Information Sciences 316 (2015) 370–396

Contents lists available at ScienceDirect

Information Sciences

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

Toward SLA-constrained service composition: An approach based on a fuzzy linguistic preference model and an evolutionary algorithm ☆

Xin Zhao [c,a,b], Liwei Shen [a,b,*], Xin Peng [a,b], Wenyun Zhao [a,b]

[a] School of Computer Science, Fudan University, Shanghai 201203, China
[b] Shanghai Key Laboratory of Data Science, Fudan University, Shanghai 201203, China
[c] School of Management, Beijing Normal University, Zhuhai, Zhuhai 519087, China

ARTICLE INFO

Article history:
Received 16 September 2013
Received in revised form 30 October 2014
Accepted 5 November 2014
Available online 18 November 2014

Keywords:
SLA
Service composition
Multi-objective optimization
Linguistic preference
Evolutionary algorithm
Weighted Tchebycheff distance

ABSTRACT

In a market-oriented service computing environment, both back-end SLA (service level agreement) offers and front-end SLA requirements should be considered when performing service composition. In this paper, we address the optimization problem of SLA-constrained service composition and focus on the following issues: the difficulties related to preference definition and to weight assignment, the limitation of linear utility functions in identifying preferred skyline solutions, and the efficiency and scalability requirements of the optimization algorithm. We present a systematic approach based on a fuzzy preference model and on evolutionary algorithms. Specifically, we first model this multi-objective optimization problem using the weighted Tchebycheff distance rather than a linear utility function. We then present a fuzzy preference model for preference representation and weight assignment. In the model, a set of fuzzy linguistic preference terms and their properties are introduced for establishing consistent preference order of multiple QoS dimensions, and a weighting procedure is proposed to transform the preference into numeric weights. Finally, we present two evolutionary algorithms, i.e., single_EA and hybrid_EA, that implement different optimization objectives and that can be used in different SLA management scenarios for service composition. We conduct a set of experimental studies to evaluate the effectiveness of the proposed algorithms in determining the optimal solutions, and to evaluate their efficiency and scalability for different problem scales.

1. Introduction

The service-oriented architecture provides a loosely coupled computing paradigm in which the business functions and data-accessing APIs can be described, located and accessed as web services using standardized technologies such as WSDL, UDDI and SOAP. With the advent of cloud computing and of software as a service (SaaS), it is expected that there will be an increasingly greater number of different types of services offered on the Internet [11]. In contrast, web services often need to be composed as work flows that use business process description languages, such as BPEL [25] and ORC [34], to perform

☆ This paper is a substantial extension of our earlier work, which is presented in [54].

* Corresponding author at: School of Computer Science, Fudan University, Shanghai 201203, China. Tel.: +86 21 51355343.
E-mail addresses: x_zhao@fudan.edu.cn (X. Zhao), shenliwei@fudan.edu.cn (L. Shen), pengxin@fudan.edu.cn (X. Peng), wyzhao@fudan.edu.cn (W. Zhao).

http://dx.doi.org/10.1016/j.ins.2014.11.016
0020-0255/

more complex tasks or to combine data from different data resources (i.e., data services [7]). The interoperations between service providers and service consumers enable the wide-ranging integration of applications crossing domain boundaries globally, hence encouraging the formation of a market-oriented computing environment [10].

Considering the dynamic and loosely coupled characteristics of service computing, the service level agreement (SLA) and SLA-oriented QoS management mechanisms are important for the establishment and maintenance of the business relationship between service providers and service consumers. As a declarative contract, the SLA clarifies not only the QoS level for which the service provider should be responsible but also the penalty for when violations occur [15,22]. Meanwhile, the aim of SLA management is to guarantee the satisfaction of QoS requirements for service consumers as well as to minimize the possibility of SLA violations for service providers.

In typical SLA management scenarios, service providers are used to predefine a set of QoS levels and prices, allocate computing resources to support these QoS levels, and publish them as SLA offers on some type of broker system or marketplace system [19]. The SLA offers can then be discovered, selected or negotiated to achieve the final SLA contract. In addition, the service provider may also facilitate a self-adaptive resource allocation mechanism to avoid SLA violations at runtime [26,29].

For the SLA management of service composition, the situation is more complex. A service composer can simultaneously play two roles (i.e., provider and consumer) when the service composition is also provided as a service (e.g., a BPEL process). On the one hand, the composer is the service provider from the perspective of its end users; therefore, the QoS should be guaranteed by the composer for the establishment of the front-end SLAs between the two counterparts. However, the composer is also a consumer of the services used in the composition. Meanwhile, there may be different providers providing functionally equivalent services, and different QoS levels can be offered by the same provider. Because the overall quality of the service composition is determined by the quality of each involved service, the composer is required to select an optimal solution. The decision can then be made regarding which providers and which of their QoS levels are desired for the service bindings and for the establishment of each back-end SLA.

Regarding the front-end SLA establishment of the service composition, the front-end QoS requirement is also changeable because the number of end users varies in an open environment. There are different SLA management strategies that are used to determine the front-end QoS requirement and to plan the composition instances for supporting front-end SLA contracting. Regardless of the SLA management strategy used, when the front-end QoS requirement is determined, the optimal solution should be determined under the condition that the front-end QoS requirement is satisfied.

In this paper, we call this optimization problem *SLA-constrained service composition*. Two main challenges make the SLA-constrained service composition a hard problem.

First, because multiple QoS dimensions are involved in SLA contracting and because conflicts exist among dimensions when performing optimization (e.g., lower response times can result in higher prices), it is a typical *Multi-objective optimization problem*. Without considering the relative importance (i.e., preference) between different dimensions, there is no single absolute optimal solution; there is a set of *Pareto optimal* solutions, which can also be called *skyline solutions* [35]. For skyline solutions, we cannot improve the QoS value in one dimension without deteriorating at least one of the values in other dimensions. Thus, the service composer needs to present their preference to identify the *optimal* solution from a set of skyline solutions. Numerous studies have leveraged the linear utility function (i.e., weighted sum) as a *scalarization function* to define the optimal solution [2–4,12,24,30,33,39,43,50,51] where the numeric weights are required and are used as quantified representations of preferences. These studies present algorithms to determine a single optimal solution in terms of a utility function. However, few mention how to elicit and represent preferences or how to assign numeric weights based on preference information. In contrast, in multi-objective optimization theory, it is well known that using weights as quantified measurements of preference in a linear utility function is only effective for convex problem [5,41]. For non-convex problems, there are skyline solutions that can never be obtained by tuning weights or by optimizing the utility function. Unfortunately, we cannot simply assume that the convexity exists for the *discrete* SLA-constrained service composition problem.

To address the above limitations, we redefine the problem and propose the use of the weighted Tchebycheff distance [9,18] rather than the linear utility function as a scalarization function, in which the weights are still used as quantified measurements of preference, but all the skyline solutions can be reached by tuning the weights. Furthermore, considering the difficulties of preference representation and weight assignment, a fuzzy linguistic preference model is introduced to help service composers elicit and specify preference information upon different QoS dimensions. In the preference model, an intuitive method based on fuzzy linguistic terms [37], such as “important than” and “more important than”, are used to define preference relations between different QoS dimensions. A consistent preference order can then be built upon the QoS dimensions based on the preference relations and on a set of properties. Finally, we present a novel weighting procedure to transform the preference order into a set of numeric weights.

Second, because the search space for the optimization process consists of all the possible candidate solutions and because each candidate solution is constructed by combining different back-end SLA offers for each service used in the service composition, it is a *combinational optimization problem*. As is well known, this is an NP-hard problem. The number of possible candidate solutions will increase exponentially with the increasing number of functionally equivalent service providers as well as the increasing number of available SLA offers attached to each provider. Assuming that there are m services required for constructing the service composition, the number of candidate solutions will be $\mathcal{N}$, and $\mathcal{N} = \prod_{i=1}^{m}(n * k)_i$, where n is the number of functionally equivalent providers for the i_{th} service and k is the number of available SLA offers provided by a certain provider. Meanwhile, many studies have noted a fast growth in the number of published web services in the period from 2006 to 2010 [1,53,55]. Cloud computing technologies enable web services to be easily deployed and equipped with different