计算机研究与发展
Journal of Computer Research and Development
DOI:10.7544/issn1000-1239.2016.20150078
53(12):2867-2881,2016

不确定感知的自适应云计算服务组合

任丽芳[1,2] 王文剑[1] 许 行[1]
[1](山西大学计算机与信息技术学院 太原 030006)
[2](山西财经大学应用数学学院 太原 030006)
(renlf@sxufe.edu.cn)

Uncertainty-Aware Adaptive Service Composition in Cloud Computing

Ren Lifang[1,2], Wang Wenjian[1], and Xu Hang[1]
[1](*School of Computer and Information Technology, Shanxi University, Taiyuan* 030006)
[2](*School of Applied Mathematics, Shanxi University of Finance & Economics, Taiyuan* 030006)

Abstract Cloud computing service composition is to select appropriate component services from numerous of services distributed in different clouds to build scalable loose coupling value-added applications. Traditional service composition methods are usually divided into selection stage and composition stage. Hardly guaranteeing the services with the best performance in the selection stage are still optimal in the execution stage because of the dynamic nature of the cloud computing environment and the stochastic nature of services evolution. Focusing on these two natures of service composition in cloud computing environment, a service composition model is built based on POMDP (partially observable Markov decision process) named as SC_POMDP (service composition based on POMDP), and a Q-learning algorithm is designed to solve the model. SC_POMDP can dynamically select the component services with outstanding QoS (quality of service) during the execution of service composition, which aims to ensure the adaptability of the service composition. Different from most existing methods, the proposed SC_POMDP regards the environment of service composition as being uncertain, and the compatibility between component services is considered, hence SC_POMDP is more in line with the real situation. Simulation experiments demonstrate that the proposed method can successfully solve the problems of service composition in different sizes. Specially, when service failure occurs, SC_POMDP can still select the optimal alternative component services to ensure the successful execution of the composite service. Compared with two existing methods, the selected composite service by SC_POMDP is best in response time and throughput, which reflects the superior adaptation of SC_POMDP.

Key words adaptive service composition; cloud computing environment; uncertainty-aware; partially observable Markov decision process (POMDP); Q-learning algorithm; quality of service (QoS)

摘 要 云计算服务组合是从众多分布在不同云计算平台上的远程服务中选择合适的组件服务来构建可伸缩的松耦合的增值应用.传统的服务组合方法通常将服务选择与服务组合分阶段进行,由于云计算环境的动态性和服务自身演化的随机性,不能保证选择阶段性能最优的服务在组合服务执行阶段依然

收稿日期:2015-01-26;**修回日期**:2015-08-11
基金项目:国家自然科学基金项目(61273291,61673249);山西省回国留学人员科研资助项目(2016-004)
This work was supported by the National Natural Science Foundation of China (61273291, 61673249) and the Research Project of Shanxi Scholarship Council of China (2016-004)
通信作者:王文剑(wjwang@sxu.edu.cn)

是最优的.考虑到云计算环境服务组合的动态性和随机性,建立基于部分可观测 Markov 决策过程(partially observable Markov decision process, POMDP)的服务组合模型 SC_POMDP (service composition based on POMDP),并设计用于模型求解的 Q 学习算法.SC_POMDP 模型在组合服务运行中动态地进行服务质量(quality of service, QoS)最优的组件服务选择,且认为组合服务运行的环境状态是不确定的,同时 SC_POMDP 考虑了组件服务间的兼容性,可保证服务组合对实际情境的适应性.仿真实验表明,所提出的方法能成功地解决不同规模的服务组合问题,在出现不同比率的服务失效时,SC_POMDP 仍然能动态地选择可用的最优组件服务,保证服务组合能成功地执行.与已有方法相比,SC_POMDP 方法所选的服务有更优的响应时间和吞吐量,表明 SC_POMDP 可有效地提高服务组合的自适应性.

关键词 自适应服务组合;云计算环境;不确定感知;部分可观测 Markov 决策过程;Q 学习算法;服务质量

中图法分类号 TP311

随着云计算技术的兴起,软件的开发、交付和维护模式正发生着巨大的变革.面向服务的计算(service oriented computing, SOC)作为一种新兴的计算模式,以服务作为构建软件应用的基本元素,引起学术界和工业界的普遍关注.Web 服务是具有标准接口描述的可编程模块,是一种灵活的标准的服务交付接口技术,常用于云计算平台服务实施中提供各种已经实现的功能或资源.然而单一的服务所能提供的计算资源和能力是有限的,为满足用户日益复杂的需求,需要从众多的分布在不同的云平台上的服务中,选择合适的组件服务,并按照一定的业务规则进行组合,构建可伸缩的松耦合的组合服务[1-2].

服务组合是 1 个复杂而具有挑战性的工作,在这方面已经进行了许多的研究.按照自动化程度,服务组合的方式可以分为手动、半自动和自动服务组合 3 大类.随着可用服务不断涌现,手动方式对现有的服务进行分析和组合已经变得不可行[3].

通常自动组合方式会将服务组合任务转化为 1 个经典的规划问题求解[4-7].首先用规划语言,如 PDDL[4], TRIPS[5]或 GOLOG[6]等,对服务进行形式化描述;然后用人工智能规划算法,如分层任务网络(hierarchical task network, HTN)规划[7]、Estimated-regression 规划[4]等来求解该规划问题;最后,将规划问题的解转变为自动的服务调用.自动服务组合通常只是寻求满足用户功能需求的组合服务,而不考虑组合服务的服务质量(quality of service, QoS),因此组合服务的性能难以保证.更重要的是经典的规划算法一般不考虑系统的动态特性,因此自动服务组合方式一般假定服务是确定不变的,然而在实际的服务组合环境中,服务的性能会受 Internet 的动态性和云服务提供组织升级演化的影响,甚至变得不可用,所以自动服务组合是易于失败的.

在半自动服务组合中,首先构建 1 个过程模型,然后以全局或局部最优为目标自动地为每个抽象任务选择服务实例[3].现有的多数服务选择方式都是 QoS 感知的[8-10],这种服务选择方式比较容易满足用户对组合服务性能的要求.还有一些半自动服务组合是基于信任或信誉管理[11],或者网页排名[12]等.在大多数这些方法[8-13]中组件服务的选择和组合服务的执行是 2 个独立的阶段,很难保证选择阶段具有最优性能的服务在执行阶段其性能依然是最优的,甚至可能出现组件服务在执行阶段失效,导致重新启动服务组合的选择和执行过程,然而重新规划也不能保证新的组合服务一定有效.

事实上,由于作为云计算支撑环境的 Internet 是动态的,导致服务的 QoS 不断地变化,如网络访问量的增加可能导致服务的响应时间延长,网络连接失败直接导致服务变得不可用等.此外,云计算提供组织也在不断地演化.如会出现一些新的服务、会有某些服务消失、也会有某些服务发生升级等.因此这种云计算环境中具有预定义过程模型的 QoS 最优的服务组合问题,可以看作随机环境中多阶段决策过程优化问题.Markov 决策过程(Markov decision process, MDP)是动态规划与 Markov 过程相结合的产物,在对决策问题建模时考虑其随机性和有序性,适合于对随机环境中多阶段决策过程进行优化控制.然而 MDP 模型假定对系统的运行状态是完全了解的,在实际的服务组合问题中这个假定很难成立.部分可观测 Markov 决策过程(partially observable Markov decision process, POMDP)模型是将 MDP

模型应用于更一般环境中的扩展,适用于建模不确定环境中的决策过程,模拟决策者和系统在信息不完全确定的情况下交互,并进行决策的过程. POMDP 在机器人导航、移动目标俘获等领域已有广泛的应用[14]. 云计算服务组合研究在云计算的动态环境中,服务组合的构建者对服务组合所处的云计算环境以及服务组合系统的运行状态不完全了解的情况下,对最优服务组合进行决策. 这种动态的不确定环境中的决策问题适合用 POMDP 进行建模,因此本文基于 POMDP 建立服务组合模型 SC_POMDP(service composition based on POMDP),该模型不仅能适应云计算环境的动态性和服务演化的随机性,而且能建模服务组合运行中系统状态的不确定性,所以 SC_POMDP 模型是自适应的. 然而 POMDP 模型的复杂性使其最优策略的求解非常困难,现有的求解算法只能对较小规模的问题进行精确求解[15]. Q 学习算法是一种增强学习技术,可以通过试错方法学习最优交互策略,使系统从环境获得最高回报[16],因此本文采用 Q 学习算法对 SC_POMDP 模型进行求解.

1 相关工作

云计算环境的动态性加之服务的自治性,使得组合服务的性能难以保证,因此构建适应这种不确定环境的服务组合成为了 1 个挑战,引起国内外学者对自适应服务组合方法进行了广泛的研究[17-21].

范小芹等人[17]考虑到组件服务 QoS 的不确定性对组合服务成功率的影响,用随机变量的均值和方差来度量 Web 服务的 QoS 指标,然后基于 MDP 建立服务组合模型,提高了服务组合的成功率. 然而,虽然均值和方差能反映 QoS 指标的平均水平及其波动程度,但是实际中并不能保证组合服务运行中组件服务的 QoS 指标真实值和 QoS 指标理论值保持一致. 因此文献[17]以组件服务 QoS 指标的理论分布作为服务选择的标准,在一定程度上提高了服务组合的成功率,但是该方法对服务组合运行环境的不确定性适应明显不足.

Wang 等人[18]考虑了 Web 服务演化的不确定性,设计了基于 MDP 的服务组合模型,在组合服务运行时进行具体服务的选择,有效地减少了服务选择阶段与组合服务运行阶段之间服务演化所带来的影响. 然而该方法在进行服务选择时没有考虑服务组合系统所处的运行环境的不确定性,事实上不同的运行环境中组件服务的实际性能有很大的差异. 文献[18]的方法提升了服务组合对服务演化的适应性,但是没有考虑服务组合对运行环境的适应性,因此该方法能保证组合服务的成功执行,但是并不能保证组合服务的性能最优.

Yang 等人[19]建立了云间服务组合中相继任务的服务对集合,并定义了服务对效用的评价函数. 该方法基于 MDP 为服务组合建立模型,选择效用函数值最优的服务对形成服务组合. 该模型在服务对的效用值计算中,认为组件服务的 QoS 属性值是确定的,而实际中组件服务的 QoS 属性值波动较大,所以该服务组合方法的适应能力有很大的局限.

Ardagna 等人在文献[20]将服务组合建模为 1 个混合整数规划问题,并以环剥技术加速优化过程,利用协商机制降低服务过程中由于用户约束不满足导致的失败率,体现了该方法对用户约束的自适应性,目标是寻找满足用户约束的可行的组合服务. 然而该方法以 QoS 属性的理论分布值作为实际的 QoS 属性值,没有考虑组合服务对不确定环境的适应性.

文献[12]利用公共注册机构的动态 Web 服务定义语言(Web Service Definition Language, WSDL)信息近似服务的瞬时状态,以瞬间快照上的链接分析代表服务在该时刻受不同用户欢迎的程度,体现该方法对组件服务信誉度的适应性. 该方法把组件服务的选择形式化为高被引服务的选择. 由于服务 QoS 属性间的不一致性,高被引服务的高访问量会引起响应时间等其他 QoS 属性性能的降低,所以该方法的适应性考虑不全面.

Klein 等人在文献[21]将 QoS 属性区分为服务自身决定的属性和与网络相关的属性,提出了网络感知的方法,用 2 种不同的方法计算 2 类不同的 QoS 属性值. 对网络相关的 QoS 属性,根据相继服务所在服务器的网络距离计算 QoS 属性值. 在此基础上,该文设计了网络感知的遗传算法,求解最优服务组合,该方法的适应性体现在根据一定的经验概率选择执行遗传操作,然而该方法中服务选择与组合服务的执行分阶段进行,没有考虑对组件服务演化的适应性,不能保证服务组合的成功执行.

不同于已有的服务组合方法,本文针对云计算环境中,服务性能受 Internet 的动态性以及服务自身演化随机性的影响,且系统对组合服务运行环境状态不完全了解,这种更一般情况的服务组合问题展开研究. 同时本文考虑了组件服务之间的兼容情况,建立了基于 POMDP 的服务组合模型 SC_POMDP,

并设计了模型的求解算法,为解决在不确定的云计算环境中自适应地进行性能最优的服务组合问题提供了一种可行方法.

2 云计算服务组合

本节给出服务组合问题中的一些相关概念,并给出原子服务、组件服务以及组合服务的形式化描述.同时,为了说明云计算环境服务组合过程及其特征,本节设计了1个云计算服务组合的场景.

2.1 相关概念

定义 1. 原子服务(atomic service)是1个独立的、功能完整的可以通过网络进行发布、定位以及访问的最小资源单位.1个原子服务可以形式化为1个三元组(N_{id},C_{fun},$\boldsymbol{M}_{qos}$),其中:

N_{id}是原子服务的标识,通过N_{id}可以唯一地确定1个原子服务;

C_{fun}代表原子服务的功能分类号,功能相同的原子服务有相同的C_{fun}.能完成某个特定任务的服务具有相同的C_{fun},这些服务组成1个集合,称为该任务的候选服务集(set of candidate services);

$\boldsymbol{M}_{qos}$代表原子服务在被调用历史中表现的服务质量,是1个矩阵,矩阵的行数是该服务被调用的次数,列数是所考察的QoS属性(如响应时间、价格、可用性等)的个数.其中行$i(m_{i1},m_{i2},\cdots,m_{in})$代表该服务在第$i$次被调用时各QoS属性的值.

2个或多个原子服务可以组合产生1个新的服务,这个新的服务可以执行更复杂的其中任何1个原子服务无法独立完成的任务.这个包含多个原子服务的新服务是1个组合服务.构造组合服务的过程可以迭代进行,也就是说,组合服务可以被用作组成另1个更复杂组合服务的组件服务.图1是原子服务、组件服务与组合服务关系的1个示例.图1中,大虚线框内的是1个组合服务,小虚线框内的是1个组件服务,每个圆圈代表1个原子服务.

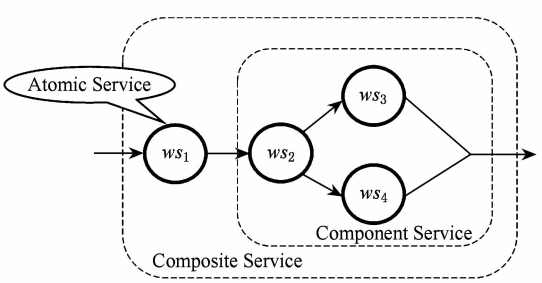

Fig. 1 An example graph of atomic service, component service and composite service.

图1 原子服务、组件服务与组合服务示例图

定义 2. 组件服务(component service)本质上是1个较小规模的组合服务,组件服务能够实现1个小规模的完整业务解决方案,所以经常作为1个整体.组件服务可以形式化地定义为1个四元组(N_{id},C_{fun},$\boldsymbol{M}_{qos}$,$\boldsymbol{V}_{memb}$),其中:

N_{id},C_{fun}和$\boldsymbol{M}_{qos}$的形式及含义与原子服务定义中相同;

$\boldsymbol{V}_{memb}$是依序构成这个组件服务的所有原子服务的N_{id}组成的序列.

服务的强大之处就在于组件服务能动态地进行集成来执行新的更复杂的任务,这个过程就是服务组合(service composition).

定义 3. 组合服务(composite service).为满足用户需求,将已有的具有不同功能的组件服务按照一定的业务逻辑进行集成,形成可伸缩的松耦合的增值应用.其中的组件服务,包括由1个原子服务组成的组件服务,即原子级的组件服务.这样,组合服务可以形式化地表示为1个序列(ws_1,ws_2,…,ws_n),其中$ws_i(i=1,2,\cdots,n)$为依序组成组合服务的所有组件服务的N_{id},n为组合服务中组件服务的个数.

由于具有循环、分支和并行结构的服务组合都能够转化为一种顺序结构[22],所以本文主要解决业务逻辑的过程模型为顺序结构的服务组合问题.

2.2 云计算服务组合的1个场景

为了说明服务组合过程及其在云计算环境的特征,考虑如下的场景:某用户捕获1个物体在运动时的1组多视角照片,希望根据这组二维图像利用用户设计的重建算法重建该物体的三维模型,并将重建的三维模型进行存储.由于拍摄时物体在运动,照片可能比较模糊,所以在三维重建之前需要进行二维图像复原,以提高二维图像质量.由于本地资源有限,用户希望通过云计算环境所提供的服务来完成这个任务.为此可以首先利用云计算环境中的模糊图像复原服务将模糊的二维图像进行复原,然后利用云计算平台提供的强大的计算能力运行用户设计的三维重建算法,得到重建的三维模型,最后将三维模型在云空间存储.图2是对该场景服务组合的描述.

显然为满足用户的需求需要对现有的服务进行组合.用户的需求可以分解为3个任务:图像复原任务t_1、三维重建计算任务t_2、云存储任务t_3,如图2所示.在Internet中,存在许多不同的云计算服务提供商,为简单起见,本文在图2所示的场景中假设所涉及的服务由图中所示的4个云提供,每个云提供

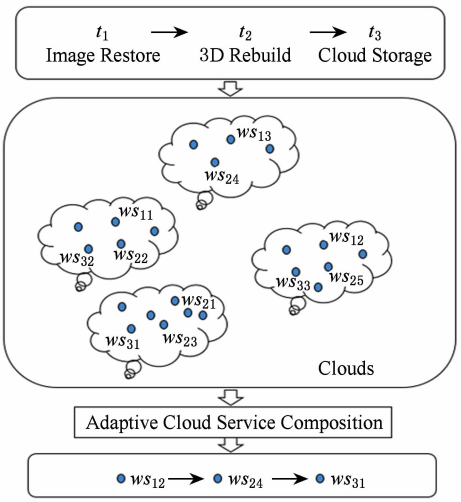

Fig. 2 A scenario of cloud services composition.

图 2 云计算环境服务组合场景

商提供许多不同的原子服务,在图中以小实心圆表示.经过在所有的云上进行服务发现,每个任务都可以由几个不同的服务独立地完成,图 2 中以 ws_{ij} 表示可以完成任务 t_i 的第 j 个候选服务,也是构成目标组合服务的可用组件服务.候选服务的次序只是区分同一候选服务集中不同的服务,这些候选服务可能分布在相同或不同的云上.

完成相同任务的不同服务组成该任务的候选服务集.在图 2 所示的场景中,任务 t_1,t_2,t_3 的候选服务集分别为

$$C_1=\{ws_{11},ws_{12},ws_{13}\},$$

$$C_2=\{ws_{21},ws_{22},ws_{23},ws_{24},ws_{25}\},$$

$$C_3=\{ws_{31},ws_{32},ws_{33}\}.$$

服务之间由于语法、语义或行为等其他原因,兼容情况(服务间的兼容性判定不在本文研究范围,故不做过多的解释)如图 3 所示:

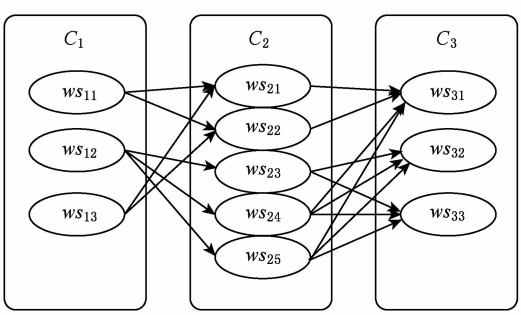

Fig. 3 The compatibility of candidate services.

图 3 候选服务之间的兼容情况

其中 C_1,C_2,C_3 分别为 3 个任务 t_1,t_2,t_3 的候选服务集,每个箭头所指向的是下一个任务的候选服务集中能与该服务兼容的服务.本文用 $R_{\mathrm{trs}}(ws)$ 表示在下一个任务的候选服务集中与服务 ws 兼容的候选服务子集.因此,图 3 中每 1 条由箭头连接的从 C_1 集合出发,经过 C_2 集合,到达 C_3 集合的路径,即为一种可行的组合方案,如 $(ws_{11},ws_{21},ws_{31})$,$(ws_{12},ws_{23},ws_{32})$,$(ws_{13},ws_{22},ws_{31})$等分别为一种可行的组合服务.这样,服务组合问题就是依照一定的策略,依次从每个任务的候选服务集中选择与前一任务所选组件服务兼容的最优服务完成该任务,进而完成复合任务.按照这种组合方式,图 2 所示的场景中选择执行的组合服务为$(ws_{12},ws_{24},ws_{31})$.

考虑到用户的满意度,应该选择使组合服务性能最优的服务.然而由于云计算环境的动态性以及服务演化的随机性,服务的 QoS 在不断变化,有时甚至会出现某个服务失效的情况.如何在这种动态环境中为每个任务选择服务使当前组合服务的性能最优是本文研究的目的.自适应云计算服务组合旨在感知组合服务执行的网络环境,选择适应该环境的组件服务,同时若被选择的组件服务由于演化升级等原因发生失效时,可以从候选服务集中重新选择适应性较高的服务替换失效的组件服务.

3 自适应服务组合模型 SC_POMDP

本节利用第 2 节定义的组件服务与组合服务,建立服务组合模型 SC_POMDP,给出 SC_POMDP 模型中服务的选择机制,并设计该模型求解的 Q 学习算法.

3.1 建立服务组合模型 SC_POMDP

在 POMDP 模型中,决策者周期地或连续地观察随机动态系统,在决策时刻根据观察到的系统所处的状态分布以及所采取的策略从可用的行动集合中选择 1 个动作.POMDP 系统下一步的状态是随机的,并且其状态转移概率具有无后效性,即下一个决策只依赖于系统当前的状态,与历史状态无关.决策者根据新的观察判断系统的状态分布,做出新的决策,如此反复地进行[14].云计算环境的服务组合,依照业务逻辑所定义的过程模型,以组合服务性能最优为目标,根据对系统所处状态分布的判断,从每个任务的候选服务集中选择当前运行状态下性能最优的组件服务,执行该服务引起系统的状态分布发生改变,且下一个服务的选择只与系统的当前状态有关,与历史状态无关.因此本文采用 POMDP 建模云计算环境的服务组合,主要步骤如下:

Step1. 根据用户需求,建立代表服务组合业务逻辑的任务流模板;

Step2. 为模板中每个任务聚集可用候选服务集,并建立服务间的兼容关系;

Step3. 建立服务组合模型 SC_POMDP;

Step4. 利用 Q 学习算法对每个服务的历史运行 QoS 数据进行学习,得出代表每个任务的各候选服务在不同系统状态的 QoS 性能的矩阵 $\boldsymbol{Q}$;

Step5. 根据系统实时运行状态分布以及矩阵 $\boldsymbol{Q}$,并结合服务之间的兼容性,选择在当前状态使组合服务有最大累计回报值的可用组件服务;

Step6. 重复 Step5 直到每个任务都完成.

其中,步骤 Step3 建立的服务组合模型定义如下:

定义 4. 基于 POMDP 的服务组合模型 SC_POMDP,该模型可以形式化为 1 个六元组(S,A,T,Z,O,R),其中:

1) S 为状态集合,$S=\{s\}$,其中任意 1 个状态 $s=(t,q)$是 1 个二元组,t 表示正在执行的任务的序号;q 表示当前组合服务运行所处的 QoS 等级状态,q 的可能取值为集合 $Q_{\text{qos}}=\{\text{A},\text{B},\text{C},\text{D}\}$中的元素.其中 A 表示服务质量为优;B 表示服务质量中等;C 表示服务质量较差;D 表示服务失败,即服务没有正确执行. 如 $s=(3,\text{B})$,表示组合服务正在运行任务 3,且系统所处的 QoS 状态为 B.

2) $A=\bigcup_{i=1}^{n}A(t_i)$是模型中所有候选组件服务组成的集合,其中 $t_1,t_2,\cdots,t_n$ 分别代表服务组合中的任务,$A(t_i)=\{ws_{i1},ws_{i2},\cdots,ws_{ik_i}\}$是任务 t_i 的候选服务集,其中的 $ws_{ij}(j=1,2,\cdots,k_i)$是任务 t_i 的第 j 个候选服务. k_i 是任务 t_i 的候选服务集中服务的个数. A 是所有 $A(t_i)$的合集,即组合服务所有可能用到的组件服务的全体构成的集合.

3) $T(s,ws,s')$是转移概率函数,表示组合服务从状态 s 调用服务 ws,使组合服务的状态转移到 s' 的概率. 设当前状态为 $s=(t_i,q)$, 其中 $i=1,2,\cdots,n$,若选择的组件服务 ws 执行失败,即任务 t_i 没有完成,则 $s'=(t_i,\text{D})$;若组件服务 ws 执行成功,则 $s'=(t_{i+1},q')$,q'为组合服务的实际运行状态,可能为 A,B 或 C.

4) $Z=\{\boldsymbol{z}\mid\boldsymbol{z}=(z_1,z_2,\cdots,z_m)\}$是观测值集合,其中,$\boldsymbol{z}$ 是 1 个向量,代表 1 次观测,每个分量 $z_i(i=1,2,\cdots,m)$代表组件服务 1 次执行中观测到的 1 个 QoS 属性值,可能是服务的价格、响应时间或吞吐量等.

5) $O(ws,s',\boldsymbol{z})$是观测函数,表示调用服务 ws 后状态转移到 s'时观测到 $\boldsymbol{z}$ 的概率. 本文根据服务执行历史数据中 QoS 值和当前服务运行的观测值,利用 Bayes 公式进行计算.

6) $R(s',s,ws)$是回报函数,表示在调用服务 ws 后系统从状态 s 转移到状态 s'所得的回报值. $r=R(s',s,ws)$是 1 个实值函数,当 $r>0$ 时表示奖励,$r<0$ 时表示惩罚. 服务 ws 的执行使组合服务的运行状态转移到的 QoS 等级越高,则回报值 r 越大;否则回报值 r 越小. 服务组合的目标是选择最优组件服务使组合服务的累计回报值最高.

不同于已有的基于 MDP 模型的服务组合,在 SC_POMDP 模型中设置了观测集合与观测函数,根据观测集合 Z 中的元素 $\boldsymbol{z}$ 与观测函数 O,推断系统处于某个状态 s 概率. 这样做能更真实地反映在服务组合过程中组件服务执行后组合服务所处状态的多种可能状况,在选择下一组件服务时,对系统所处的状态判断更加准确.

3.2 SC_POMDP 模型中的服务选择

在 POMDP 模型的决策过程中,由于决策者对系统所处状态不完全可知,所以在模型中引入信任函数 $b(q)$,表示认为系统处于 QoS 状态等级为 $q(q\in Q_{\text{qos}})$的概率,且有 $\sum_{q\in Q_{\text{qos}}}b(q)=1$. 如当前信任状态 $\boldsymbol{b}=(0.2,0.7,0.1,0)$,表示当前的组合服务运行的 QoS 状态以 20% 的概率处于等级 A,以 70% 的概率处于等级 B,以 10% 的概率处于等级 C,不可能处于等级 D. 特别地,初始信任状态 $\boldsymbol{b}_0$ 一般是已知的.

SC_POMDP 中,执行服务 ws,引起系统状态 s 转移,决策者根据观察 $\boldsymbol{z}$ 更新信任向量 $\boldsymbol{b}$,计算回报 r_{ws},并进行下一动作的决策,如此反复,直到组合服务中的所有任务依次成功执行.

由于当前系统以概率 $b(q)$处于状态 q,选择服务 ws 向下一状态为 $s'\in S$ 转移的概率为 $T(s,ws,s')$,并以 $O(ws,s',\boldsymbol{z})$观察到 $\boldsymbol{z}$,所以能观察到 $\boldsymbol{z}$ 的总概率为

$$p_z=\sum_{q\in Q_{\text{qos}}}b(q)\sum_{s\in S}T(s,ws,s')O(ws,s',\boldsymbol{z}). \quad (1)$$

当收到新的观察后,就要更新信任状态,根据 Bayes 定理,对下一状态处于 q'的信任为

$$b'(q')=\sum_{q\in Q_{\text{qos}}}b(q)T(s,ws,s')O(ws,s',z)/p_z. \quad (2)$$

从信任状态 $b(q)$调用服务 ws 可得回报的期望值为

$$r_{ws}=\sum_{q\in Q_{\text{qos}}}\sum_{s'\in S}b(q)R(s,ws,s'). \quad (3)$$

POMDP 模型中,策略是从信任空间到动作空间的映射. 决策过程即是在策略的指导下,根据对系

统所处状态的信任函数,最大化期望回报值的过程.如果存在策略 π^*,对于信任状态 $b(q)$,采用策略 π^* 获得的回报值大于采用其他任何策略 π 所获得的回报,则称 π^* 为最优策略,其对应的值函数 V^* 为最优值函数.

考虑 SC_POMDP 中所有可能的信任状态更新和观测值,得递归方程:

$$V_{t_i}^*(\boldsymbol{b}) = \max_{ws \in A(t_i)} \left(r_{ws} + \gamma \sum_{q \in Q_{\text{qos}}} p_z V_{t_{i+1}}^*(b'(q)) \right), \tag{4}$$

$$\pi_{t_i}^*(\boldsymbol{b}) = \arg\max_{ws \in A(t_i)} \left(r_{ws} + \gamma \sum_{q \in Q_{\text{qos}}} p_z V_{t_{i+1}}^*(b'(q)) \right), \tag{5}$$

其中,r_{ws} 为执行服务 ws 所得的即时回报;p_z 是在执行第 t_i 个任务时观察到 z 的概率;$\gamma(0 \leqslant \gamma \leqslant 1)$ 是折扣因子,体现未来的回报相对近期的回报权重要低.若 $\gamma=1$,表示没有折扣,未来的回报与当前回报权重相等;若 $\gamma=0$,表示只考虑即时回报,未来回报被忽略.

SC_POMDP 模型研究的目标就是在服务组合运行中,根据对系统所处状态的判断,从候选服务集中选择使式(4)取最大值的组件服务,所选择的组件服务依次组成的组合服务即为服务组合问题的最优策略.

3.3 求解 SC_POMDP 模型的 Q 学习算法

理论上,将服务组合问题建模为 SC_POMDP 后,系统可以动态地选择最优策略,服务组合以实时最优的结果满足用户的需求.然而在实践中,由于 POMDP 模型的精确求解已被证明是 PSPACE 完全的[15],加之云计算环境中组件服务 QoS 的动态性,使其最优策略的求解非常困难.因此,我们采用学习的方法来求解最优策略.Q 学习是一种增强学习方法,通过动作执行结果的回报值,强化使系统状态更好地动作[16].该算法需要从动作的多次执行中学习,然而由于组件服务大多是商业应用,在服务组合执行过程中不可能在选择服务前多次调用候选服务以测试其性能,因此我们保存服务调用历史中服务执行的 QoS 数据,用于 Q 学习算法求解 SC_POMDP 模型时对矩阵 $\boldsymbol{Q}$ 的学习.SC_POMDP 模型的 Q 学习算法如下.

算法 1. 求解 SC_POMDP 的 Q 学习算法.

输入:任务数 n、候选服务集 A、学习率参数 α、折扣因子 γ;

输出:性能矩阵 $\boldsymbol{Q}$.

① 初始化矩阵:$\boldsymbol{Q} \leftarrow zeros(3n, m)$;

② while $\boldsymbol{Q}$ 没有达到收敛 do

③ 初始化任务序号:$t \leftarrow 1$;

④ 初始化信任向量:$\boldsymbol{b} \leftarrow (1,0,0,0)$;

⑤ while $t \leqslant n$ do

⑥ 随机选择 $A(t)$ 中服务 ws;

⑦ 执行服务 ws;

⑧ 观测到响应时间 t_r 和吞吐量 t_p;

⑨ 计算信任向量 $\boldsymbol{b}'$;

⑩ 计算回报值 $\boldsymbol{r}$;

⑪ if $b(4)<0.1$

⑫ for $i=1$ to 3

⑬ $row \leftarrow 3(t-1)+i$;

⑭ $Q(row, ws) \leftarrow (1-\alpha)Q(row, ws) + \alpha(r(i) + \gamma \max_{ws' \in R_{\text{trs}}(ws)} Q(row+i, ws'))$;

⑮ end for

⑯ 更新信任向量及任务序号:$\boldsymbol{b} \leftarrow \boldsymbol{b}'$, $t \leftarrow t+1$;

⑰ end if

⑱ end while

⑲ end while

在上述 SC_POMDP 模型的 Q 学习算法中,行①初始化矩阵 $\boldsymbol{Q}$ 为 $3\times n$ 行 m 列的全 0 矩阵,n 为组合服务的任务数,矩阵 $\boldsymbol{Q}$ 的 1 个元素代表对 1 个任务的 1 个可选服务在某种状态成功执行的回报值的累计,如 $Q(5,3)$ 表示任务 2 的第 3 个候选服务在第 2 个 QoS 等级(即等级 B)的累计回报,对状态为 D 不进行回报值累计,所以每个任务在矩阵 $\boldsymbol{Q}$ 中分 3 行累计每个候选服务的回报值.行②~⑲的循环是不断迭代计算,直到矩阵 $\boldsymbol{Q}$ 收敛,循环中止.在每趟循环中,行③初始化代表当前任务序号的变量 t 为 1,行④初始化信任向量 $\boldsymbol{b}$ 为(1,0,0,0),表示在没有执行任何服务之前,认为系统的状态是确定的,且 QoS 等级为 A.在任务没有全部执行完毕之前,执行内循环行⑤~⑱.行⑥基于 ε-贪心策略从任务 t 的候选集合中选择 1 个服务 ws,执行 ws,观察到响应时间 t_r、吞吐量 t_p 等(从 ws 的执行历史数据中随机地选取某次执行的 t_r,t_p,代表学习算法中的 1 次执行).行⑨根据式(2)及观察到的 t_r,t_p 计算信任向量 $\boldsymbol{b}'$,行⑩根据信任向量和回报函数计算选择服务 ws 完成任务 t 的即时回报.行⑪~⑰表示在服务成功执行(本文认为服务执行后若系统状态为 D 的概率小于 1,则表示服务成功执行)时,更新矩阵 $\boldsymbol{Q}$,可以执行下一任务,行⑯更新 t 为 $t+1$.否则,表示服务执行失败可能性较大,t 不变,重新为任务 t 选择

服务.其中行⑬计算当前累计值所在行 row,行⑭迭代计算矩阵 $\boldsymbol{Q}$ 第 row 行第 ws 列的元素,其中 α 为预设的学习率参数,α 越大收敛速度越快,α 越小学习效果越好,$\max\limits_{ws'\in R_{\mathrm{trs}}(ws)} Q(row+i,ws')$ 表示取下一任务候选服务集中与服务 ws 兼容,且与当前运行状态相同的矩阵 $\boldsymbol{Q}$ 元素的最大值.

4 仿真实验

为检验所提出的服务组合方法的性能,本文的实验分 3 部分:1)2.2 节场景中的服务组合问题求解,以验证模型及其求解算法的正确性;2)在真实数据集上进行测试,以验证本文所提出的方法在实际的较大规模的服务组合问题上的性能;3)对模型适应性进行测试,分别在实例问题和实际数据集中出现服务失效的情况下对 SC_POMDP 模型的适应性进行验证,并与 2 个有代表性的方法进行适应性比较.

4.1 正确性验证

首先为 2.2 节的场景建立 SC_POMDP 模型.为简单起见,设服务的可观察信息只有响应时间 t_{r},各候选服务(本例共 11 个)的历史观测响应时间的数据(本例中服务的历史数据最多有 7 次记录)构成7×11 的矩阵 $\boldsymbol{T}_{\mathrm{resp}}$ 如下所示,其中数据单位为 s.$\boldsymbol{T}_{\mathrm{resp}}$ 中每列代表 1 个服务,每行代表 1 次调用,如 $T_{\mathrm{resp}}(i,j)$ 表示第 j 个服务在第 i 次调用时的响应时间.其中,$T_{\mathrm{resp}}(i,j)=\mathrm{NaN}$ 表示第 j 个服务调用次数不足 i 次,所以 $T_{\mathrm{resp}}(i,j)$ 未知.

$$\boldsymbol{T}_{\mathrm{resp}}=\begin{pmatrix}0.502 & 0.202 & 0.717 & 0.51 & \mathrm{NaN} & 4.089 & 0.199 & 0.291 & 0.255 & 0.371 & 0.476\\ 0.69 & 0.2 & -1 & 0.378 & \mathrm{NaN} & -1 & 0.2 & 0.374 & 0.238 & 0.421 & 0.669\\ 0.686 & 0.305 & 0.6890 & 0.379 & \mathrm{NaN} & 0.358 & 0.236 & 0.451 & 0.2 & 0.358 & 0.375\\ \mathrm{NaN} & 0.32 & \mathrm{NaN} & 0.45 & \mathrm{NaN} & -1 & 0.219 & 0.365 & 0.203 & \mathrm{NaN} & 0.602\\ \mathrm{NaN} & 0.21 & \mathrm{NaN} & \mathrm{NaN} & \mathrm{NaN} & 0.671 & 0.187 & \mathrm{NaN} & 0.214 & \mathrm{NaN} & \mathrm{NaN}\\ \mathrm{NaN} & 0.2 & \mathrm{NaN} & \mathrm{NaN} & \mathrm{NaN} & \mathrm{NaN} & 0.222 & \mathrm{NaN} & 0.256 & \mathrm{NaN} & \mathrm{NaN}\\ \mathrm{NaN} & 0.19 & \mathrm{NaN} & \mathrm{NaN} & \mathrm{NaN} & \mathrm{NaN} & \mathrm{NaN} & \mathrm{NaN} & \mathrm{NaN} & \mathrm{NaN} & \mathrm{NaN}\end{pmatrix}.$$

设最大可接受的响应时间为 5 s,服务的最快响应时间为 0.1 s,所以利用

$$\boldsymbol{T}_{\mathrm{norm}}=(5-\boldsymbol{T}_{\mathrm{reps}})/(5-0.1)\tag{6}$$

对矩阵 $\boldsymbol{T}_{\mathrm{resp}}$ 进行归一化,得矩阵 $\boldsymbol{T}_{\mathrm{norm}}$.$\boldsymbol{T}_{\mathrm{resp}}$ 中元素值为 −1 时,表示服务失败,不参与归一化处理,在 $\boldsymbol{T}_{\mathrm{norm}}$ 矩阵中保持原值 −1.对 $\boldsymbol{T}_{\mathrm{norm}}$ 中元素,若 $T_{\mathrm{norm}}(i,j)\geqslant 0.95$,则对应的响应时间等级 $\boldsymbol{T}_{\mathrm{rank}}(i,j)=1$;若$0.90\leqslant \boldsymbol{T}_{\mathrm{norm}}(i,j)<0.95$,则 $\boldsymbol{T}_{\mathrm{rank}}(i,j)=2$;若 $0.85\leqslant\boldsymbol{T}_{\mathrm{norm}}(i,j)<0.90$,则 $\boldsymbol{T}_{\mathrm{rank}}(i,j)=3$;若 $\boldsymbol{T}_{\mathrm{norm}}(i,j)=-1$,则 $\boldsymbol{T}_{\mathrm{rank}}(i,j)=4$.可得如下所示的响应时间等级矩阵 $\boldsymbol{T}_{\mathrm{rank}}$.

$$\boldsymbol{T}_{\mathrm{rank}}=\begin{pmatrix}2 & 1 & 3 & 3 & \mathrm{NaN} & 4 & 1 & 2 & 1 & 2 & 2\\ 3 & 1 & 4 & 2 & \mathrm{NaN} & 4 & 1 & 2 & 1 & 2 & 3\\ 3 & 2 & 3 & 2 & \mathrm{NaN} & 2 & 1 & 2 & 1 & 2 & 2\\ \mathrm{NaN} & 2 & \mathrm{NaN} & 2 & \mathrm{NaN} & 4 & 1 & 2 & 1 & \mathrm{NaN} & 3\\ \mathrm{NaN} & 1 & \mathrm{NaN} & 2 & \mathrm{NaN} & 3 & 1 & \mathrm{NaN} & 1 & \mathrm{NaN} & \mathrm{NaN}\\ \mathrm{NaN} & 1 & \mathrm{NaN} & \mathrm{NaN} & \mathrm{NaN} & \mathrm{NaN} & 2 & \mathrm{NaN} & 1 & \mathrm{NaN} & \mathrm{NaN}\\ \mathrm{NaN} & 1 & \mathrm{NaN} & \mathrm{NaN} & \mathrm{NaN} & \mathrm{NaN} & \mathrm{NaN} & \mathrm{NaN} & \mathrm{NaN} & \mathrm{NaN} & \mathrm{NaN}\end{pmatrix}.$$

设矩阵 $\boldsymbol{Q}_{\mathrm{rank}}$ 为服务历次运行后经计算得出的综合 QoS 等级(可能由服务组合代理机构根据服务调用中各 QoS 属性值进行综合计算得出,也可能由用户反馈给出),代表对服务执行的 QoS 综合评价.其中矩阵元素 A,B,C,D 分别表示服务执行的 QoS 等级,NaN 表示无调用记录.

$$\boldsymbol{Q}_{\mathrm{rank}}=\begin{pmatrix}\mathrm{A} & \mathrm{A} & \mathrm{C} & \mathrm{C} & \mathrm{NaN} & \mathrm{C} & \mathrm{A} & \mathrm{A} & \mathrm{B} & \mathrm{A} & \mathrm{B}\\ \mathrm{B} & \mathrm{A} & \mathrm{D} & \mathrm{B} & \mathrm{NaN} & \mathrm{D} & \mathrm{A} & \mathrm{A} & \mathrm{A} & \mathrm{C} & \mathrm{C}\\ \mathrm{B} & \mathrm{A} & \mathrm{B} & \mathrm{B} & \mathrm{NaN} & \mathrm{C} & \mathrm{B} & \mathrm{C} & \mathrm{A} & \mathrm{B} & \mathrm{A}\\ \mathrm{NaN} & \mathrm{B} & \mathrm{NaN} & \mathrm{C} & \mathrm{NaN} & \mathrm{D} & \mathrm{A} & \mathrm{A} & \mathrm{A} & \mathrm{NaN} & \mathrm{B}\\ \mathrm{NaN} & \mathrm{A} & \mathrm{NaN} & \mathrm{B} & \mathrm{NaN} & \mathrm{C} & \mathrm{A} & \mathrm{NaN} & \mathrm{A} & \mathrm{NaN} & \mathrm{NaN}\\ \mathrm{NaN} & \mathrm{A} & \mathrm{NaN} & \mathrm{NaN} & \mathrm{NaN} & \mathrm{NaN} & \mathrm{B} & \mathrm{NaN} & \mathrm{B} & \mathrm{NaN} & \mathrm{NaN}\\ \mathrm{NaN} & \mathrm{A} & \mathrm{NaN} & \mathrm{NaN} & \mathrm{NaN} & \mathrm{NaN} & \mathrm{NaN} & \mathrm{NaN} & \mathrm{NaN} & \mathrm{NaN} & \mathrm{NaN}\end{pmatrix}.$$

用下面的矩阵 $\boldsymbol{C}_{\text{trs}}$ 表示图 3 所示的组件服务间兼容情况,矩阵 $\boldsymbol{C}_{\text{trs}}$ 的行由所有任务的候选服务构成,所以行数是所有任务的候选服务的总数,即集合 $A=\bigcup_{i=1}^{n}A(t_i)$ 中元素的个数. 矩阵 $\boldsymbol{C}_{\text{trs}}$ 的列代表可能的兼容服务,所以列数是每个任务的候选集大小的最大值(本例中最大候选服务集 c_2,包含 5 个候选服务).

$$\boldsymbol{C}_{\text{trs}}=\begin{pmatrix}1&1&0&0&0\\0&0&1&1&1\\1&1&0&0&0\\1&0&0&0&0\\1&0&0&0&0\\0&1&1&0&0\\1&1&1&0&0\\1&1&1&0&0\\0&0&0&0&0\\0&0&0&0&0\\0&0&0&0&0\end{pmatrix}.$$

矩阵 $\boldsymbol{C}_{\text{trs}}$ 中元素 $C_{\text{trs}}(i,j)=1$,表示第 i 行的服务与下一个任务的第 j 个候选服务兼容,否则,$C_{\text{trs}}(i,j)=0$,表示第 i 行的服务与下一个任务的第 j 个候选服务不兼容. 如第 6 行(0,1,1,0,0,0)表示 ws_{23} 与下一任务的第 2 和第 3 个服务(ws_{32} 和 ws_{33})兼容,与其他服务(ws_{31},ws_{34},ws_{35})都不兼容,即 $R_{\text{trs}}(ws_{23})=\{ws_{32},ws_{33}\}$. 最后 1 个任务的候选服务集中服务不再调用其他服务,所以相应行的元素全为 0.

各任务调用相应的候选集中的服务,回报值的计算可以根据代表回报函数的回报向量 $\boldsymbol{r}=(10,8,4,-10)$,以及服务执行后更新的信任 $\boldsymbol{b}'$ 计算. 回报向量 $\boldsymbol{r}$ 表示当 QoS 等级为 A 时,回报值为 10;当 QoS 等级为 B 时,回报值为 8;当 QoS 等级为 C 时,回报值为 4;QoS 等级为 D 时,回报变为惩罚,值为 -10.

在组合服务运行中,观测到服务的响应时间,根据式(6)及等级划分规则,可以将得到的响应时间换算为等级. 由式(2),推算出响应时间等级 $t_{\text{rank}}=i$ 时,QoS 等级 $q=k$ 的概率:

$$P(q=k\mid t_{\text{rank}}=i)=\frac{P(q=k\bigcap t_{\text{rank}}=i)}{P(t_{\text{rank}}=i)}=\frac{\boldsymbol{Q}_{\text{rank}}\text{中元素为 }k\text{ 且对应 }\boldsymbol{T}_{\text{rank}}\text{中元素为 }i\text{ 的次数}}{\boldsymbol{T}_{\text{rank}}\text{中元素 }i\text{ 出现的次数}}.\tag{7}$$

例如:在状态 $s=(1,(1,0,0,0))$,这里用信任 $\boldsymbol{b}$ 表示系统状态,即(1,0,0,0)表示当前状态的 QoS 等级为 A,B,C,D 的概率分别为 1,0,0,0. 此时,若调用服务 ws_{12} 观测到的响应时间 $t_{\text{rs}}=0.305$ s,换算为响应时间等级 $t_{\text{rk}}=2$. 根据式(7)可得:

$$P(q=A\mid t_{\text{rank}}=2)=\frac{\boldsymbol{Q}_{\text{rank}}\text{中元素为 A 且对应 }\boldsymbol{T}_{\text{rank}}\text{中元素为 2 的次数}}{\boldsymbol{T}_{\text{rank}}\text{中元素 2 出现的次数}}=\frac{7}{18}=0.3889.$$

同理可得:

$P(q=B\mid t_{\text{rank}}=2)=0.3889$,

$P(q=C\mid t_{\text{rank}}=2)=0.2222$,

$P(q=D\mid t_{\text{rank}}=2)=0$.

因此当观察到响应时间 0.305 s 时,可以判断组合服务的 QoS 等级为 A 的概率为 0.3889,QoS 等级为 B 的概率为 0.3889,QoS 等级为 C 的概率为 0.2222,QoS 等级为 D 的概率为 0,所以更新信任:

$$\boldsymbol{b}'=(0.3889,0.3889,0.2222,0).$$

由式(3),计算执行服务 ws_{12} 每个状态的回报:

$$r_{ws_{12}}=\boldsymbol{b}\cdot\boldsymbol{r}=(0.3889,0.3889,0.2222,0)\cdot(10,8,4,-10)=(3.889,3.1112,0.8888,0).$$

并更新状态为

$$s'=(2,(0.3889,0.3889,0.2222,0)).$$

在此基础上,继续下一任务的组件服务选择,直到为每个任务选择组件服务,得到最优服务组合. 照此原理,利用 Q 学习算法对该场景中服务组合问题求解,所得 $\boldsymbol{Q}$ 矩阵为

$$\boldsymbol{Q}=\begin{pmatrix}8.8090&14.1768&0&0&0\\4.4972&5.1891&0&0&0\\1.7090&1.5778&0&0&0\\8.2020&10.7469&7.4767&11.1376&8.7330\\4.9324&4.1484&5.2034&4.0419&4.9491\\1.8512&1.0339&2.1021&0.9542&1.7872\\5.4101&5.1059&4.0847&0&0\\2.5284&2.6449&2.9159&0&0\\0.7147&0.7940&1.1352&0&0\end{pmatrix}$$

矩阵 $\boldsymbol{Q}$ 表示系统 QoS 分别在 A,B,C 这 3 个状态时,每个任务的各候选服务的累计回报. 如 $Q(3,2)=1.5578$ 表示在 QoS 状态为 C 时,执行任务 1 的第 2 个候选服务(即 ws_{12})的累计回报为 1.5578. 根据该矩阵 $\boldsymbol{Q}$,为每个任务选择适应当前信任状态的最优组件服务并执行,情况如表 1 所示. 表 1 中从初始信任状态 $\boldsymbol{b}=(1,0,0,0)$,为任务 1 选择在 QoS 状态为 A 时具有最大回报值的服务 ws_{12}. 执行服务 ws_{12},信任状态更新为 $\boldsymbol{b}=(0.8125,0.1875,0,0)$. 此时,在任务 2 的候选服务集中选择与已执行服务

ws_{12}兼容的(即 $C_{trs}(2,j)=1$)且性能最优(使 $\sum_{q\in Q_{qos}}(b(q)Q(3+q,j))$取最大值)的候选服务 ws_{24},执行 ws_{24},更新信任状态,并在新的信任状态为任务 3 选择与已执行服务 ws_{24}兼容的最优的候选服务 ws_{31},形成最优组合服务为:$(ws_{12},ws_{24},ws_{31})$.

Table 1 The Belief Vectors and Selected Service for Each Task

表 1 信任向量以及为每个任务所选的服务

b	t	ws
(1.000 0 0.000 0 0.000 0 0.000 0)	1	ws_{12}
(0.812 5 0.187 5 0.000 0 0.000 0)	2	ws_{24}
(0.388 9 0.388 9 0.222 2 0.000 0)	3	ws_{31}

4.2 性能验证

为测试 SC_POMDP 在实际数据上的性能,本文在香港中文大学博士郑子彬等人[23-24]的研究成果 WS-DREAM 中的 QoSDataset2 数据集上进行实验.该数据集包括用户信息、服务信息、响应时间和吞吐量 4 个文件,本文利用其中 339 个用户对 5 825 个服务调用的响应时间数据 rtmatrix 和吞吐量数据 tpmatrix 构造了相应的 QoS 等级数据.经过大量实验,权衡收敛速度与学习效果,选择本次实验中 Q 学习的学习率为 $\alpha=0.2$,ε-贪心策略的贪心率为 $\varepsilon=0.6$,折扣因子为 $\gamma=0.8$.

4.2.1 候选服务集大小对性能的影响

固定任务数 $n=30$,令候选服务集的大小分别为 $c=10,30,50$,实验结果如图 4 所示.从图 4 可以看出,候选子集大小为 $c=10$ 时,收敛速度较快,矩阵 $\boldsymbol{Q}$ 中的最高累计回报收敛到 1 个较小的值;随着候选服务集增大,收敛速度减慢,但是矩阵 $\boldsymbol{Q}$ 中最高累计回报值增加.这与实际情况相吻合,候选集越大,矩阵 $\boldsymbol{Q}$ 元素越多,所以收敛速度减慢,同时由于候选集越大时较优秀的服务会越多一些,能选到 QoS 较优的组件服务的可能性就会增大,所以矩阵 $\boldsymbol{Q}$ 的最高累计回报收敛到更高的值.

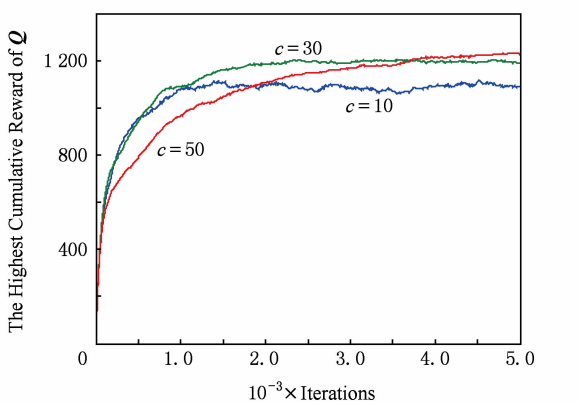

Fig. 4 The situation of different size of candidate sets.

图 4 不同大小候选服务集的情形

从图 4 还可以看出,当候选服务集大小 $c>30$ 时,矩阵 $\boldsymbol{Q}$ 的最高累计回报增长缓慢,所以可以认为,一般情况下,1 个组合服务系统,平均每个任务有 30 个以上的候选服务时,SC_POMDP 性能稳定于较高水平.

4.2.2 任务数目对性能的影响

固定每个任务的候选集大小为 $c=30$,在任务的个数分别为 n 为 10,20,30 时,实验结果如图 5 所示:

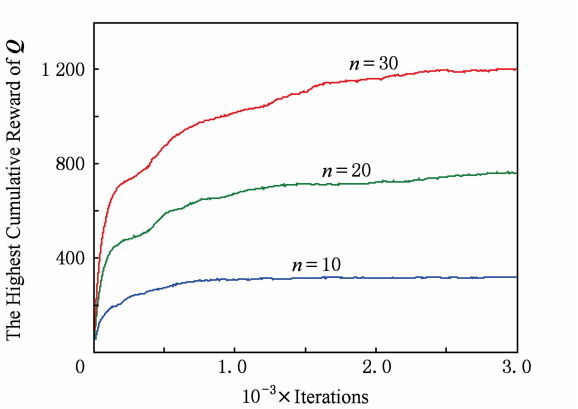

Fig. 5 The situation of different number of subtasks.

图 5 不同任务数目的情形

从图 5 可以看出,随着任务数 n 增加,收敛速度减慢,同样因为矩阵 $\boldsymbol{Q}$ 的元素增加,导致 Q 学习需要更长的时间;而随着任务数 n 增加,矩阵 $\boldsymbol{Q}$ 的最高累计回报收敛于较高水平,这是因为每个任务都要对最高累计回报有贡献,所以任务数越多矩阵 $\boldsymbol{Q}$ 的最高累计回报就越高.图 5 也表明,在任务数较多时,SC_POMDP 仍然能有效地执行.

4.2.3 组合服务性能比较

为了研究 SC_POMDP 建模服务组合问题的性能,与其他方法比较服务组合的结果,即组合服务的累计回报.Mean 方法是从历史数据中选择平均性能最好的服务,不考虑当前系统的运行状态与服务的演化情况.MDP 是 Wang 等人在文献[18]提出的服务组合模型,该模型考虑了云计算环境中服务自身的演化对服务可用性的影响,但是他们认为对系统状态是完全了解的,这是一种理想状态,实际的组合服务运行中对组合服务的运行环境以及系统的运行状态,不可能有精确的了解.而本文提出的 SC_POMDP 用 POMDP 建模服务组合问题,认为组合服务以某概率处于某个在运行状态.

图 6(a)(b)(c)(d)分别是在任务数 n 为 5,10,

20,30 时,候选服务集大小 c 从 5 增加到 10,20,30 时,3 种方法所计算的组合服务的平均累计回报值比较. 图 6 中每个取点都是在一种情况下重复 100 次实验的运行结果的平均值. 从图 6 可以看出,随着候选服务集增加,3 种方法得到的组合服务平均回报都逐渐增加,但是基于 MDP 方法的平均累计回报总是优于基于 Mean 方法,而 SC_POMDP 模型的方法平均累计回报总是优于基于 MDP 的方法,从而验证本文所提出的 SC_POMDP 模型及其求解方法有较好的性能.

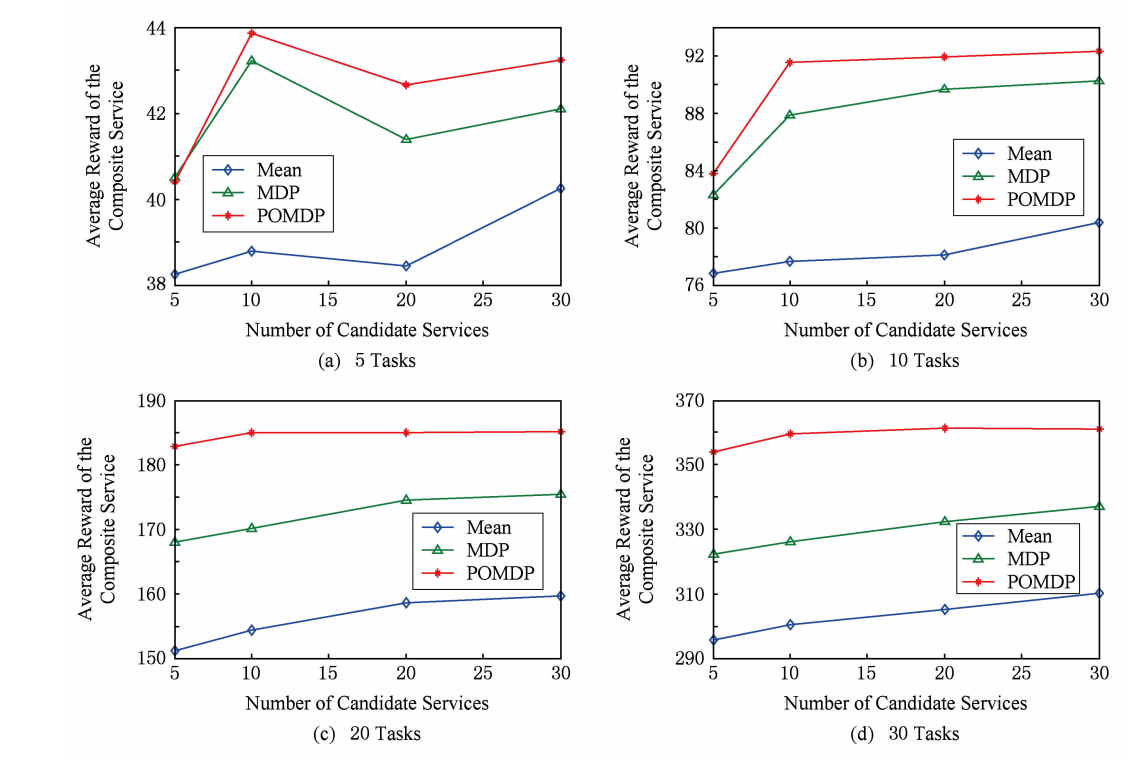

Fig. 6 The experiment results comparison of different approaches.

图 6 3 种方法的组合服务平均回报对比

图 7 是任务数 $n=30$、候选服务集大小 $c=30$ 时,3 种方法分别运行 100 次的散点图. 从图 7 可以看出,SC_POMDP 模型的运行结果都集中于值较高的区域,基于 MDP 的方法大部分运行结果集中于中间值区域,也有部分较大或较小值,而基于 Mean 的方法运行结果值较为分散,且分布在值较低的区域. 图 7 可以说明 SC_POMDP 方法不仅具有较好的平均性能,而且性能比较稳定.

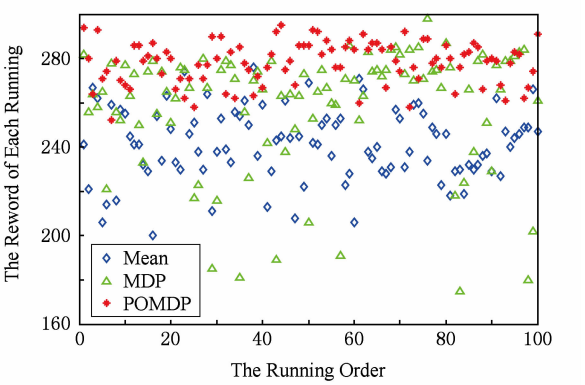

Fig. 7 The scatter plot of the 3 approaches running 100 times.

图 7 3 种方法 100 次运行结果的散点图

4.3 自适应性测试

在云计算环境中,组件服务分布在不同的云上,所以 Internet 的动态性和服务自身演化的随机性对服务组合是否能成功执行有很大的影响,自适应性是服务组合成功执行的可靠保证.

4.3.1 模拟实例中自适应性验证

为验证 SC_POMDP 的自适应性,对 1.2 节的场景,假设组合服务(ws_{12},ws_{24},ws_{31})中组件服务 ws_{24} 和 ws_{31} 均失效,在 4.1 节的实验环境下再次运行服务组合,SC_POMDP 能选择替代服务组合(ws_{12},ws_{25},ws_{32}),实验结果如表 2 所示. 根据历史观测数据学习的结果矩阵 $\boldsymbol{Q}$,选择到服务 ws_{24} 和 ws_{31} 时,由于服务失效,不能完成相应的任务,所以在信任保持不变的情况下,从正在执行的任务的候选集中选择与上一个已成功执行的组件服务相兼容

的最优服务,在选择替换服务时不仅考虑服务的性能,而且考虑服务间的兼容性,保证服务组合能成功执行,本例分别选择服务 ws_{25} 和 ws_{32} 作为替换服务.本实验说明在最优组件服务失效时,SC_POMDP 能自动选择可用组件服务参与服务组合,说明 SC_POMDP 的自适应性.

Table 2 The Service Composition when Component Service Failures Occur

表 2 组件服务失效时服务组合运行情况

b				t	ws
(1.000 0	0.000 0	0.000 0	0.000 0)	1	ws_{12}
(0.812 5	0.187 5	0.000 0	0.000 0)	2	ws_{24}
(0.812 5	0.187 5	0.000 0	0.000 0)	2	ws_{25}
(0.388 9	0.388 9	0.222 2	0.000 0)	3	ws_{31}
(0.388 9	0.388 9	0.222 2	0.000 0)	3	ws_{32}

4.3.2 实际数据上的自适应性

为验证 SC_POMDP 模型在较大规模实际数据上的自适应性,在 4.2 节的实验环境下,固定任务数为 $n=30$,令每个任务的候选服务集 c 大小分别为 10,20,30,40,50,在每个候选集上分别设定服务全部有效,20% 的服务失效和 40% 的服务失效的情况,对算法进行仿真实验.图 8 为服务组合运行中出现不同比率的服务失效的实验结果,其中纵轴为每种情况下组合服务所得累计回报值.

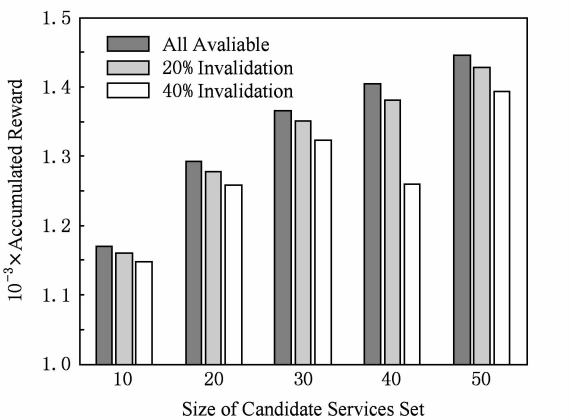

Fig. 8 The different ratio of service failures.

图 8 不同比率的服务失效的实验结果

从图 8 可以看出,组合服务的回报值会受到服务失效的影响,随着服务失效率的增长组合服务的回报值降低,但是服务组合仍然能成功执行.事实上,只要存在性能较好的可用候选服务,在失效率更高的情况下,服务组合仍然能较好地执行,这也表明 SC_POMDP 的高度自适应性.

4.3.3 与同类方法的自适应性对比实验

为检验 SC_POMDP 对服务组合过程自适应性的提升,选择 2 个具有代表性的自适应服务组合方法[17-18]进行对比实验,其中文献[17]将服务的 QoS 看作随机变量,以随机变量的均值与方差刻画服务 QoS 的随机性,本文称之为 RQoSA_SC(random QoS aware service composition);文献[18]考虑了服务演化的随机性,用 MDP 建模服务组合过程,提出了 WSC_MDP 模型并用 Q 学习方法求解.现将 RQoSA_SC 与 WSC_MDP 的方法移植到本文的实验环境中进行服务组合,并与 SC_POMDP 的服务组合结果进行对比.

实验中固定任务数 $n=20$,每个任务的候选服务集大小 $c=30$,且 3 个方法用相同的候选服务集,设置服务失效率均为 20%,分别用 RQoSA_SC,WSC_MDP 和 SC_POMDP 进行服务组合,考察组合服务的成功率以及各组件服务的 QoS(本文以响应时间与吞吐量为代表).表 3 所示的是 3 种方法在 1 次实验中所选的组件服务的响应时间与吞吐量对比,事实上几乎每次的实验结果都和表 3 的情形类似.

表 3 中 RT 代表响应时间,TP 代表吞吐量.实验结果表明 3 种方法的服务组合成功率均为 100%,表明 3 种方法都具有对服务失效的自适应能力.而一般情况,SC_POMDP 所选的组件服务有较低的响应时间与较高的吞吐量,偶尔也出现 RQoSA_SC 或 WSC_MDP 在某个任务的组件服务的响应时间或吞吐量优于 SC_POMDP 的情况,这是因为仿真环境的不确定性,不能保证每次服务组合的运行环境以及组件服务运行性能都一致.

表 4 是 3 种方法所选的组件服务的平均响应时间和平均吞吐量数据,以及最优最差响应时间和吞吐量任务数的对比.

从平均水平来看 SC_POMDP 所选的组件服务的平均响应时间为 0.090 25 s,平均吞吐量为 89.349 Kbps;WSC_MDP 所选的组件服务的平均响应时间为 0.110 05 s,平均吞吐量为 49.431 6 Kbps;RQoSA_SC 所选的组件服务的平均响应时间为 0.619 1,平均吞吐量为 46.037 55 Kbps.可以看出 SC_POMDP 有最优的平均吞吐量与响应时间.比较 3 种方法所选的组件服务响应时间最优的任务数,SC_POMDP 有 12 个,WSC_MDP 有 8 个,RQoSA_SC 有 1 个,其中 SC_POMDP 与 WSC_MDP 执行任务 13 时的响应时间与吞吐量相同,而所选的组件服

务响应时间最差的任务数 SC_POMDP 少于 WSC_MDP,WSC_MDP 又少于 RQoSA_SC. 吞吐量方面的对比有相同的结论. 因此整体上来看,SC_POMDP 有更优的自适应性.

Table 3 The Comparison of the Component Services' Response Time and Throughput Result from the 3 Methods

表 3 3 种服务组合方法所选的组件服务的响应时间与吞吐量对比

Task	Response Time/s			Throughput/Kbps		
	RQoSA_SC	WSC_MDP	SC_POMDP	RQoSA_SC	WSC_MDP	SC_POMDP
1	0.263	0.031	0.081	11.406	190	98.765
2	0.049	0.044	0.017	102.04	113.636	176.47
3	0.119	0.033	0.029	16.806	90.909	103.448
4	0.261	0.03	0.023	7.662	100	86.956
5	1.147	0.074	0.025	12.205	121.621	120
6	0.153	0.387	0.025	19.607	10.335	80
7	5.533	0.097	0.037	190	30.927	108.108
8	0.28	0.045	0.01	7.142	66.666	190
9	0.349	0.151	0.048	8.595	19.867	166.666
10	0.683	0.142	0.028	4.392	21.126	107.142
11	0.364	0.076	0.023	8.241	52.631	190
12	0.007	0.054	0.018	190	37.037	190
13	0.151	0.086	0.086	19.867	11.627	11.627
14	0.752	0.117	0.118	10.638	17.094	25.423
15	0.387	0.094	0.16	5.167	21.276	18.75
16	0.849	0.179	0.195	186.101	11.173	10.256
17	0.199	0.145	0.191	15.075	13.793	10.471
18	0.506	0.145	0.172	7.905	20.689	69.767
19	0.044	0.177	0.254	90.909	16.949	11.811
20	0.286	0.094	0.265	6.993	21.276	11.32

Table 4 The Statistic Data Comparison of the 3 Methods

表 4 3 种方法的统计数据对比

Methods	Average Response Time/s	Average Throughput/Kbps	Response Time		Throughput	
			Best Times	Worst Times	Best Times	Worst Times
RQoSA_SC	0.619 1	46.037 55	1	17	4	13
WSC_MDP	0.110 05	49.431 6	8	2	5	4
SC_POMDP	0.090 25	89.349	12	1	11	4

5 总结与展望

本文建立了一种基于 POMDP 的服务组合模型 SC_POMDP,并应用 Q 学习算法对模型进行求解. 大量实验表明,SC_POMDP 在任务众多、候选服务集较大或较小的情况下,都能有效地选择组件服务进行服务组合,与已有的方法比较,对于代表组合服务 QoS 性能的累计回报,SC_POMDP 总是优于其他方法,这是由于 SC_POMDP 对系统状态的不确定性有更真实的反映;并且在出现不同比率的服务失效的情况下,SC_POMDP 仍然能成功地执行服务组合,与同类方法的比较中,SC_POMDP 也表现出较高的自适应性.

本文工作的主要贡献在于:

1) 考虑到云计算环境的动态性和组件服务自

身演化的随机性,将组件服务选择与服务组合执行同时进行,提高了方法的自适应性;

2) 考虑到在服务组合中系统运行状态的不确定性,利用 POMDP 建模组合服务,提高了服务组合的可靠性;

3) 考虑了组件服务间的兼容性,这是在实际服务组合中必须面对而在许多研究工作中被忽视的环节.

在云计算环境中,集成已有的组件服务,构建自适应的服务组合,构成新的增值的 Web 应用,是一种新的软件构建模式. 本文提出的 SC_POMDP 模型是云计算环境中系统运行状态不确定的一般情况下,一种自适应地服务组合的方法. 未来需要进一步研究的问题是针对具体的用户需求研究满足用户 QoS 约束的服务组合.

参 考 文 献

[1] Ari I, Muhtaroglu N. Design and implementation of a cloud computing service for finite element analysis [J]. Advances in Engineering Software, 2013, 60: 122-135

[2] Rezaei R, Chiew T K, Lee S P, et al. A semantic interoperability framework for software as a service systems in cloud computing environments [J]. Expert Systems with Application, 2014, 41(13): 5751-5770

[3] Sheng Quan Z, Qiao Xiaoqiang, Vasilakos A V, et al. Web services composition: A decade's overview [J]. Information Sciences, 2014, 280: 218-238

[4] McDermott D. Estimated-Regression planning for interactions with Web services [C] //Proc of the 6th Int Conf on Artificial Intelligence Planning and Scheduling. Melon Park, CA: AAAI Press, 2002: 204-211

[5] Sheshagiri M, desJardins M, Finin T. A planner for composing services described in DAML-S [J]. Web Services and Agent-based Engineering-AAMAS, 2003, 3: 1-5

[6] McIlraith S, Son T C. Adapting golog for composition of semantic Web services [C] //Proc of the 8th Int Conf on Knowledge Representation and Reasoning. Toulouse, France: CiteSeer, 2002: 482-493

[7] Sirin E, Parsia B, Wu D, et al. HTN planning for Web service composition using SHOP2 [J]. Web Semantics: Science, Services and Agents on the World Wide Web, 2004, 1(4): 377-396

[8] He Qiang, Han Jun, Yang Yun, et al. QoS-driven service selection for multi-tenant SaaS [C] //Proc of the 5th IEEE Int Conf on Cloud Computing. Piscataway, NJ : IEEE, 2012: 566-573

[9] Alrifai M, Skoutas D, Risse T. Selecting skyline services for QoS-based Web service composition [C] //Proc of the 19th Int Conf on World Wide Web. New York : ACM, 2010: 11-20

[10] Luo Juan, Zhou Feng, Li Renfa. Dynamic service composition mechanism based on OSGi [J]. Journal of Computer Research and Development, 2014, 51(2): 420-428 (in Chinese)
(罗娟, 周峰, 李仁发. 基于分布式 OSGi 的动态服务组合算法[J]. 计算机研究与发展, 2014, 51(2): 420-428)

[11] Vu L-H, Hauswirth M, Aberer K. QoS-based service selection and ranking with trust and reputation management [C] //Proc of the Cooperative Information System Conf. Berlin: Springer, 2005: 466-483

[12] Mei Lijun, Chan W K, Tse T H. An adaptive service selection approach to service composition [C] //Proc of the 20th IEEE Int Conf on Web Services. Piscataway, NJ: IEEE, 2008: 70-77

[13] Fan Xiaoqin, Jiang Changjun, Fang Xianwen, et al. Dynamic Web service selection based on discrete particle swarm optimization [J]. Journal of Computer Research and Development, 2010, 47(1): 147-156 (in Chinese)
(范小芹, 蒋昌俊, 方贤文, 等. 基于离散微粒群算法的动态 Web 服务选择[J]. 计算机研究与发展, 2010: 47(1): 147-156)

[14] Littman M L. A tutorial on partially observable Markov decision processes [J]. Journal of Mathematical Psychology, 2009, 53(3): 119-125

[15] Hauskrecht M. Value-function approximations for partially observable Markov decision processes [J]. Journal of Artificial Intelligence Research, 2000, 13(1): 33-94

[16] Mitchell T M. Machine Learning [M]. New York: McGraw-Hill Science/Engineering/Math, 1997 (in Chinese)
(Mitchell T M. 机器学习[M]. 曾华军, 张银奎, 等译. 1 版. 北京: 机械工业出版社, 1997)

[17] Fan Xiaoqin, Jiang Changjun, Wang Junli et al. Random-QoS-Aware reliable Web service composition [J]. Journal of Software, 2009, 20(3): 546-556 (in Chinese)
(范小芹, 蒋昌俊, 王俊丽, 等. 随机 QoS 感知的可靠 Web 服务组合[J]. 软件学报, 2009, 20(3): 546-556)

[18] Wang Hongbing, Zhou Xuan, Zhou Xiang, et al. Adaptive and dynamic service composition using Q-learning [C] //Proc of the 22nd IEEE Int Conf on Tools with Artificial Intelligence. Piscataway, NJ: IEEE, 2010: 145-152

[19] Yang Jun, Lin Wenming, Dou Wanchun. An adaptive service selection method for cross-cloud service composition [J]. Concurrency and Computation: Practice and Experience, 2013, 25(18): 2435-2454

[20] Ardagna D, Pernici B. Adaptive service composition in flexible processes. [J]. IEEE Trans on Software Engineering, 2007, 33(6): 369-384

[21] Klein A, Ishikawa F, Honiden S. SanGa: A self-adaptive network-aware approach to service composition [J]. IEEE Trans on Services Computing, 2014, 7(3): 452-464

[22] Cardoso J, Sheth A, Miller J, et al. Quality of service for workflows and Web service processes [J]. Journal of Web Semantics: Science, Services and Agents on the World Wide Web, 2004, 1(3): 281-308

[23] Zheng Zibin, Zhang Yilei, Lyu M R. Distributed QoS evaluation for real-world Web services [C] //Proc of the 8th IEEE Int Conf on Web Services. Piscataway, NJ: IEEE, 2010: 83-90

[24] Zhang Yilei. Zheng Zibin, Lyu M R. Exploring latent features for memory-based QoS prediction in cloud computing [C] //Proc of the 30th IEEE Symp on Reliable Distributed Systems. Piscataway, NJ: IEEE, 2011: 1-10

Ren Lifang, born in 1976. PhD candidate, lecturer. Her main research interests include service computing and trustworthy software.

Wang Wenjian, born in 1968. PhD, professor. Senior member of China Computer Federation. Her main research interests include data mining and machine learning theory.

Xu Hang, born in 1987. PhD candidate. Her main research interests include machine learning and service computing (xuh102@126.com).