ISSN(Online): 2320 - 9801
ISSN (Print) : 2320 - 9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 6, June 2016

A Dynamic Service Selection Policy for Improve QOS & Composite Service

K. V. Aditya, A. Vijaya Lakshmi

Assistant Professor, Dept. of MCA, Narayana Engineering College, Nellore, AP, India

Student, Dept. of MCA, Narayana Engineering College, Nellore, AP, India

ABSTRACT: As web service has become a popular way for engineering software on the Internet, quality of service (QoS) which describes non-functional characteristics of web services is often employed in service composition. Since QoS is an aggregated concept consisting of several attributes, service composition on enormous candidate sets is a challenging multi-objective optimization problem. In this paper, we study the problem from a general Pareto optimal angle, seeking to reduce search space in service composition. Pareto set model for QoS-aware service composition is introduced, and its relationship with the widely used utility function model is theoretically studied, which proves the applicability of our model. QoS attributes are systematically studied according to their different types of aggregation patterns in service composition, and QoS-based dominance relationships between candidates and between work flows are defined. Taking advantage of pruning candidates by dominance relationships and constraint validations at candidate level, a service composition algorithm using partial selection techniques is proposed. Furthermore, a parallel approach is designed, which is able to significantly reduce search space and achieve great performance gains. A careful analysis of the optimality of our approach is provided, and its efficacy is further validated by both simulation experiments and real-world data based evaluations.

KEYWORDS: Qos (Quality of Service) Attributes, Web Services Description Language (WSDL) , Simple Object Access Protocol (SOAP), partial selection, service-oriented architecture (SOA).

I. INTRODUCTION

A web service [1] is a programmable module with standard interface descriptions and can be accessed through a standard protocol, which provides a new way of engineering software to quickly develop and deploy web applications. Services computing which enables the techniques of web services centered on service-oriented architecture (SOA) in a well defined framework have become a cross-discipline subject covering various aspects of business and IT services [2]. With the emergence of web services and SOA [3], many service providers publish their services on a public services registry, and a number of industry standards and specifications for web services have emerged, such as Web Services Description Language (WSDL) and Simple Object Access Protocol (SOAP), etc.

As multiple components and parties can be involved in a business process, service composition which combines multiple web services to collaborate with each other and form a composite web service is necessary and important [2]. With the increasing number and scale of services, many providers have started to offer candidate services that are functionally equivalent but have different quality of service (QoS) levels [4]. In order to meet user demands and serve business purposes, besides functionalities, non-functional requirements which are usually described by QoS is often employed in web services [4], [5], [6], and service level agreement (SLA) is becoming a conventional concern in web service composition. Web service composition has become QoS-desired functionality as well as optimize QoS objectives while satisfying user's end-to-end constraints .The aware, which is to select candidates from each service set and combine such candidates to achieve the current web service QoS standards are focused on several attributes, such as response time, throughput, reliability, security, etc. [5], [6]. Therefore, QoS-aware web service composition has to be formulated as a multi-objective optimization problem. An existing idea for solving such a problem is to transform the problem into single objective optimization by using a utility function for comprehensively evaluating the multiple attributes of composite services, and thus it can be formulated into a well known(traditional)optimization problem which can be solved by techniques that have been studied for years [4], [7], [8]. On the other hand, however, another approach has been developed in recent years, which is more general than the former ones and able to tackle the tradeoff

ISSN(Online): 2320 - 9801
ISSN (Print) : 2320 - 9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 6, June 2016

between various objectives [9]. The basic idea is to find the Pareto set of the solutions instead of a single one, guaranteeing that at least one objective is optimal if all the other objectives remain constant.

In this paper, we focus on the problem of reducing search space in multi-objective optimization, and propose partial selection techniques for efficient web service composition based on the Pareto set model. More general than existing approaches, we study the QoS attributes in web services comprehensively, and classify them into six categories according to their properties in aggregation patterns. Besides additive and multiplicative functions, we further study Boolean operations, such as union and intersection, which are important in web services but usually ignored by existing works. We discuss the dominance relationship between workflows, based on which we define the Pareto optimal compositions. The connection of Pareto optimal compositions with utility function solutions is discussed, which elucidates that the Pareto set model is more general and the utility function method can be used based on it. Taking advantage of partial selection, we theoretically prove that pruning candidates outside the local Pareto sets can remarkably reduce the search space while guaranteeing the optimality. Moreover, we define the bad cumulative effect of QoS attributes, and state how to move the constraints validation to the candidates level, which can further reduce the number of promising candidates. A distributed service composition algorithm using partial selection techniques is designed, which can be implemented in a parallel way and improves efficiency greatly. Theoretical analysis of the performance of our algorithm is presented. Both simulation experiments and real-world data based evaluation are conducted, which verify the optimality and effectiveness of our approach.

Y. Chen, C. Lin, and J. Hu are with the Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China. E-mail: chenying12@mails.tsinghua.edu.cn , chlin@tsinghua.edu.cn , jiehu1990@gmail.com .
J. Huang is with the State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China. E-mail: huangjw05@gmail.com
Manuscript received 31 Aug. 2014; accepted 2 Dec. 2014. Date of publication 17 Dec. 2014; date of current version 12 June 2015. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below. Digital Object Identifier no. 10.1109/TSC.2014.2381493

II. BACKGROUND OR RELATED WORK

2.1 Models

Generally, QoS-aware web service composition is often formulated as an optimization problem, seeking to maximize or minimize the QoS attributes, such as minimizing response time, maximizing throughput, etc. Some works studied the problem considering only one objective, while other researches introduced more than one QoS criteria and studied them together, making QoS-aware web service composition become multi-objective optimization problem [4], [5], [6], [7], [8], [9]. With more and more QoS attributes being considered, how to tackle the tradeoff between different objectives becomes an important challenge. For example, a user may want to have minimum response time while paying least money, which is hard to achieve since services with less latency usually cost higher.

2.2 Solutions

Several approaches have been proposed for QoS-aware web service composition, which has been proven NP-hard in the strong sense [7]. One approach is to perform an exhaustive search and emulate all the possible compositions. This can guarantee to find the optimal composition, but it suffers from combinatorial explosion and inefficiency. With the increasing number of services and candidates, the composition space size will grow rapidly, making the searching process time consuming and resource-consuming. It has been widely recognized that exhaustive search is not practical for real-time execution in most of service composition scenarios [7].

III. MODEL OF SERVICE COMPOSITION

3.1 QoS Attributes

Let A ¼ð a1;a2;...;aLÞdenote the vector of L QoS attributes of the system. The performance of a workflow SP is the cumulative effect of the performance of m candidates that compose it. We define the aggregate function of the candidates' QoS values to the workflow's QoS values as (1)

f :c1 j1;c2 j2;...;cm jm————————————————! SP; (1)

which describes how to obtain the workflow's QoS values according to candidates' QoS values.

3.2 Service Composition.

ISSN(Online): 2320 - 9801
ISSN (Print) : 2320 - 9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 6, June 2016

In this paper, we consider not only regular operations but also Boolean operations of QoS attributes. The QoS attributes in web service system are classified into six categories based on their properties of aggregate functions, which are additive, multiplicative, minimum, maximum, union and intersection. We study the QoS attributes according to their categories of aggregate functions instead of paying attention to the concrete individual QoS attributes, which makes our model more general and scalable. For example, the attribute confidentiality can be represented by a 0-1 variable denoting whether the service is encrypted or not, thus its aggregate function is Boolean operation. Note that the classifications of QoS attributes are not limited to sequential workflows. For instance, the execution time of a parallel workflow with synchronized merge can be classified into maximum aggregate function, the confidentiality in a serial workflow is classified into intersection category

Definition 1 (Feasible Composition Solution): Given any task composed of services I ¼f 1;2...mg and the QoS constraints, a feasible solution is any composition solution SP that selects a candidate service ci ji from each service set Ci and satisfies all the QoS constraints, i.e., oprðRr;boundrÞ is true for each 1 r M.

3.3 Optimal Solutions

To tackle the tradeoff between different objectives, we combine the whole attributes together and use original QoS values to find the optimal compositions, which is convenient and will not lose information of QoS attributes.

IV. APPROACH FOR FINDING OPTIMAL COMPOSITIONS

Composition Algorithm

In this subsection, we discuss how to obtain the optimal compositions based on our partial selection approach. Let S0 represent the set of composition solutions based on e Ci, which is the Cartesian product of e Ci expressed by (31), S0 ¼ e C1

C2 Cm: (31) Note that the compositions in S0 violating QoS constraints are not feasible solutions, and they are also not promising optimal solutions. So we can validate each solution in S0 and remove those compositions not satisfying constraints before performing pair wise comparison on the solutions, which can reduce space and save time. Let SP 2 S0 be a composition solution, and the vector of its constraints values is VR ¼ FRðSPÞ¼ðR1;R2;...;R r;...;RMÞ: (32) Recall that Rr is the rth QoS constraint value of SP, and M is the number of QoS constraints

Parallel Approach

Based on Ci;iþ1, we prune those unpromising combinations and only keep the potential ones. To be more specific, we make use of the attributes that have bad cumulative effect and prune those combinations that violate QoS constraints, since they are not promising components for optimal compositions. Then we obtain the set after constraints validation as C0i;iþ1. Furthermore, we can find the corresponding Pareto set of C0i; iþ1 expressed as e Ci;iþ1 and prune those unpromising combinations, which can further reduce space. The merge process on different pairs fe Ci;e Ciþ1g can be done concurrently on multiple servers/processors. After all pairs have finished the pruning process and obtained their corresponding sets e Ci; iþ1, we say one round ends. Then the next round begins and the processes continue and iterate until we obtain the big set which includes all the m categories of services. An intuitive process of the parallel approach is shown in Fig. 1. To analyze the efficiency of our algorithm using parallel approach, we mainly focus on the composition space size and operation time. Consider a task with m services, each service has n candidates. Assume in the first round, the number of potential candidates for each set Ci is reduced to pin after constraints validation and finding the corresponding Pareto sets. The pruning process for each service set can be done concurrently and in the worst case, this would take n2 operations. Then in the second round, the m services sets are divided into dm 2e pairs. We obtain Ci; iþ1 as new sets and the size of Ci; iþ1 is pipiþ1n2. The pruning process on Ci; iþ1 can take at mostðpipiþ1n2Þ2 operations and the sets obtained are e Ci; iþ1. Assume the size of e Ci; iþ1 can be reduced to pipiþ1pi;iþ1n2. Then the third round begins and we obtain Ci; iþ1; iþ2; iþ3 as new set, which is the Cartesian product of Ci;iþ1 and Ciþ2;iþ3. The size of Ci;iþ1; iþ2;iþ3 is pipiþ1piþ2 piþ3pi;iþ1piþ2;iþ3n4.

ISSN(Online): 2320 - 9801
ISSN (Print) : 2320 - 9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 6, June 2016

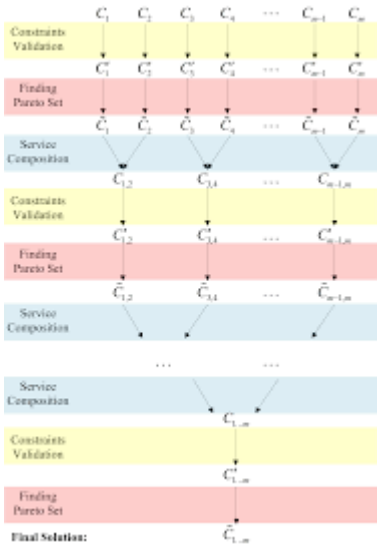


Fig. 1. A parallel approach for DPSA algorithm.

*Algorithm*1: NLA $(P_{i-1}, P_i, s_{ij}, s_{(i+1)k})$

Input: User's start location P_{i-1} and end location P_i,
concrete services' location s_{ij} and $s_{(i+1)k}$

Output: $< t_{ij}, s_{ij} >$

1 $D(s_{ij}, s_{(i+1)k})$ = get distance between s_{ij} and $s_{(i+1)k}$ applying(3)
2 d_{ij} = getDelayTime($D(s_{ij}, s_{(i+1)k})$)
3 $D(s_{ij}, P_{i-1}, P_i)$ = get distance between the user and s_{ij} applying(6)(7)(9)
4 u_{ij} = getDelayTime($D(s_{ij}, P_{i-1}, P_i)$)
5 $t_{ij} = d_{ij} + u_{ij}$
6 **return** $< t_{ij}, s_{ij} >$

ISSN(Online): 2320 - 9801
ISSN (Print) : 2320 - 9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 6, June 2016

Algorithm2: SCMI (S, C, P, n, m)

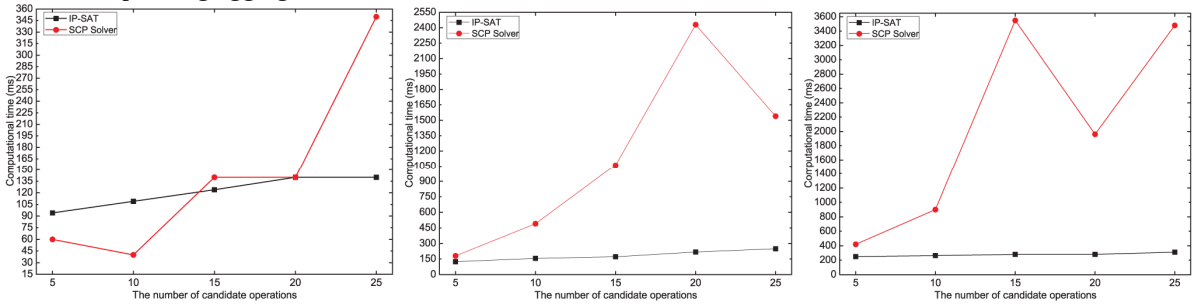
Input: the set of all the abstract tasks is S,
the set of all the concrete services is C,
the set of users locations is P,
the number of tasks is n
and the number of concrete services for each task is m
Output: the optimal service scheme $\overline{S}$

$\overline{S}$ = empty ArrayList
for $i = 1$ to n
if($i! = n - 1$)
C_i = all the concrete services corresponding to S_i
C_{i+1} = all the concrete services corresponding to S_{i+1}
\\getLocation(i) is used to get user's locations
$P_i = getLocation(i)$
$P_{i-1} = getLocation(i-1)$
for $k = 1$ to m
for $j = 1$ to m
$< t_{ij}, s_{ij} >= NLA(P_{i-1}, P_i, s_{ij}, s_{(i+1)k})$
if $(i = 1)$
$\tau_{ij} = min\{t_{ij}\}$
else
$\tau_{ij} = min\{t_{ij} + \overline{S_{ij}}\tau_{(i-1)l}\}$
$\overline{S_{(i+1)k}}\tau_{(i-1)l} =< S_{ij}, \tau_{ij} >$
else
C_i = all the concrete services corresponding to S_i
P_i = getLocation(i)
P_{i-1}= getLocation($i-1$)
for $j = 1$ to m
$D(s_{ij}, P_{i-1}, P_i)$=get distance applying (6)(7)(9)
$u_{ij} = getDelayTime(D(s_{ij}, P_{i-1}, P_i))$
\\assume s_{ij} already selects the concrete service $s_{(i-1)l}$
$\tau_{ij} = min\{u_{ij} + \overline{S_{ij}}\tau_{(i-1)l}\}$
\\This is the last abstract task and τ_{ij} is the shortest network latency,\\so s_{ij} is one part of the best service scheme
$\overline{S}.add(s_{ij})$
for $i = 1$ to n
C_i = all the concrete services corresponding to S_i
for $j = 1$ to m
if $(s_{ij} = \overline{S}.get(n-i))$
$\overline{S}.add()$**break;**
return

V. EVALUATION

5.1 Simulation Experiments

We focus on a task in the form of a sequential workflow (other structures can be transformed to sequential workflow by existing techniques [34]). There are five services in the task, and each service has 20 candidates which can offer the same functionality but have different QoS values. For QoS objectives and constraints, response time and throughput are considered, which commonly act as two important attributes in several QoS-aware service composition problems. The corresponding aggregate



(a) The number of workflow tasks = 7 (b) The number of workflow tasks = 14 (c) The number of workflow tasks = 20

ISSN(Online): 2320 - 9801
ISSN (Print) : 2320 - 9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 6, June 2016

VI. CONCLUSION

In this paper, we study QoS-aware web service composition, focusing on efficiently finding its Pareto optimal solutions. Pareto set model for service composition is proposed, whose general applicability is demonstrated by our proof of the connection between the solutions of Pareto optimal approach and utility function method. Six categories of QoS attributes are systematically studied, and their aggregate functions are discussed. Partial selection techniques are proposed to reduce the search space, and a distributed algorithm called DPSA is designed. Taking advantage of multi-level component pruning and parallelization, our scheme is able to significantly improve the efficiency while guaranteeing the optimality of service composition. Theoretical analysis and experimental results are given to validate the efficacy of our approach. This work is expected to provide both a theoretical framework and practical solutions for service composition in large-scale web service systems.

VII. ACKNOWLEDGMENTS

This work is supported by the National Natural Science Foundation of China (No. 61020106002) and Tsinghua University Initiative Scientific Research Program (No. 20121087999). Ying Chen is the corresponding author.

REFERENCES

[1] H. Haas and A. Brown, "Web services glossary," W3C Working Group Note (11 February 2004), 2004.
[2] L.-J. Zhang, J. Zhang, and H. Cai, Services Computing. New York, NY, USA: Springer and Tsinghua Univ. Press, 2007.
[3] E. Newcomer and G. Lomow, Understanding SOA with Web Services (Independent Technology Guides). Reading, MA, USA: AddisonWesley, 2004.
[4] L. Qi, Y. Tang, W. Dou, and J. Chen, "Combining local optimization and enumeration for QoS-aware web service composition," in Proc. 8th IEEE Int. Conf. Web Serv., 2010, pp. 34–41.

BIOGRAPHY

.

ADITHYA K.V. is a Assistant Professor in the Department of Master of Computer Applications, Narayana Engineering College, Nellore. His research interest in A Dynamic Service Selection Policy for Improve QoS & Composite Service

VIJAYA LAKSHMI.A completed master of computer applications in Narayana Engineering College, Nellore. Her research interest in A Dynamic Service Selection Policy for Improve QoS & Composite
Service