cloud service or a network service virtualized from network infrastructure through the NaaS paradigm. Each service component S_i has l_i atomic candidate services S_{ij}, $j = 1, 2, \cdots, l_i$. Note that candidate services for each service component are pre-selected based on the functional requirements for the component specified

in the user request and service descriptions published by available network and Cloud services. For each service request, the composition process finds a service path from the service entrance portal to the service exit portal that traverses the selected candidate service of each component that meets the QoS requirements for the composite service. For example the path $S_0 \rightarrow S_{11} \rightarrow S_{22} \rightarrow \cdots \rightarrow S_{Hl_H} \rightarrow S_l$ colored in red in Figure 3 is a composite service that consists of H service components. The presented model allows network–Cloud service composition to take QoS requirements into account naturally. Therefore, selecting optimal service pathes from a pool of network and Cloud services while satisfying various QoS requirements is the main optimization focus for converged network–Cloud service provisioning.

Still considering the lab data processing case, the composite network–Cloud service shown in Figure 1 have 3 network and 2 Cloud service components. A variety of network services, for example services provided by AT&T, Verizon, and Comcast, are candidates for realizing the 3 network service components. Amazon S3, Google drive, and Microsoft Skydrive are candidate services for Cloud service 1 (the storage service). Amazon EC2 and Google compute engine are candidate services for Cloud service 2 (the computing service). The composition procedure considers available capacities and delay performance of these candidate services and selects a service path to achieve the minimum delay with sufficient service capacity. For example, Verizon network for network service 1 (transmitting data from the user to storage), S3 for Cloud service 1, AT&T network for network service 2 (data communications between storage and computing), EC2 for Cloud service 2, and Comcast network for network service 3 (transmitting results back to the user).

Given this converged service provisioning model, we can formulate the problem of QoS-aware network–Cloud service composition as follows.

A service network formed by interconnecting service components can be modeled as a directed graph $G(V, E)$ with n vertices and m edges. Each edge $e \in E$ is associated with K weights $w = (w_1, w_2, \cdots, w_K)$ representing QoS parameters. Let $W = (W_1, W_2, \cdots, W_K)$ denotes the K end-to-end QoS constraints, then a series of selected services is represented as a path p in the service network.

Note that atomic components in a composite service can be executed either in a sequential order or in other orders, including parallel, conditional, and loop flow structures. The typical interconnect relationship between atomic services are Sequential, AND split (fork), XOR split (conditional), Loop, AND join (Merge), and XOR join (Trigger). It has been shown in [21] that service selection in a function graph with general flow structures can be eventually transformed to selection of service components with sequential order in a directed graph. Therefore, we focus on selecting service components to form a sequential service path that meets end-to-end QoS requirements. In the above model a service path p denotes a set of tandem atomic service components. QoS parameters can be categorized to be either *positive* (quality increases as parameter value increases, e.g., reliability) or *negative* (quality decreases as parameter value increases, e.g., delay). In this model we assume all QoS parameters are negative and concentrate on the *additive* QoS parameters.

Definition 1: Feasible Service Composition. A composed service, i.e., a path p in the service network, is said to be feasible if $\forall v \in p$, $w_k(p) \leq W_k$ for all $1 \leq k \leq K$.

Denote $\{p^f\}$ as all feasible service compositions in $G(V, E)$, to each $p_i^f \in \{p^f\}$, there exists a smallest value $\eta_i \in (0, 1]$ such that $w_k(p_i^f) \leq \eta_i \cdot W_k$, $1 \leq k \leq K$ respectively.

Problem 1: QoS-aware Service Composition (QSC). QSC is to find an optimal composition of services p^{opt} among feasible service compositions in $G(V, E)$ and the corresponding smallest value η^{opt} among all η_i such that $w_k(p^{opt}) \leq \eta^{opt} \cdot W_k$, $k \in [1, K]$ where $K \geq 2$.

It can be proved that QSC problem is NP-hard since solution of QSC maps that of MCOP directly [19]. In order to solve QSC, we propose an approximation algorithm based on our previous work on MCOP. Approximation algorithm is a powerful tool to address NP-hard problem. Formally, an algorithm is a β-approximation algorithm (or simply, an approximation algorithm) for QSC if the algorithm generates service path p such that $w_k(p) \leq \beta \cdot \eta^{opt} \cdot W_k$, and the running time of the algorithm is bounded by a polynomial in the input size of the instance as well as in $1/\beta$.

IV. An Approximation Algorithm for QoS-Aware Network and Cloud Service Composition

We propose a Service Composition Approximation (SCA) algorithm (shown in Algorithm 1) to solve the QSC problem. The proposed algorithm includes the following three steps.

The first step (line 1) transforms the original service network graph to a simpler graph in order to make the QSC problem solvable.

In the second step (lines 2 to 14) we assume that nodes s and t denote the service entrance portal and the service exit portal respectively. The notion $d^v[\delta_2, \cdots, \delta_K]$ represents the least first weight, namely, w_1 among path p from s to v such that $w_k(p) \leq \delta_k, 2 \leq k \leq K$. $p^v[\delta_2, \cdots, \delta_K]$ is used to record the predecessor of v on the path p such that $w_1(p) = d^v[\delta_2, \cdots, \delta_K]$ and $w_k(p) \leq \delta_k, 2 \leq k \leq K$. It is worth mentioning that this step is a dynamic programming process for tackling three or more constraints, which follows the same philosophy of graph-extending process in [20].

The third step examines the feasibility of the path generated by the second step. If the path satisfies $w_k(p) \leq W_k$, $2 \leq k \leq K$, the algorithm completes with this path as an output; otherwise the algorithm terminates with no feasible path.

Now we show some theoretical properties of SCA.

Algorithm 1: SCA

Input: Graph $G(V, E, w, W, H)$, Parameter ϵ.
Output: Path p.

To each $e \in E$ in $G(V, E)$, compute the new weights $w_k^N(e) = \left\lfloor \frac{w_k(e)}{W_k} \cdot \frac{H-1}{\epsilon} \right\rfloor$, $2 \le k \le K$, and set $W_1^N = \cdots = W_K^N = \Delta = \left\lfloor \frac{H+1}{\epsilon} \right\rfloor$;
for $\delta_k = 0$ *to* Δ, $2 \le k \le K$ **do**
  $d^v[\delta_2, \cdots, \delta_K] \leftarrow \infty$;
  $p^v[\delta_2, \cdots, \delta_K] \leftarrow \text{NULL}, \forall v \in V$;
  $d^s[\delta_2, \cdots, \delta_K] \leftarrow 0$;
end
for *all* $(\delta_2, \cdots, \delta_K) \in \{0, 1, \cdots, \Delta\}^{K-1}$ *in increasing lexicographic order* **do**
  for *each* $(u, v) \in E$ *s.t.* $\lambda_k \triangleq \delta_k - w_k(u, v) \ge 0$ **do**
    if $d^v[\delta_2, \cdots, \delta_K] > d^u[\lambda_2, \cdots, \lambda_K] + w_1(u, v)$ **then**
      $d^v[\delta_2, \cdots, \delta_K] \leftarrow d^u[\lambda_2, \cdots, \lambda_K] + w_1(u, v)$;
      $p^v[\delta_2, \cdots, \delta_K] \leftarrow u$;
    end
    if $d^v[\delta_2, \cdots, \delta_K] > d^v[\delta_2, \cdots, \delta_j - 1, \cdots, \delta_K]$ **then**
      $d^v[\delta_2, \cdots, \delta_K] \leftarrow d^v[\delta_2, \cdots, \delta_j - 1, \cdots, \delta_K]$;
      $p^v[\delta_2, \cdots, \delta_K] \leftarrow p^v[\delta_2, \cdots, \delta_j - 1, \cdots, \delta_K]$, $2 \le j \le K$;
    end
  end
end
if $d^t[\delta_2, \cdots, \delta_K] \le D$ **then**
  Find a source-destination path p s.t. $w_1(p) \le D$ and $w_k(p) \le \delta_k$, $2 \le k \le K$;
end
if $w_k(p) \le W_k$, $2 \le k \le K$ **then**
  return path p;
else
  return No feasible path, EXIT;
end

Theorem 1: The worst-case time complexity of Algorithm SCA is $O\left(m\left(\frac{H}{\epsilon}\right)^{K-1}\right)$ time.

Proof: The first step of SCA for pruning the topology spends $O(n + m)$ time. The second step takes $O\left(m\left(\frac{H+1}{\epsilon}\right)^{K-1}\right)$ time in the worst-case to calculate the path. Checking the feasibility of the obtained path in the third step spends $O(K)$ time. Therefore the worst-case time complexity of SCA is $O\left(m + n + m\left(\frac{H+1}{\epsilon}\right)^{K-1} + K\right) = O\left(m\left(\frac{H}{\epsilon}\right)^{K-1}\right)$. ■

Theorem 2 indicates that the proposed SCA is faster than the best currently avaiable MCOP algorithm, i.e., FPTAS [20]. This is due to the fact that the number of hops of the final path is known in the context of service composition, which guides the SCA to find the path in a smaller searching space. In addition, the space overhead of SCA is also much smaller than that of FPTAS. Since the FPTAS would extend the original graph and the scale of the extended graph increases exponentially with the number of constraints, FPTAS needs more space to store the extended graph. Leveraging algebraical form of graph extending, SCA dose not need much storage space for the extended graph. Hence, SCA is more efficient than FPTAS for solving the QSC problem.

Next, we show the theoretical properties of path p obtained by SCA.

Theorem 2: The path p obtained by SCA satisfies $w_k(p) \le (1 + \rho) \cdot \eta^{opt} \cdot W_k$, $2 \le k \le K$, where $\rho = \frac{\epsilon}{\eta^{opt}}$ is an approximation ratio of SCA.

Proof: Skipped due to the space limitation. ■

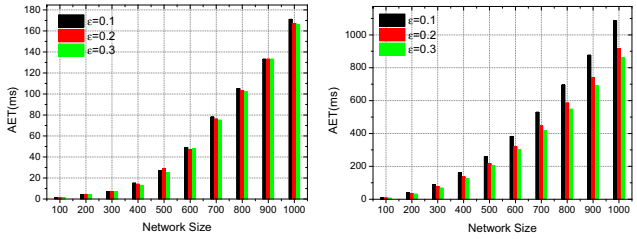
Theorem 3 implies that the later $(K - 1)$ weights of p is able to asymptotically approximate the optimal with approximation ratio $\rho = \frac{\epsilon}{\eta^{opt}}$ while the first weight of p achieves the smallest. This shows that the proposed SCA algorithm provides a provably performance guarantee; thus is effective and efficient for solving the QSC problem.

V. Performance Evaluation

A. Evaluation Settings

We implemented a simulator to evaluate the performance of the proposed algorithm. The simulator is publicly available at [22]. we compared the performance of SCA against that of a variant of ADAPT [14] as well as FPTAS [20] through numerical experiments. As we know, ADAPT and FPTAS [20] are the currently best-known algorithms for DCLC and MCOP problems. Moreover, QSC maps to a special case of MCOP directly when the topology is pruned in advance to satisfy the nodes' constraints. Therefore, ADAPT and FPTAS with minor modification can be applied for resolving QSC. The modification details of ADAPT and FPTAS in solving QSC are skipped here due to space limit and we still use their original names for the variants here for comparison.

In the first set of evaluations, we generated a set of random service networks with node numbers ranged from 100 to 1000. Each link in the generated networks has two QoS parameters and they are uniformly distributed. As for the second set of evaluations, we generated three networks with 60, 80, 100 nodes respectively since a larger network scale could overflow in FPTAS. Each link in these three networks has three QoS parameters and they are also uniformly distributed. Note that the uniform distribution used here are only for illustration, other distributions can also be worked in a similar way. We assume that the service components are spread in five service categories, i.e., $H = 5$, which we believe is reasonable for typical Cloud service scenarios. In order to evaluate the performance under different parameter



(a) SCA (b) ADAPT

Figure 4. AET comparisons for SCA and ADAPT with different network sizes and ϵ values.

configurations, we set $\epsilon \in [0.1, 0.3]$ for both SCA and ADAPT. Also, we assume that the requests are always feasible; that is, $W_1 = 50$, $W_2 \in [2.5 \times 10^{-5}, 5 \times 10^{-5}]$. The reported results were obtained by running both algorithms 100 times independently, and all of the experiments were conducted on IBM P4 2.6GHz with 2G memory space and Linux system.

B. Performance Metric

The performance metrics that we use in this section for evaluating algorithms are defined as follows.

Definition 2: Average Execution Time (AET). AET is the average execution time of an algorithm calculated from 100 independent runs. This metric is used for evaluating the time cost performance of an algorithm that reflects the time that a user has to wait for getting a service composition result.

Definition 3: Path Weights Distance (PWD). For a path p returned by an algorithm,

$$PWD(p) = \sqrt{\sum_{k=1}^{K} \left(1 - \frac{w_k(p)}{W_k}\right)^2}.$$

PWD denotes the distance between the weights of a found service path and the QoS constraints specified in the service request. This metric reflects the level of QoS guarantee that can be achieved by a service composition algorithm to end users.

C. Evaluation Results

Our experiment results quantify performance of evaluated algorithms in two ways. First, we test the impact of parameter ϵ variation on algorithms' performance by setting constant $W_2 = 3.0 \times 10^{-5}$. Second, we examine the performance of algorithms with constant $\epsilon = 0.1$ for different W_2 settings.

Figure 4 gives AET for SCA (shown in Figure 4(a)) and ADAPT (shown in Figure 4(b)) with different ϵ settings. The graphs in this figure show that AET increases with the network size, which is intuitive because both algorithms consume more time for larger scale networks. In addition, we observed that the AET values of both algorithms increase when the parameter ϵ decreases. This is because a smaller ϵ gives a larger path searching space that slows down both algorithms.

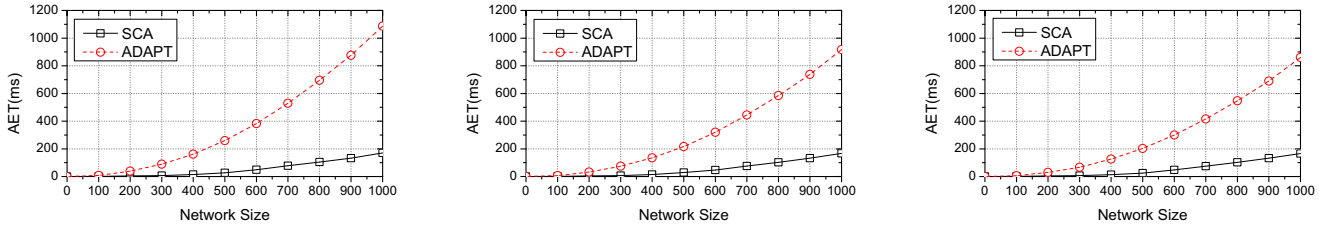
Figure 5 compares the AET of SCA against that of ADAPT with different ϵ values, where Figure 5(a), Figure 5(b), and Figure 5(c) plot the case when $\epsilon = 0.1$, $\epsilon = 0.2$, and $\epsilon = 0.3$, respectively. The curves show that for a certain ϵ value, SCA has smaller AET than ADAPT does, especially in large scale networks; that is, SCA runs much faster than ADAPT. This is because the search space of SCA relies on the number of hops in path, which is very small and can be known in advance. ADAPT employs an approximation test procedure to generate a set of upper and lower bounds for searching the optimal. The search space for ADAPT is determined by the distance between the upper and lower bounds, which is larger compared to SCA. Therefore, SCA has better AET performance than ADAPT.

The AETs (in millisecond) of SCA and ADAPT with different ϵ values for three network sizes, namely, 100, 500, 1000, are shown in Figure 6. From this figure we can see that regardless of network scale, SCA runs faster than ADAPT, which confirms our observation from Figure 6. We also observed that SCA has stable AET values for different ϵ while the AET values of ADAPT drop when ϵ increases. That means that SCA is insensitive to ϵ variation but ADAPT is not. Essentially the parameter ϵ plays a vital role in ADAPT for narrowing down the gap between upper and lower bounds, which makes the algorithm sensitive to ϵ variation.

Figure 7 shows the AET the relation between the values of SCA and ADAPT and the constraint W_2 for three specific network sizes. The results show that the AET of ADAPT is impacted by not only the ϵ but also the constraint. It can be seen from Figure 7(a) that ADAPT AET becomes low when W_2 is loose. This is due to the fact that a looser W_2 makes the search space of ADAPT small; thus reducing its AET. On the other hand, we observed that AET of SCA stays in a relatively constant level, which matches our theoretical analysis result that indicates the running time is not affected by any constraint.

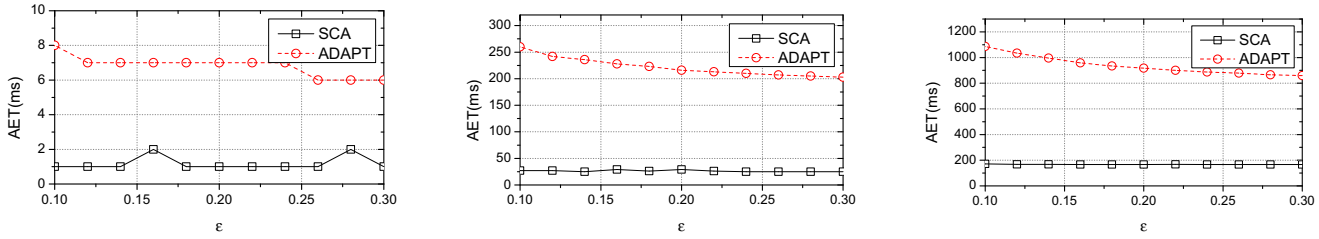
One should be noticed that in the above AET comparisons the paths found by SCA and ADAPT are the same for each specific settings of parameters; therefore the PWD of both algorithms are identical. In other words, we compares the AET under the condition that both of algorithms find the same path. Next, we examine the PWD of our proposed algorithm with different constraints and network sizes.

Figure 8 presents PWD comparisons for SCA with different constraint values and network sizes. Figure 8(a) gives the PWD results for different W_2 for three network sizes (NS=100, 500, 1000). Figure 8(b) shows the PWD for different network scales. These two figures indicate that SCA would find a path with possibly greater PWD for large scale networks. Moreover, it can be seen that the PWD of SCA



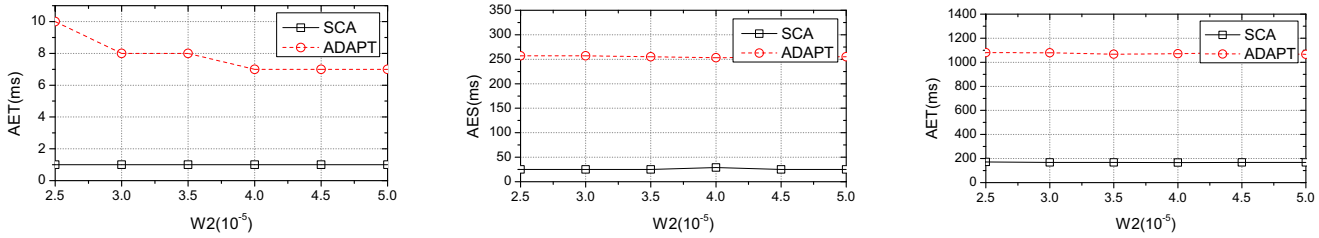
(a) $\epsilon = 0.1$ (b) $\epsilon = 0.2$ (c) $\epsilon = 0.3$

Figure 5. AET values of SCA and ADAPT for various network scales with different ϵ.



(a) $Network\ \ size = 100$ (b) $Network\ \ size = 500$ (c) $Network\ \ size = 1000$

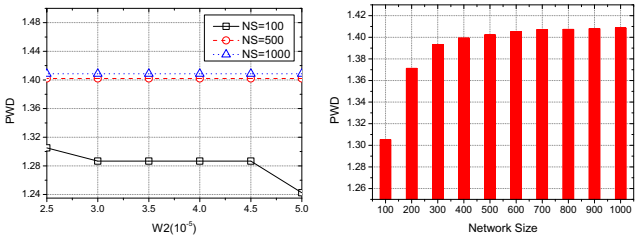
Figure 6. AETs of SCA and ADAPT for various ϵ values in different network sizes.



(a) $Network\ \ size = 100$ (b) $Network\ \ size = 500$ (c) $Network\ \ size = 1000$

Figure 7. AETs of SCA and ADAPT for various constraint values in different network sizes.

changes little in the condition of loosing W_2. The reason for this is that SCA searches the path primarily based on the path hops. If the path in the right hops that satisfies the constraints, the SCA terminates immediately. Therefore, variations in W_2 have little impact on SCA performance.



(a) (b)

Figure 8. PWDs for SCA with different constraint values and network sizes.

Table I and Table II give the data about AET comparisons for three networks with different ϵ and constraint values. Likewise, the same insight can be obtained; that is, for the same ϵ and network size, SCA executes faster than FPTAS

Table I
AET COMPARISONS FOR THREE NETWORKS WITH DIFFERENT ϵ.

Nodes	$\epsilon = 0.1$		$\epsilon = 0.2$	
	SCA	FPTAS	SCA	FPTAS
60	0.156	0.140	94	32
80	0.281	0.250	78	32
100	0.438	0.375	93	47

while finding the same path. The loosing of constraints and variations in ϵ values almost have no impact on the AET of SCA. In contrast, variations of constraint and ϵ values would affect the performance of FPTAS. Therefore, SCA outperforms FPTAS in terms of AET in the condition of achieving identical PWD.

VI. Conclusions

QoS-aware composition across network and Cloud services is a key technology for realizing converged network–Cloud service provisioning. In this paper, we have addressed this challenging and important topic by formulating QoS-

aware network–Cloud service composition as a variant of Multi-Constrained Optimal Path (MCOP) problem and developed an approximation algorithm for solving the problem. We have given theoretical analysis on properties of the proposed algorithm and also conducted thorough numerical experiments for performance evaluation. Both analytical and experimental results show that our algorithm is effective for meeting multiple QoS constraints and efficient to achieve much shorter response time for network–Cloud service composition.

ACKNOWLEDGMENT

This work is supported by NSFC (Grant No. 61309031), Program for Innovation Team Building at Institutions of Higher Education in Chongqing (Grant No. KJTD201310), Natural Science Foundation of Chongqing, (Grant No. cstc2013jcyjA40026), Scientific and Technological Research Program of Chongqing Municipal Education Commission (Grant No. KJ130523), and CQUPT Research Fund for Young Scholars (Grant No. A2012-79).

REFERENCES

[1] K. R. Jackson, K. Muriki, S. Canon, S. Cholia, and J. Shalf, "Performance analysis of high performance computing applications on the Amazon Web services Cloud," in *Proc. of the 2010 IEEE International Conference on Cloud Computing Technology and Science*, Nov. 2010, pp. 159–168.

[2] T. Magedanz, N. Blum, and S. Dutkowski, "Evolution of SOA concepts in telecommunications," *IEEE Computer Magazine*, vol. 40, no. 11, pp. 46–50, 2007.

[3] Q. Duan, "Modeling and performance analysis on network virtualization for composite network-cloud service provisioning," in *Proc. of 2011 IEEE World Congress on Services*, Aug. 2011, pp. 548–555.

[4] J. Rao and X. Su, "A survey of automated Web service composition methods," in *Proc. of the 1st International Workshop on Semantic Web Services and Web Process Composition*, Jul. 2004.

[5] S. Dustdar and W. Schreiner, "A survey on Web services composition," *International Journal of Web and Grid Services*, vol. 1, no. 1, pp. 1–30, 2005.

[6] L. Zeng, B. Benatallah, A. H. H. Ngu, M. Dumas, J. Kalagnanam, and H. Chang, "QoS-aware middleware for Web services composition," *IEEE Transaction on Software Engineering*, vol. 30, no. 5, 2004.

[7] T. Yu, Y. Zhang, and K. Lin, "Efficient algorithms for Web services selection with end-to-end QoS constraints," *ACM Transaction on Web*, vol. 1, no. 1, 2007.

[8] A. Klein, I. Fuyuki, and S. Honiden, "Sanga: A self-adaptive network-aware approach to service composition," *IEEE Transactions on Services Computing*, vol. PP, no. 99, pp. 1–1, 2013.

[9] H. Bao and W. Dou, "A QoS-aware service selection method for cloud service composition," in *Proc. of the IEEE 26th International Parallel and Distributed Processing Symposium*, May 2012, pp. 2254–2261.

[10] X. Li, J. Wu, and S. Lu, "Qos-aware service selection in geographically distributed clouds," in *Proc. of 2013 22nd International Conference on Computer Communications and Networks*, July 2013, pp. 1–5.

[11] A. Strunk, "QoS-aware service composition: A survey," in *Proc. of the 8th IEEE European Conference on Web Services*, Dec. 2010, pp. 67–74.

[12] A. Klein, F. Ishikawa, and S. Honiden, "Toward network-aware service composition in the Cloud," in *Proc. of the 2012 International World Wide Web Conference*, Apr. 2012, pp. 959–968.

[13] X. Huang, S. Shanbhag, and T. Wolf, "Automated service composition and routing in networks with data-path services," in *Proc. of the 19th IEEE International Conference on Computer Communication Network*, Aug. 2010, pp. 1–8.

[14] F. Ergun, R. K. Sinha, and L. Zhang, "An improved FPTAS for Restricted Shortest Path," *Information Processing Letters*, vol. 83, no. 5, pp. 287–291, 2002.

[15] G. Xue, A. Sen, W. Zhang, J. Tang, and K. Thulasiraman, "Finding a path subject to many additive QoS constraints," *IEEE/ACM Transactions on Networking*, vol. 15, no. 1, pp. 201–211, 2007.

[16] G. Xue, W. Zhang, J. Tang, and K. Thulasiraman, "Polynomial time approximation algorithms for multi-constrained QoS routing," *IEEE/ACM Transactions on Networking*, vol. 16, no. 3, pp. 656–669, 2008.

[17] J. Huang, X. Huang, and Y. Ma, "An effective approximation scheme for multiconstrained quality-of-service routing," in *Proc. IEEE Global Communication Conference 2010*, Dec. 2010, pp. 1–6.

[18] S. G. R. G. Garroppo and L. Tavanti, "A survey on multi-constrained optimal path computation: Exact and approximate algorithms," *Compt. Netw.*, vol. 54, no. 17, pp. 3081–3107, 2010.

[19] J. Huang, Y. Liu, and Q. Duan, "Service provisioning in virtualization-based Cloud computing: Modeling and optimization," in *Proc. of IEEE Global Communication Conference 2012*, Dec. 2012, pp. 1728–1733.

[20] J. Huang and Y. Tanaka, "QoS routing algorithms using fully polynomial time approximation scheme," in *Proc. ACM/IEEE Internation Workshop on Quality of Service 2011*, Jun. 2011, pp. 1–3.

[21] J. Cardoso, J. Miller, L. Poernomo, and J. Arnold, "Quality of service for workflows and web service processes," *Web Semant.*, vol. 1, pp. 281–308, 2004.

[22] J. Huang. (2013) Service selection simulator. [Online]. Available: https://code.google.com/p/simulator2013/downloads/list