软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2014,25(6):1196−1211 [doi: 10.13328/j.cnki.jos.004456]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

一种面向多用户的负载感知动态服务选择模型*

朱 勇, 李 伟, 罗军舟

(东南大学 计算机科学与工程学院,江苏 南京 211189)

通讯作者: 朱勇, E-mail: zhuyong@seu.edu.cn

摘 要: 在面向多用户的动态环境中进行基于QoS的服务选择需要面临诸多挑战,而动态的服务负载就是其中之一.当前的服务选择方法难以在多用户多业务的开放环境下应对服务执行时的负载动态变化,缺乏实时感知负载的应变能力.针对这一问题,首先,提出一种基于负载等级的服务多维QoS模型(load level based multidimensional QoS, 简称 LLBMQoS);在此基础上,提出了一种面向多用户的负载感知的动态服务选择模型(load-aware dynamic service selection model,简称 LADSSM)以实现动态负载环境下的服务优化选择.该模型采用两阶段服务选择:在组合服务规划阶段,生成候选服务队列;在组合服务执行阶段,依据当前负载状态实现服务的动态选择;最后,仿真实验的结果表明:该模型较好地适应了多用户动态环境下的服务负载变化,能够在保证用户端到端QoS需求的前提下,及时而有效地提供效用优化的服务选择方案.

关键词: 两阶段服务选择;负载感知;负载等级;多维QoS模型;面向多用户

中图法分类号: TP311

中文引用格式: 朱勇,李伟,罗军舟.一种面向多用户的负载感知动态服务选择模型.软件学报,2014,25(6):1196−1211. http://www.jos.org.cn/1000-9825/4456.htm

英文引用格式: Zhu Y, Li W, Luo JZ. Multi-User oriented load-aware dynamic service selection model. Ruan Jian Xue Bao/ Journal of Software, 2014,25(6):1196−1211 (in Chinese). http://www.jos.org.cn/1000-9825/4456.htm

Multi-User Oriented Load-Aware Dynamic Service Selection Model

ZHU Yong, LI Wei, LUO Jun-Zhou

(School of Computer Science and Engineering, Southeast University, Nanjing 211189, China)

Corresponding author: ZHU Yong, E-mail: zhuyong@seu.edu.cn

Abstract: There are many challenges associated with service selection in a multi-user, dynamic environment. One of these challenges is the frequently changing workload. Traditional approaches to service selection, however, don't adapt to the frequently changing workload in the open and multi-user environment and can't deal with the varied workload in real time. This study is to address the problem, First, a load level based multidimensional QoS model (LLBMQoS) for services is presented; Next, a load aware dynamic service selection model (LADSSM) is proposed based on LLBMQoS to optimize service selection in dynamic environments. The model adopts a two-phase service selection framework, which generates candidate queues in design time and dynamically selects the services in terms of the current workload in execution time. Finally, simulation experiment results are provided to show the proposed model can adapt to the varied workload in a multi-user environment and provide an optimal service selection scheme while meeting the end-to-end QoS constraints.

Key words: two-phase service selection; load aware; load level; multidimensional QoS model; multi-user oriented

* 基金项目: 国家重点基础研究发展计划(973)(2010CB328104); 国家自然科学基金(61320106007); 国家高技术研究发展计划(863)(2013AA013503); 国家科技支撑计划(2010BAI88B03, 2011BAK21B02); 国家教育部高等学校博士点学科专项科研基金(20110092130002); 江苏省科技计划(BY2013095-2-07); 江苏省网络与信息安全重点实验室资助项目(BM2003201); 计算机网络和信息集成教育部重点实验室(东南大学)(93K-9)

收稿时间: 2012-06-20; 修改时间: 2012-10-24; 定稿时间: 2013-06-27

服务因其具有松耦合、与平台无关等特性,在当今开放的网络环境中得以快速发展,各种面向服务的应用大量涌现.随着服务应用规模的日益扩大,单个服务已经难以满足用户的需求.在众多功能相同但具有不同非功能属性(如 QoS 属性)的服务中选择满足用户需求的优质服务参与服务组合,已经成为研究的热点.

然而,组合服务的执行通常处于面向多个并发用户的动态环境之中,这种环境常常会导致服务质量的降级或服务组合的失败[1,2].在这种动态环境中进行基于 QoS 的服务选择与组合需要面临诸多挑战,而动态的服务负载就是其中之一[3,4].当前的服务选择方法大多面向单个用户需求,以优化基于 QoS 的组合服务效用为目标.因而,当短期内有大量的用户请求到达时,优质的服务常常会频繁地被选择,以至于发生过载或导致短时间内的服务质量降级(如响应时间延长等),甚至拒绝用户的请求.例如,基于 skyline 的服务选择方法[5]认为:选择具有较优 QoS 的服务参与组合,一定优于具有较劣 QoS 的服务.而事实上,这些优质服务可能因为被较多用户选择而使其短期内的实际性能表现并不优于具有较劣 QoS 的服务.因此,为了达成满足用户 QoS 需求的服务组合目标,在构建一个组合服务时,服务选择必须要考虑到其他用户的服务选择行为对自身服务选择造成的影响,即动态负载因素对服务选择的影响.更进一步讲,服务选择方法应该具备应对这种变化的能力.因此,在开放的、面向多用户的动态服务负载环境下,研究满足用户 QoS 需求的组合服务动态选择优化问题将更具有现实意义.

具体地讲,当前的研究存在两个方面的不足:

- 一方面,针对多个用户需求的服务选择与组合研究较少[6].大多工作是针对单个用户需求的研究,而服务运行的实际环境通常是开放的、面向多用户和多业务的(即多个组合服务),这种实际的运行环境会导致服务负载发生动态变化;
- 另一方面,已有的相关服务选择方法难以适应动态变化的服务负载.大多数服务选择方法[1,5,7−14]在组合服务设计规划阶段就确定服务选择方案.事实上,在组合服务设计规划阶段的服务状态(例如负载),可能不同于该服务被执行时的状态.对于单个用户请求(或需求)而言,这类方法生成的服务组合方案难以根据组合服务执行阶段动态变化的服务负载进行动态调整,容易造成服务过载或组合服务性能下降,缺乏实时感知负载的应变能力.

为弥补上述不足,本文将服务选择过程划分为两个阶段(即:组合服务选择设计阶段和组合服务选择执行阶段),并提出了一种负载感知的动态服务选择模型(load-aware dynamic service selection model,简称 LADSSM).该模型在组合服务规划阶段实现候选服务队列的生成;在组合服务执行阶段,依据当前负载实现服务的动态选择.该模型在满足用户端到端 QoS 的前提下,能够实时感知服务负载并进行服务的动态选择,最终实现优化的服务组合,从而弥补了当前服务选择方法的不足.

本文主要贡献可以归纳为 3 点:

(1) 提出了一种基于负载等级的多维 QoS 模型(load level based multidimensional QoS,简称 LLBMQoS);

(2) 在此基础上,提出了一种负载感知的动态服务选择模型;

(3) 提出了实现该模型优化的关键算法.

本文第 1 节介绍相关工作.第 2 节提出基本问题的场景.第 3 节提出一种基于负载等级的多维 QoS 模型.第 4 节详细阐述负载感知的动态服务选择模型,包括模型的基本组成结构、数学模型和关键算法.第 5 节进行仿真实验,并分析比较实验结果.第 6 节做出总结并提出下一步研究方向.

1 相关工作

文献[1]采用全局优化方法发现全局最优的组合服务,该方法对服务组合问题建立整数规划模型,实现了服务的优化选择;文献[5]对具有不同 QoS 的候选服务进行筛选,选出 skyline service 参与组合服务选择,这样极大地减少了候选服务的数量,提高了选择方法的效率;文献[7]针对工作流下的服务选择问题,构建了 MMKP (multidimensional multiple-choice knapsack)模型和 MCOP(multi-constrained optimal path)模型,并提出了启发式算法,从而实现了基于全局的服务优化选择;文献[8]综合了全局和局部选择的优势实现了一种混合选择方法,将全局 QoS 约束分解为局部约束,在局部选择中考虑全局因素,在实现服务优化选择的同时,显著降低了服务选择

的开销;文献[9]在文献[5]的基础上,研究了动态服务环境下基于 skyline service 的服务选择问题,提出了动态环境下 skyline service 的筛选算法;文献[10]分析了多种工作流模式下的 QoS 聚合,提出了一种优化的服务选择算法,该算法主要分为两个阶段:第 1 阶段在局部根据设定的阈值排除一些候选服务,第 2 阶段根据分支定界法生成全局优化的结果;文献[11]提出了一种基于全局 QoS 约束分解的动态服务选择方法,该方法基于模糊逻辑的自适应调整方法和自适应粒子群优化算法,将全局 QoS 约束自适应地分解为满足用户偏好的局部约束,然后再进行局部优化选择;文献[12]将服务选择全局优化问题转化为一个带约束条件的多目标服务组合优化问题,提出一种基于多目标遗传算法的 QoS 全局优化选择方法,最终产生一组满足约束条件的非劣服务聚合流程集;文献[13]将服务组合模型建立在线性规划的基础上,引入 QoS 协商机制发现可行路径,并进一步优化了服务选择;文献[14]对候选服务事先按照一定规则进行筛选,通过预先筛选,减少了可供选择的服务数量,再采用基于全局优化的线性规划方法生成最终的组合方案.在上述研究中,各服务组合过程独立地进行选择优化,优质的服务会被多个组合服务选中,从而在这些服务节点可能出现过载,导致服务组合性能降级或失败.而失败后,无论采用重规划的方法还是使用备份服务的方法,都无法保证新生成的组合服务不会发生上面提到的问题.因而,以上的方法无法适用于多用户的动态负载环境.

文献[15]在服务选择中引入信任,提出了基于蚁群算法的服务动态选择算法,在组合规划中避免所有组合服务的执行都经过同一条路径.但该研究未考虑单个服务节点上的负载变化,因而难以保证单个节点上不发生过载和性能的下降;文献[16]研究了基于 per-flow 的优化组合问题,在预测用户任务到达速率的基础上,对多个组合服务选择方案统一进行规划,但对于同一个流中的用户请求该文献仅考虑固定统一的全局 QoS 约束,未考虑用户 QoS 需求的多样性;文献[6]研究了满足多个用户需求的服务组合问题,对多个用户需求进行可满足性折中优化;文献[17]针对多个用户请求提出了基于整数规划的服务选择优化方法;文献[18]以非合作博弈的方法来解决多用户的服务选择优化问题.以上研究的不足之处在于:一是采用离线方式进行优化,假设事先就能够获取所有用户请求信息,而在某些实际应用中用户请求的到达具有随机性和突发性,难以提前获得,因而需要采用在线优化的方式加以解决;二是在服务组合的规划阶段就生成确定不变的服务选择方案,这种方式难以适应在开放的、多用户、多业务环境下组合服务执行时的负载动态变化.

此外,在服务负载均衡方面,文献[19]按照不同的概率在多个服务中选择一个服务来执行请求,以实现系统的总体平均响应时间最小;文献[20]针对服务覆盖网络中服务选择和负载均衡问题,通过对候选服务的负载状况进行测算,从而建立一条适当的组合服务路径,实现了多个服务副本之间的负载均衡.然而,以上研究均未考虑用户端到端的多维 QoS 需求.

2 基本问题场景

一个用户需求通常可划分为多个任务,例如,一个旅游规划需求包括了天气查询、预订机票等任务.假设一个用户需求由一组任务 $T_1,\ldots,T_n$ 组成,并且一个任务 $T_j(1\leqslant j\leqslant n)$存在一组服务可以完成,这些服务具有相同功能属性和不同的 QoS.它们组成的集合表示为服务类 S_j,$S_j=\{s_{1j},\ldots,s_{mj}\}$.可见,实体服务 $s_{ij}(1\leqslant i\leqslant m)$是服务类 S_j 的一个成员,表示一个具体服务或服务类 S_j 的实例.

定义 1(虚拟组合服务流程 VCSF(virtual composite service flow)). 是由一组服务类构成并且各服务类之间具有明确拓扑关系的组合流程模板.VCSF 可以用 DAG(directed acyclic graph)图来表示,通过组合不同的服务类,可以实现更加复杂的业务功能.

定义 2(服务负载 l_{ij}). 表示服务 s_{ij} 当前并发处理的用户请求数量.一个服务可以承受的最大并发请求数量(即最大负载)表示为 $l_{ij}^{\max}$.当并发请求数量超过最大负载时,服务将无法工作或者拒绝部分请求.

一个用户请求(或需求)由两部分组成:一是虚拟组合服务流程 VCSF(即功能性需求);另一个是用户端到端的 QoS 需求(即非功能性需求).在多用户的应用环境中,用户的需求是多样的,这表现在两个方面:一是功能需求的多样化,这意味着开放的服务运行环境存在着多个 VCSF,并且这些 VCSF 在某些任务节点可能相交;二是 QoS 需求的多样化,这意味着不同的用户有着不同的 QoS 需求.图 1 描述了这样一个应用场景,存在两类有不同功能

需求的用户:一类需求是旅游规划,对应于 VCSF:$S_1 \rightarrow S_3 \rightarrow S_4$;另一类是会议规划,对应于 VCSF:$S_2 \rightarrow S_3 \rightarrow S_5$.两类 VCSF 存在一个交集 S_3.以往的服务选择大多不考虑多个用户需求,而在图 1 所示的多 VCSF 场景中,由于存在交集 S_3,考虑到服务的负载因素,对一个 VCSF 的服务选择可能会影响到另一个 VCSF 的服务选择;并且在同一个 VCSF 中,针对一个用户请求的服务选择也会影响另一个用户请求的服务选择结果.此外,在开放的、多用户的服务环境中,用户请求到达难以提前预知,因此需要以在线的方式加以优化处理.

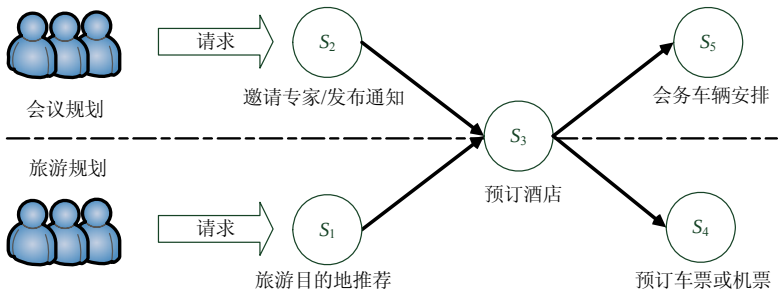


Fig.1 Multi-User oriented application scenario of composite services

图 1 面向多用户的组合服务应用场景

本文所讨论的问题可以描述为:根据依次到达的在线用户请求(即多种 VCSF 和端到端的 QoS 约束),在感知服务负载的基础上实现在线优化服务选择.为了便于描述和理解,本文给出如下假设:

假设 1. 在一个虚拟组合服务流程 VCSF 中,共有 n 个服务类 $S_j(1 \leqslant j \leqslant n)$,并且每个 S_j 中有 m 个实体服务 s_{ij} $(1 \leqslant i \leqslant m)$.QoS 向量的维度为 r,由于 QoS 通常是多维的,故本文设 $r \geqslant 2$.

假设 2. 用户请求按一定平均速率到达,请求的内容(即功能需求和非功能需求)事先不可知;并且各用户请求的优先级相同,依据先来先服务的原则进行服务选择与执行.

3 基于负载等级的 QoS 模型

服务 QoS 是选择服务的主要依据,优质的服务常常会频繁地被选择.然而,一个支撑服务运行的硬件设备不可能具有无限多的资源和无限大计算能力,这造成了两个方面的影响:一是服务受到本地资源能力的限制,因而需要限制服务的最大负载,以防止服务过载;二是在不同的负载状态下,服务性能或 QoS 会发生变化,因而需要对服务在不同负载状态下的 QoS 进行表示和评估.本文提出了一种基于负载等级的多维 QoS 模型(LLBMQoS).

3.1 基本定义

服务 s_{ij} 的负载区间由一个负载上界和负载下界组成,其中,d 为一个服务负载区间标识,表示该区间的负载等级,其值为大于 0 的整数,它反映了服务繁忙的状态与程度,如轻负载、中等负载和重负载,其等级越高(即 d 值越大),说明服务当前负载越大.显然,最大负载等级的负载上界为 $l_{ij}^{\max}$.

定义 3. 当一个服务 s_{ij} 具有 z 个负载等级时,基于负载等级的多维 QoS 模型 LLBMQoS 可表示为

$$\langle Q^{(1)}(s_{ij}),\ldots,Q^{(z)}(s_{ij})\rangle,$$

其中,$Q^{(d)}(s_{ij})$是服务 s_{ij} 在负载等级 $d(1 \leqslant d \leqslant z)$下的 r 维 QoS 向量,$Q^{(d)}(s_{ij})=[q_1^{(d)}(s_{ij}),...,q_r^{(d)}(s_{ij})]$,其中,$q_k^{(d)}(s_{ij})$ 表示服务 s_{ij} 在负载等级 d 下第 $k(1 \leqslant k \leqslant r)$个 QoS 属性的值.

典型的服务 QoS 属性有响应时间、可靠性、可用性、成本等,其具体定义见文献[1,10].

QoS 属性根据其方向特征包括积极度量和消极度量两类:积极度量的 QoS 属性是指其值越大,其反映的质量属性越高,反之越差,如可靠性、可用性等;消极度量的 QoS 属性是指其值越大,其反映的质量属性越差,反之越高,如响应时间、服务成本等.

事实上,属于积极度量类的 QoS 属性值可以通过乘以–1 而转化成消极度量类的属性[8].

本文假设所有的 QoS 属性是消极度量类的或者已经转化为消极度量类.为描述简洁,本文后面的叙述如非特殊说明,都将采用这一假设.

服务通常运行在不同的负载等级下,其 QoS 表现也各不相同.常用的 QoS 属性如响应时间、可用性、可靠性、价格等,它们与负载等级之间的关系可以分为相关和不相关两类:

- 与负载等级相关的属性有响应时间、可用性和可靠性等,随着负载等级的增加,这些 QoS 值可能会发生变化.这种变化大致可以分为两类,即正相关变化和负相关变化:
  - 正相关变化即随着负载增加,QoS 值可能会逐步增加.例如,响应时间在重负载环境下的值要高于低负载状态下的值[4];
  - 负相关变化即随着负载增加,QoS 值可能会逐步减小.例如,可靠性和可用性会随负载增加而出现下降的趋势[3],这是由于随着负载的增加,部分请求的响应时间超过最大限度,从而导致响应失败而降低了服务可靠性;另一方面,由于负载的增加导致了服务过载概率的增加,从而降低了服务的可用性;
- 与负载等级不相关的属性主要是服务价格等.

本文主要考虑与负载等级相关的 QoS 属性,因此,采用基于负载等级的多维 QoS 模型可以以较细的粒度展现不同负载环境下的服务 QoS,为服务选择提供符合实际环境的决策依据.

为了定量地描述负载与 QoS 之间的关系,需要构建基于负载等级的多维 QoS 模型.通常可以采用两种方法:一种是由服务提供商在服务正式发布前对服务进行不同等级的压力测试,获取不同负载等级下的 QoS 值,并与其他服务注册信息一同发布;另一种是在服务运营过程中,通过服务监测获取不同负载等级下的 QoS 值.本文不研究服务 QoS 的测试与监测获取问题,并假设这些数据已知.

3.2 基于QoS的效用

在 VCSF 已知的前提下,为了实现服务的优选和组合服务的优化评估,需要计算基于 QoS 的效用,本文采用文献[1]的方法对 QoS 进行归一化处理,并计算效用值.

定义 4. $v_{ijk}^{(d)}$ 表示经过归一化处理的 $q_k^{(d)}(s_{ij})$,其计算方法如下:

$$v_{ijk}^{(d)}=\begin{cases}\dfrac{Q\max_j^{(k)}-q_k^{(d)}(s_{ij})}{AQ\max^{(k)}-AQ\min^{(k)}}, & AQ\max^{(k)}\neq AQ\min^{(k)}\\ 1, & AQ\max^{(k)}=AQ\min^{(k)}\end{cases} \tag{1}$$

其中,$Q\max_j^{(k)}$ 表示服务类 S_j 中在各负载等级下所有服务的第 k 维 QoS 属性的最大值,$Q\min_j^{(k)}$ 表示服务类 S_j 在各负载等级下所有服务的第 k 维 QoS 属性的最小值,$AQ\max^{(k)}$表示在当前 VCSF 中第 k 维聚合 QoS 属性的最大值,$AQ\min^{(k)}$表示在当前 VCSF 中第 k 维聚合 QoS 属性的最小值.具体计算如下:

$$Q\max_j^{(k)}=\max_{1\leqslant d\leqslant z\wedge s_{ij}\in S_j}(q_k^{(d)}(s_{ij})) \tag{2}$$

$$Q\min_j^{(k)}=\min_{1\leqslant d\leqslant z\wedge s_{ij}\in S_j}(q_k^{(d)}(s_{ij})) \tag{3}$$

$$AQ\max^{(k)}=G_k(Q\max_1^{(k)},...,Q\max_n^{(k)}) \tag{4}$$

$$AQ\min^{(k)}=G_k(Q\min_1^{(k)},...,Q\min_n^{(k)}) \tag{5}$$

其中,$G_k(\cdot)$表示在当前 VCSF 中,关于 n 个服务的第 k 维聚合 QoS 函数.本文考虑 4 种基本的虚拟组合服务流程结构:顺序、并发、选择和循环,这 4 种基本结构还可以通过组合形成更复杂的流程结构,其聚合 QoS 的计算方法在文献[21]有详细的讨论,本文不再赘述.

定义 5. $u_{ij}^{(d)}$ 表示实体服务 $s_{ij}(1\leqslant i\leqslant n,1\leqslant j\leqslant m)$在负载等级 d 下的效用,计算如下:

$$u_{ij}^{(d)}=\sum_{k=1}^{r}v_{ijk}^{(d)}\omega_k \tag{6}$$

其中,ω_k 为各 QoS 属性的权重,且 $\sum_{k=1}^{r}\omega_k=1$.

定义 6. u_{ij} 表示实体服务 s_{ij}($1\leqslant i\leqslant n,1\leqslant j\leqslant m$)的期望效用,计算如下:

$$u_{ij}=\sum_{d=1}^{z}p_{ij}^{(d)}u_{ij}^{(d)} \tag{7}$$

其中, $p_{ij}^{(d)}$ 表示服务 s_{ij} 处于负载等级 d 的概率.

以上考虑的仅仅是局部 QoS 的归一化处理以及单个服务效用的计算,然而,用户 QoS 需求表现为组合服务的 QoS 需求.用户能够感知到的是组合服务的聚合 QoS,而非单个实体服务的 QoS 及其效用.因此,需要获取实际执行时的服务 QoS 并对组合服务的聚合 QoS 进行归一化处理,以便进一步计算组合服务的实际效用.

定义 7. U 表示由实体服务构成的组合服务在实际负载环境下的执行效用.假设组合服务 CS 由 n 个原子服务构成,可表示为 $\langle s_{i_1,1},...,s_{i_n,n}\rangle$,并且在该组合服务的执行过程中,各原子服务执行时的负载等级分别对应为$\langle d_1,d_2,\ldots,d_n\rangle$,则执行效用按式(8)计算:

$$U=\sum_{k=1}^{r}V_k\omega_k \tag{8}$$

其中,V_k 表示表示经过归一化处理的第 k 维聚合 QoS 值,其计算方法如下:

$$V_k=\begin{cases}\dfrac{AQ\max^{(k)}-G_k(q_k^{(d_1)}(s_{i_1,1}),...,q_k^{(d_n)}(s_{i_n,n}))}{AQ\max^{(k)}-AQ\min^{(k)}}, & AQ\max^{(k)}\neq AQ\min^{(k)}\\ 1, & AQ\max^{(k)}=AQ\min^{(k)}\end{cases} \tag{9}$$

4 负载感知的动态服务选择模型

4.1 基本模型

组合服务的生命周期可以大致分为两个部分:一是设计阶段,在这一阶段生成服务组合方案;二是组合服务执行阶段,在这一阶段依据前面的方案执行组合服务.

传统的服务选择只涉及第 1 阶段,而忽略了组合服务执行阶段的服务状态变化,从而造成了服务选择(或组合)方案难以适应动态变化的环境(如服务负载).本节将服务选择方法拓展到了组合服务的整个生命周期,提出了一种两阶段服务选择模型.这两个阶段分别为:

- 在服务组合规划阶段(即第 1 阶段)进行服务选择预规划,生成候选服务队列 CSQ(candidate service queue);
- 在组合服务执行阶段(即第 2 阶段),依据 CSQ 以及相关服务的当前负载状态进行运行时的动态选择.

具体过程是(如图 2 所示):

1) 当用户请求到达时,服务选择预规划模块根据用户的功能性需求(即 VCSF)选择相关的虚拟服务池,也就是对应的服务类;
2) 从中获取候选服务的 QoS 信息;
3) 在每个相关的服务类中按一定策略选取若干个候选服务,生成候选服务队列 CSQ,并发送给对应的动态选择代理;
4) 当组合服务执行到某个任务(或服务类)节点时,相应的动态选择代理根据候选服务队列中候选服务的当前负载状态,选择局部效用优化的实体服务参与组合.

其中,步骤 1)~步骤 3)属于服务选择的第 1 阶段;步骤 4)属于服务选择的第 2 阶段.

值得一提的是,在本模型中,可以同时设置多个服务选择预规划模块来为不同的用户或业务服务,以防止单个模块带来的性能瓶颈问题.

模型主要由 3 个部分组成:虚拟服务池集合、服务选择预规划模块和动态选择代理集合,如图 2 所示.

- 虚拟服务池集合由若干个虚拟服务池组成,每个虚拟服务池聚集了功能相同但非功能属性(如 QoS)不同的一组实体服务集合,一个虚拟服务池对应一个服务类 S_j;
- 服务选择预规划模块通过对服务选择进行预处理,在各服务类 S_j 的原始候选服务中,筛选出满足一定条件的候选服务生成候选服务队列 CSQ_j,并分发给对应的动态选择代理.具体的算法将在后面讨论;
- 动态选择代理集合由多个动态选择代理组成,每个动态选择代理与服务类和虚拟服务池一一对应.当组合服务执行到该服务类时,对应的动态选择代理从该组合服务的 CSQ_j 中按一定的算法选择实体服务参与组合服务的执行.具体的算法将在后面讨论.

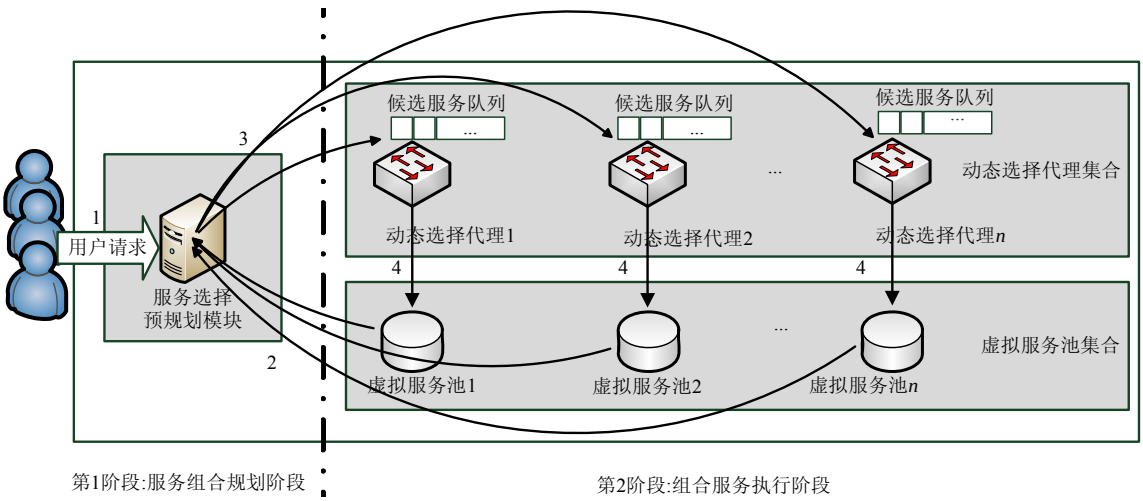


Fig.2 Two-Phase service selection

图 2 两阶段服务选择

动态选择代理主要由 3 个部分组成:负载统计模块、动态选择模块和候选服务队列集合.图 3 显示了各部分之间的关系:

1) 当任务到达时,触发服务选择;
2) 动态选择模块从候选服务队列集合中选取属于该服务类 CSQ_j;
3) 动态选择模块从负载统计模块获取有关实体服务的当前负载等级;
4) 输出服务选择结果;
5) 如果结果不为∅,负载统计模块需要根据服务被调用的情况更新该服务的负载状态.

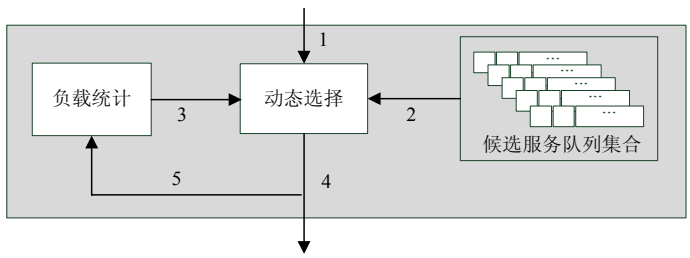


Fig.3 Dynamic selection agent

图 3 动态选择代理

具体方法是:当调用该服务时,该服务的并发任务数量加 1;当该服务的一个任务结束(即动态选择代理收到请求的响应)时,其并发任务数量减 1.值得一提的是,本模型假设了虚拟服务池中的实体服务只能由动态选择代

理来选择调用,而不能由其他用户直接选择调用,如此才能保证统计负载的真实性.

图 4 显示了两阶段动态服务选择过程的一个实例(为保持过程的简洁,该图删除了从虚拟服务池获取 QoS 数据的过程).在该实例中共有 5 个服务类(对应的有 5 个虚拟服务池和 5 个动态选择代理),每个服务类有若干个实体服务.这些服务类共形成了 2 个 VCSF(见表 1),即代表了两种不同的组合服务功能(或者业务功能).同时,在该实例中为每个VCSF分别设置了一个服务选择预处理模块,用来服务于使用该VCSF的用户.在任意一个服务(或任务)节点 S_j 的执行过程中,选择代理依据一定策略从对应的服务候选队列中选择一个服务绑定执行.

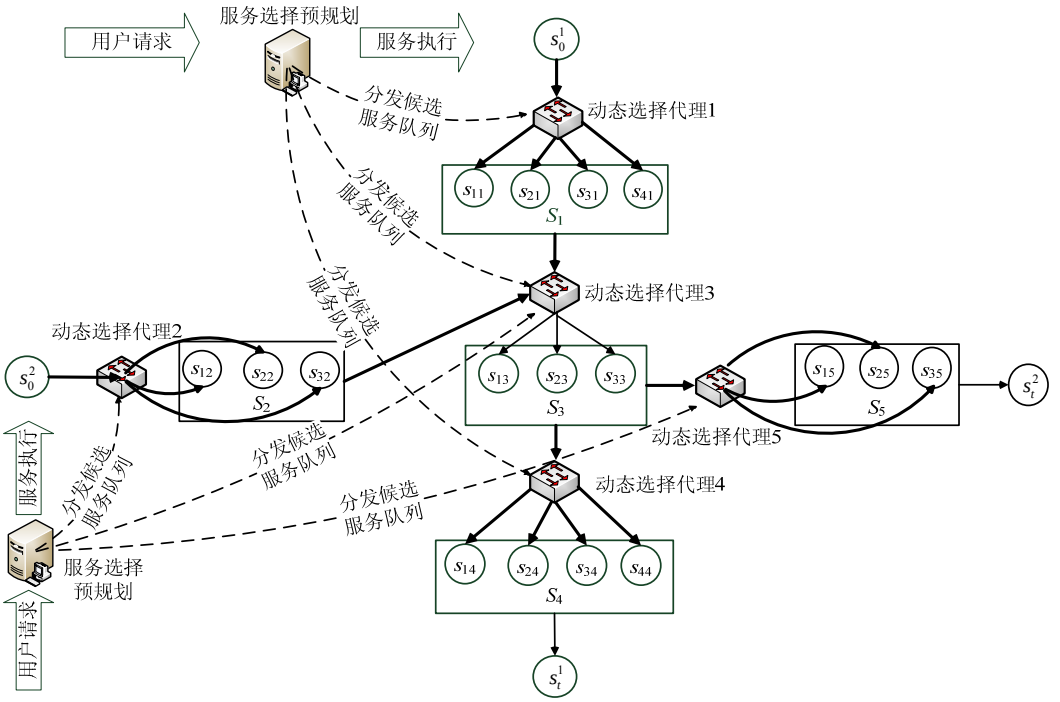


Fig.4 An example of two-phase service selection

图 4 两阶段的动态服务选择过程实例

Table 1 Two examples of VCSF

表 1 两个 VCSF 的例子

No.	VCSF
1	s_0^1(VCSF 1 的起点)→S_1→S_3→S_4→ s_t^1(VCSF 1 的终点)
2	s_0^2(VCSF 2 的起点)→S_2→S_3→S_5→ s_t^2(VCSF 2 的终点)

在动态的运行环境中,为了实现负载敏感的服务优化选择,本文提出了一种基本的动态服务选择优化数学模型.该模型可以表示为公式(10):

$$\begin{cases} Object\ \max imize\ U \\ Subject\ to\ G_k(AQ_k(s_{i_1,1}),...,AQ_k(s_{i_n,n})) \leqslant Q_k^{\max},\ 1 \leqslant k \leqslant r \\ AQ_k(s_{ij}) = \sum_{i=1}^{m}\sum_{d=1}^{z} x_{ij}^{(d)} q_k^{(d)}(s_{ij}), \qquad x_{ij}^{(d)} = \{0,1\};1 \leqslant k \leqslant r;1 \leqslant j \leqslant n \\ \sum_{i=1}^{m}\sum_{d=1}^{z} x_{ij}^{(d)} = 1, \qquad x_{ij}^{(d)} = \{0,1\};1 \leqslant j \leqslant n \end{cases} \tag{10}$$

其中:

- $AQ_k(s_{ij})$为执行第 j 个任务(即服务类 S_j)时的第 k 个 QoS 属性值,该值与服务执行时其所处的负载状态有关;
- $x_{ij}^{(d)}$ 是二元变量,取值为 0 或 1:当值为 1 时表示 s_{ij} 被选中,并且该服务被执行时,其负载等级为 d;否则,其值为 0;
- $Q_k^{\max}$ 表示第 k 个 QoS 属性的全局约束或最大值(假设所有属性均为消极度量的).

在上面的模型中,目标为最大化组合服务的执行效用 U.约束条件为组合服务的各 QoS 属性需满足全局 QoS 约束,并且每个任务只能由一个服务来完成.同时,在一次服务执行过程中,一个服务只能处于一种负载等级.显然,该模型属于经典的 NP-难问题.然而,服务负载在运行阶段常常发生不可预知的动态变化,在组合服务的设计阶段难以直接对服务选择进行优化.因而,本文在两阶段服务选择模型下提出了相应的优化关键算法.

4.2 模型的关键算法与分析

本模型涉及到的关键算法有候选服务队列生成算法和动态选择算法,其中,候选服务队列生成算法部署在服务选择预规划模块,动态选择算法部署在动态选择代理中.

4.2.1 候选服务队列生成算法

该算法将全局端到端的 QoS 约束分解为局部 QoS 约束 LQ_j(即每个参与组合的服务类 S_j 的 QoS 约束),在各服务类节点 S_j 挑选出可能满足局部 QoS 约束的候选服务,生成候选服务队列 CSQ_j.

该算法的核心是解决如何将全局端到端的 QoS 约束分解为对每个服务类的局部约束.本文将该问题定义为一个多维约束下的极大极小优化问题,其数学模型如下:

$$\begin{cases}\max \min\limits_{1\leqslant j\leqslant n}(F(LQ_j)) \\ \text{subject to } G_k(LQ_1^{(k)},...,LQ_n^{(k)})\leqslant Q_k^{\max},\ 1\leqslant k\leqslant r\end{cases} \tag{11}$$

其中,

- $LQ_j=[LQ_j^{(1)},...,LQ_j^{(r)}](1\leqslant j\leqslant n), LQ_j^{(k)}$ 表示在服务类 S_j 的第 $k(1\leqslant k\leqslant r)$维局部 QoS 约束,并且
$$LQ_j^{(k)}\in\{q_k^{(d)}(s_{ij})\,|\,1\leqslant i\leqslant m;1\leqslant d\leqslant z\};$$
- $Q_k^{\max}$ 表示第 k 维的全局 QoS 约束;
- $G_k(\cdot)$表示第 k 维的聚合 QoS 函数.

约束条件保证了满足局部约束的服务组合后,其聚合 QoS 仍然满足用户的 QoS 需求(假设所有 QoS 属性都是消极度量类的).

$F(LQ_j)$表示关于局部 QoS 约束的目标函数,计算如下:

$$F(LQ_j)=\sum_{i=1}^{m}P_{ij}(LQ_j) \tag{12}$$

其中,$P_{ij}(LQ_j)$为服务 s_{ij} 在所有负载等级下满足局部约束 LQ_j 的概率,计算公式如下:

$$P_{ij}(LQ_j)=\sum_{d=1}^{z}p_{ij}^{(d)}Y(Q^{(d)}(s_{ij}),LQ_j) \tag{13}$$

其中,$Y(Q_{ij}^{(d)},LQ_j)$ 为指示函数,当值为 1 时表示服务 s_{ij} 的 $Q^{(d)}(s_{ij})$满足局部约束 LQ_i;当值为 0 时表示不满足局部约束:

$$Y_{ij}(Q^{(d)}(s_{ij}),LQ_j)=\begin{cases}1, & \forall k, q_k^{(d)}(s_{ij})\leqslant LQ_j^{(k)}(1\leqslant k\leqslant r)\\ 0, & \text{else}\end{cases} \tag{14}$$

以上数学模型可以描述为:将一个全局 QoS 约束分解为对多个服务类(即虚拟服务池)的局部 QoS 约束 LQ_j $(1\leqslant j\leqslant n)$,使得目标函数 $F(LQ_j)$的最小值最大化.这种分解方法的好处在于,考虑到了参与组合的服务有可能因负载原因而使服务组合性能降级或过载.不同于以往只考虑全局求和的情况,本文提出的分解方法着重考虑优化某些可能会形成瓶颈的服务类节点,以有效地应对可能出现的性能降级或过载的情况,最终使得在这些服务类节点能够提供更多的服务选择,而并非生成单一的基于全局优化的服务选择结果.

根据假设 1 和文献[22]可知,公式(11)的模型可以表示为一个多约束路径优化问题(multi-constrained optimal path,简称 MCOP),至少属于 NP 难问题.本文设计了一种启发式算法(CSQGen),实现对这一模型的求解.在介绍本算法之前,本文首先引入与算法相关的定义.

定义 8(QoS 递增量). 由Δrs_j表示,是对更新前和更新后的服务类 S_j 局部约束向量增加程度的度量,其定义如下:

$$\Delta rs_j=(LQ'_j-LQ_j)\left[\frac{Q_a^{(1)}}{Q_1^{\max}},\ldots,\frac{Q_a^{(r)}}{Q_r^{\max}}\right]^T \tag{15}$$

其中,LQ'_j 和 LQ_j 分别表示服务类 S_j 更新后和更新前的局部约束向量,$Q_a=[Q_a^{(1)},\ldots,Q_a^{(r)}]$ 为更新前已聚合的 QoS 约束向量,有:

$$Q_a^{(k)}=G_k(LQ_1^{(k)},\ldots,LQ_n^{(k)}),1\leqslant k\leqslant r \tag{16}$$

值得一提的是,为了提高算法的效率,使其能够适应较大的服务规模,减少计算时间成本以及分发时的数据量,本模型对生成的候选服务队列 CSQ 的最大长度做了限制.本文用 h 表示候选服务队列的最大长度,即 CSQ 最多能够容纳的实体服务数量.显然,$h\leqslant m$.

在算法 1 的 Step 1 中,需要遍历每个服务类中 m 个候选服务;又因为共有 n 个服务类,故其时间复杂度为 $O(nmzr)$.同理,Step 2 的时间复杂度也为 $O(nmzr)$.Step 3~Step 7 每次循环的结果是增加了一个满足局部约束的 QoS,故对于 n 个服务类,总共需要循环次数的上界为 nzh;单次循环的时间复杂度为 $O(nr+mzr)$,故 Step 3~Step 7 的时间复杂度为 $O(nhmrz^2+rzhn^2)$.Step 8 的时间复杂度 $O(nh\log h)$.

综上所述,其总体的时间复杂度为 $O(nmrhz^2+rzhn^2+nh\log h)$.由于 $h\leqslant m$,当 m 较大时,需要恰当地选择 h 值,确保维持较短的计算时间.

算法 1. 候选服务队列生成算法(CSQGen).

输入:全局 QoS 约束 $Q^{\max}$,$S_1,\ldots,S_n$,h;

输出:$CSQ_1,\ldots,CSQ_n$ 或者为 *null*.

Step 1. 对每个 S_j 和 QoS 属性 k,令 $LQ_j^{(k)}=\min\{q_k^{(d)}(s_{ij})\}$ $(1\leqslant i\leqslant m;1\leqslant d\leqslant z)$,检查初始向量组 $LQ_1,\ldots,LQ_n$ 是否满足全局约束 $Q^{\max}$;若满足,则进入 Step 2;否则,候选服务队列生成失败,输出 *null*,退出算法;

Step 2. 对每个服务类 S_j,将 S_j 中在各负载等级下都满足当前 LQ_j 约束的实体服务加入到 CSQ_j,其他的实体服务加入集合 Φ_j,并计算其目标函数值 $F(LQ_j)$;

Step 3. 选择目标函数值最小的服务类 S_c;

Step 4. 如果 CSQ_c 队列长度$\geqslant h$ 或者 Φ_c 为∅,执行 Step 8;

Step 5. 对 Φ_c 中各实体服务 s_{ic} 的所有 QoS 属性 $k(1\leqslant k\leqslant r)$,令:

$$LQ_{ic}'^{(k)}=\max\{LQ_c^{(k)},q_k^{(d)}(s_{ic})\}\ (1\leqslant d\leqslant z) \tag{17}$$

其中,$LQ'_{ic}=[LQ_{ic}'^{(1)},LQ_{ic}'^{(2)},\ldots,LQ_{ic}'^{(r)}]$,表示候选的新局部 QoS 约束,它是根据 S_c 中第 i 个实体服务 s_{ic} 的 QoS 属性所生成的;

Step 6. 将替换后仍然满足全局约束的 LQ'_{ic} 组成局部约束候选集合 ψ_c,如果 ψ_c 为∅,执行 Step 8;

Step 7. 在 ψ_c 中选择具有最小Δrs_c的 LQ'_{ic};更新局部约束 $LQ_c=LQ'_{ic}$,计算新的目标函数值;如果 s_{ic} 在各负载等级下的 QoS 均满足 LQ_c,则将 s_{ic} 加入 CSQ_c,并从 Φ_c 中删除 s_{ic};返回 Step 3;

Step 8. 遍历每个 CSQ_j,如果 CSQ_j 的长度等于 h,直接输出 CSQ_j,算法结束;否则,对于服务类 S_j 中每一服务,按其 $P_{ij}(LQ_j)$的值(即满足局部约束 LQ_j 的概率)进行非递增排序,对于 $P_{ij}(LQ_j)$函数值相同的服务,按其 u_{ij} 值非递增排序(可采用快速排序算法),并获取排在前面的 h 个服务组成 CSQ_j,输出 CSQ_j,算法结束.

4.2.2 动态选择算法

组合服务执行过程中,当需要执行某个任务时,在该任务所对应服务类 S_j 中触发动态选择算法.该算法的主

要思想是:根据服务的当前负载等级 d,从 CSQ_j 中选出满足局部 QoS 约束(即 LQ_j)且效用 $u_{ij}^{(d)}$ 最大的服务执行该任务,即在满足局部约束的前提下,尽量选择当前负载等级下效用值最高的服务参与组合.

从算法 2 中可以看出:由于候选服务队列 CSQ_j 最大长度为 $h(h\leqslant m)$,故其时间复杂度为 $O(h)$.可见,时间复杂度与 n 和 m 都无关,能够满足动态服务选择在服务执行阶段对处理时间的苛刻要求.

算法 2. 动态选择算法(DS).

输入:CSQ_j;

输出:s^* or ∅.

```
Ω_j=null;                                  /*初始化*/
For each s_ij in CSQ_j
{
    get s_ij's current workload d_ij;      /*获取服务 s_ij 的当前负载等级 d_ij*/
    if s_ij is not overloaded              /*判断服务 s_ij 是否发生过载,即并发任务数量是否超过 l_ij^max */
        if Q^(d_ij)(s_ij) ⩽ LQ_j  Ω_j←s_ij   /*当服务 s_ij 未过载时,如果服务在当前负载等级下可以满足局
                                           部 QoS 约束,则加入集合Ω_j*/
}
If Ω_j=null Return ∅;                      /*选择失败,没有合适的服务*/
    Else  s* = arg max_{s_ij∈Ω_j} u_ij^(d_ij) ;   /*获取集合中具有最大效用值的服务 s_ij */
Return s*.
```

5 实验分析

5.1 实验环境配置

本文通过仿真实验来模拟服务选择环境并验证了本模型的有效性.实验环境为一台 HP Intel P4 CPU/4G RAM/Windows 2003/C#.采用混合整数规划 MIP(mixed integer programming)工具 Lpsolver version 5.5 实现线性规划求解;采用 C#实现本文所提出的选择方法.

仿真实验采用 3 个 QoS 属性(响应时间、可靠性和可用性)和 3 个负载等级,并分别使用两种不同的数据:

一是采用随机生成的服务 QoS 数据(响应时间在 1ms~800ms 之间,可靠性和可用性在 0.5~1.0 之间)作为生成 QoS 模型的参考数据;

二是采用 QWS 数据集[23]的响应时间、可靠性和可用性数据作为生成 QoS 模型的参考数据.

生成基于负载等级 QoS 模型的具体方法为:分别以随机数据集和 QWS 数据集中的数据为负载等级 2 时的 QoS 值,由于各服务在不同负载等级下表现出的 QoS 特性各不相同,故对于响应时间,其负载等级 1 和负载等级 3 时的值分别为负载等级 2 时的值乘以随机因子 $\alpha_1(0.8<\alpha_1<1)$和 $\alpha_2(1<\alpha_2<1.2)$;而对于可靠性和可用性,其负载等级 1 和负载等级 3 时的值分别为负载等级 2 时的值乘以随机因子 $\beta_1(1<\beta_1<1.1)$和 $\beta_2(0.9<\beta_2<1)$.并且,当所计算出的可靠性或可用性的值大于等于 1 时,则对应的属性值仍采用原来数据集中的值.如此,可以保证各 QoS 属性值满足不同负载等级下的依赖关系.

同时,在随机数据集中随机生成各服务的最大负载 $l_{ij}^{\max}$;在 QWS 数据集中采用最大吞吐量数据作为服务的最大负载 $l_{ij}^{\max}$,并根据设定的负载等级数量平均划分为 3 个负载区间.

表 2 显示了不同数据集下的实验环境配置.

Table 2 Experimental parameters in different data sets

表 2 不同数据集下的主要实验环境参数

	随机数据集	QWS 数据集	备注
用户请求速率(λ, reqs/s)	40~200	40~200	每个 VCSF 的λ均相等
每个服务类的实体服务数(m)	500~2500	100~1000	
QoS 维数(r)	3	3	各维 QoS 的权重相同
候选队列最大长度(h)	2~10	2~20	
负载等级数量(z)	3	3	各负载等级出现概率相同
端到端的 QoS 约束($Q^{\max}$)	随机生成	随机生成	至少能生成一个满足约束的组合服务
任务(或服务类)数量(n)	10~80	10~50	

实验中的服务类数量为 $n(n\geqslant 10)$,并假设每个服务类节点都拥有相同数量的实体服务.组合服务一般包含顺序、并发、选择和循环这 4 种基本结构,而选择和循环结构又可以转化为顺序和并发结构[13],故本实验的服务流程只考虑这两种结构.实验采用图 5 所示的由 n 个服务类构成的两种虚拟组合服务流程 VCSF:一个为顺序结构;另一个为顺序和并发的混合结构.

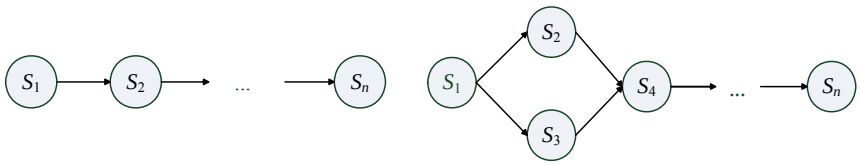


Fig.5 Two virtual composite service flows adopted in the experiment

图 5 实验采用的两个虚拟组合服务流程

5.2 实验结果与分析

在前面的相关工作中,本文已经分析了文献[6,17,18]等针对多个用户需求的服务选择与组合的研究情况,指出了这些研究都基于这样一个假设,即所有用户需求信息可以事先获得.而这一假设并不适用于开放的、动态的服务选择与组合环境.本文研究主要针对的是用户需求信息无法事先获得、用户请求离散随机到达的情况,故本文模型需要实时地响应用户的服务组合请求.通过前面与相关研究的比较,已经定性地分析了本文模型 LADSSM 的优势,为了进一步验证 LADSSM 的有效性,本实验将该模型与经典的、具有代表性的全局优化服务选择方法[1](以下用 Global 表示)以及近年提出的混合优化法[8](以下用 Hybrid 表示)进行定量的实验比较.为了方便分析,本文提出 3 个评价指标:执行成功率(ESR)、平均执行效用(AU)和计算时间(CT).

定义 9(执行成功率(ESR)). 指成功响应的用户请求数量占用户请求总数量的比例,表示为

$$\mathrm{ESR}=num_s/num \tag{18}$$

其中,num_s 为成功响应的用户请求数量,num 为用户请求总数量.请求被成功响应的判定标准有两个:一个是组合服务要满足端到端的 QoS 约束;另一个是各服务节点不发生过载.该指标反映了服务选择方法在多用户的动态负载环境下的应变能力.

定义 10(平均执行效用(AU)). 指对成功响应的用户请求而言,所有组合服务执行效用 U(见定义 7)的均值.该指标反映了本文模型的优化能力.

定义 11(计算时间(CT)). 指算法生成组合服务所需的平均计算时间.该指标反映了算法的效率以及适应大规模环境的能力.值得一提的是,由于本模型动态选择算法的运行时间极短,可以忽略,故本实验将候选服务队列生成算法(CSQGen)的计算时间和全局优化方法 Global 的计算时间进行了比较.

5.2.1 执行成功率

第 1 组实验分别采用随机数据集和 QWS 数据集,测试了本文模型 LADSSM 与 Global 方法和 Hybrid 方法在不同用户请求速率λ下执行成功率 ESR 的变化.

从图 6 中可以看出:随着λ从 10reqs/s 增加到 100reqs/s 时(即组合服务的负载增加),执行成功率逐步下降.其中,Global 方法的执行成功率下降得较快;而本文方法的执行成功率下降得相对较慢,特别是 h 较大时,执行成功率下降得更慢.如:当λ=90reqs/s,使用随机数据集的 Global 和 Hybrid 方法的 ESR 为 20%左右,而本文模型的 ESR 分别约为 60%(h=2)和 90%(h=5);使用 QWS 数据集时也有类似结果.可见,无论是随机数据集还是 QWS 数据集的实验结果都表明,本文模型的执行成功率 ESR 要极大地高于 Global 方法和 Hybrid 方法.

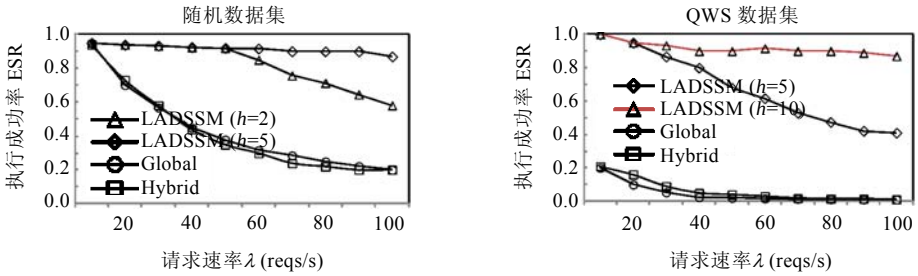


Fig.6 Variation of execution success rate with request rate λ

图 6 执行成功率随请求速率λ的变化

第 2 组实验分别采用随机数据集和 QWS 数据集,测试了本文模型在不同 h 下执行成功率的变化.

从图 7 中可以看出,执行成功率随 h 的增加而增长.在 h 相同的情况下,当λ较低(即组合服务的负载较小)时,执行成功率 ESR 相对较大;反之,ESR 相对较小.在随机数据集中,当 h 增加到 6 时,执行成功率 ESR 就已达到了 85%以上;而在 QWS 数据集中,当 h 增加到 12 时,执行成功率 ESR 达到了 85%以上.

通过比较可以看出:本文方法可以有效地提高动态环境下服务的执行成功率;当组合服务的负载较大时,可以通过适当提高 h 来维持较高的执行成功率.

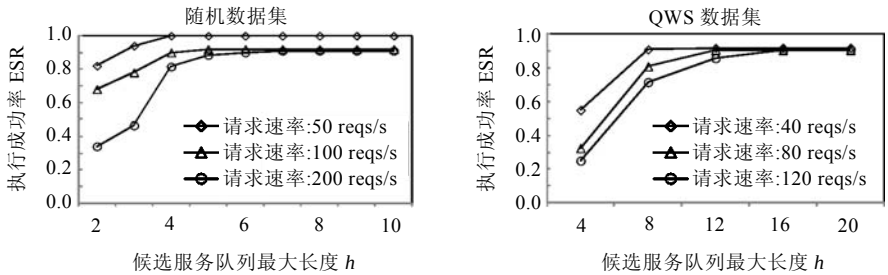


Fig.7 Variation of execution success rate with the maximum length of candidate queue h

图 7 执行成功率随候选服务队列最大长度 h 的变化

5.2.2 平均执行效用

第 3 组实验分别采用随机数据集和 QWS 数据集,测试本文模型(LADSSM)的平均执行效用 AU 随候选实体服务数量 m 和服务类数量 n 的变化.

从图 8 和图 9 可以看出:本文模型在不同的服务规模(m 和 n)下都可以取得一个较好的平均执行效用 AU;在相同的用户请求速率(λ,单位:reqs/s)下,本文模型所获得的平均执行效用要高于采用 Global 方法和 Hybrid 方法所获得的平均执行效用.显然,LADSSM 考虑了服务的负载以及在不同负载下的 QoS,因而提高了组合服务运行时的执行效用.此外,当用户请求速率λ较大时,平均执行效用会有所降低.这说明,当组合服务的负载较大时,本模型会在保证用户端到端的 QoS 需求不被违背的前提下,选择效用次优的服务,以防止优质服务的过载或性能降级,从而实现了负载感知的自适应服务优化选择.

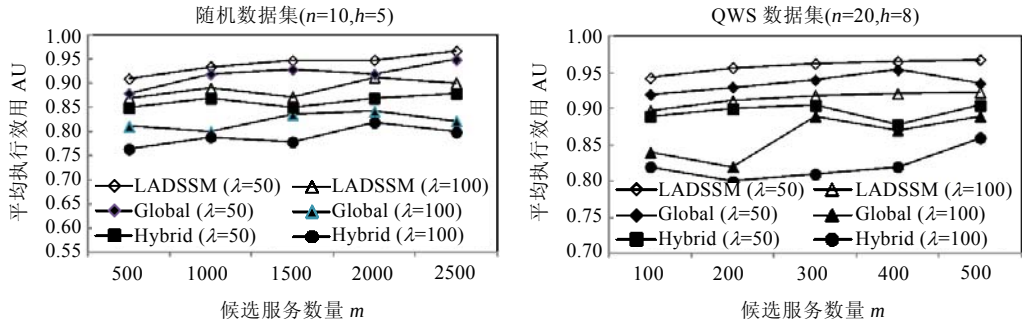


Fig.8 Variation of AU with the number of candidate services m

图 8 平均执行效用 AU 随候选服务数量 m 的变化

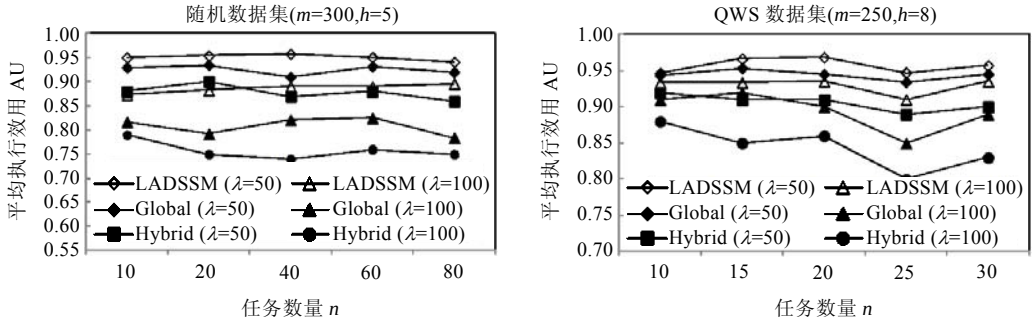


Fig.9 Variation of AU with the number of tasks n

图 9 平均执行效用 AU 随任务数量 n 的变化

5.2.3 计算时间

第 4 组实验测试了本文算法(CSQGen)与全局优化方法(Global)和混合优化方法(Hybrid)的计算时间.

从图 10 和图 11 可以看出,全局优化方法(Global)的计算时间随 m 和 n 增加而增长较快.混合优化方法(Hybrid)的计算时间随 m 和 n 增加而增长相对较慢;本文算法的计算时间随 m 和 n 增加也增长相对较慢.从总体上讲,本文算法在计算开销上优于上述两种算法,因而可以有效应对较大规模的服务场景.由此可见,本文算法具有较低的计算开销.此外,从结果还可以看出,h 对计算时间有影响:当 h 增长时,计算时间相对较大.所以,为维持较短的计算时间和较高的执行成功率,需要选择一个合适的 h 值.可见,本模型能够通过调节 h 来适应不同的组合服务环境以及服务组合性能要求.

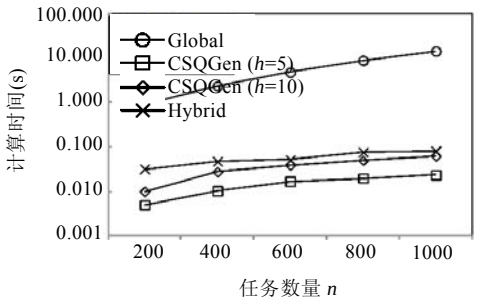


Fig.10 Variation of time cost with m

图 10 计算时间随候选服务数量 m 的变化

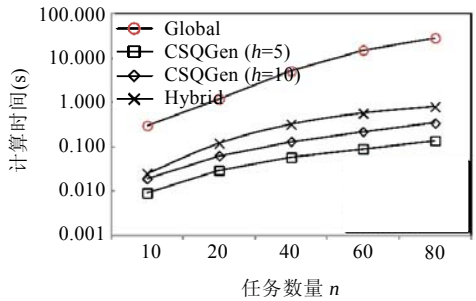


Fig.11 Variation of time cost with n

图 11 计算时间随任务数量 n 的变化

6 结束语

本文提出了一种面向多用户的负载感知的动态服务选择模型 LADSSM,该模型能够在多用户、多业务的环境下较好地适应服务负载的变化,及时、动态地生成组合服务.仿真实验测试和比较了本文模型的执行成功率、平均执行效用和计算时间.结果表明:本文模型能够较好地感知服务负载变化,并及时有效地提供效用优化的服务选择方案.与以往相关研究对比,本文模型具有 3 个特点:① 构成组合服务的各实体服务是在执行过程中被动态选择并绑定执行的;② 支持开放的多用户、多业务环境,服务选择无需事先获得所有用户请求信息;③ 在满足不同用户 QoS 约束的前提下,感知服务节点的负载,实现适应负载变化的组合服务动态优化选择.

在实际应用中,当用户请求数量较大时,对多个服务实体的"负载等级"等数据的获取会造成一定的开销.因此,当用户请求数量较大时,需要减少候选服务队列最大长度 h(即将 h 设为较小的值),如此可以减少运行时备选服务的数量,从而减少获取服务实体相关动态数据的开销.而如何有效地设定 h,则需要根据实际运行环境在测量实验的基础上通过权衡效用、成功率和开销来实现.本文研究所适用的场景主要是用户实时请求离散、随机到达的情形,因而是一种在线式的动态服务选择方法;而对于大量成批连续到达的用户请求,则适用于离线式的服务选择方法,这是本文下一步的研究方向.

References:

[1] Zeng LZ, Benatallah B, Ngu AHH, Dumas M, Kalagnanam J, Chang H. QoS-Aware middleware for Web services composition. IEEE Trans. on Software Engineering, 2004,30(5):311−327. [doi: 10.1109/TSE.2004.11]

[2] Canfora G, Penta MD, Esposito R, Villani ML. QoS-Aware replanning of composite Web services. In: Proc. of the IEEE Int'l Conf. on Web Services (ICWS 2005). 2005. 121−129. [doi: 10.1109/ICWS.2005.96]

[3] Meulenhoff PJ, Ostendorf DR, Živković M, Meeuwissen HB, Gijsen BMM. Intelligent overload control for composite Web services. In: Baresi L, Chi CH, Suzuki J, eds. Proc. of the ICSOC-ServiceWave 2009. LNCS 5900, 2009. 34−49. [doi: 10.1007/78-3-642-10383-4_3]

[4] Schmid HA. Service congestion: The problem, and an optimized service composition architecture as a solution. In: Proc. of the IEEE Int'l Conf. on Web Services (ICWS 2006). 2006. 505−514. [doi: 10.1109/ICWS.2006.120]

[5] Alrifai M, Skoutas D, Risse T. Selecting skyline services for QoS-based Web service composition. In: Proc. of the 19th Int'l Conf. on World Wide Web (WWW 2010). 2010. 11−20. [doi: 10.1145/1772690.1772693]

[6] Wang XZ, Wang ZJ, Xu XF, Liu Y. A service composition method for tradeoff between satisfactions of multiple requirements. Journal of Computer Research and Development, 2011,48(4):627−637 (in Chinese with English abstract).

[7] Yu T, Zhang Y, Lin KJ. Efficient algorithms for Web services selection with end-to-end QoS constraints. ACM Trans. on WEB (TWEB), 2007,1(1):1−26. [doi: 10.1145/1232722.1232728]

[8] Alrifai M, Risse T. Combining global optimization with local selection for efficient QoS-aware service composition. In: Proc. of the 18th Int'l Conf. on World Wide Web (WWW 2009). 2009. 881−890. [doi: 10.1145/1526709.1526828]

[9] Wu J, Chen L, Deng SG, Li Y, Kuang L. QoS-Skyline based dynamic service selection. Chinese Journal of Computers, 2010,33(11):2136−2146 (in Chinese with English abstract). [doi: 10.3724/SP.J.1016.2010.02136]

[10] Huang AFM, Lan CW, Yang SJH. An optimal QoS-based Web service selection scheme. Information Sciences, 2009,179(19):3309−3322. [doi: 10.1016/j.ins.2009.05.018]

[11] Wang SG, Sun QB, Yang FC. Web service dynamic selection by the decomposition of global QoS constraints. Ruan Jian Xue Bao/Journal of Software, 2011,22(7):1426−1439 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/3842.htm [doi: 10.3724%2fSP.J.1001.2011.03842]

[12] Liu SL, Liu YX, Zhang F, Tang GF, Jing N. A dynamic Web services selection algorithm with QoS global optimal in Web services composition. Ruan Jian Xue Bao/Journal of Software, 2007,18(3):646−656 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/18/646.htm [doi: 10.1360/jos180646]

[13] Ardagna D, Pernici B. Adaptive service composition in flexible processes. IEEE Trans. on Software Engineering, 2007,33(6):369−384. [doi: 10.1109/TSE.2007.1011]

[14] Qi LY, Tang Y, Dou WC, Chen JJ. Combining local optimization and enumeration for QoS-aware Web service composition. In: Proc. of the IEEE Int'l Conf. on Web Services (ICWS 2010). Miami, 2010. 34−41. [doi: 10.1109/ICWS.2010.62]

[15] Wang Y, Dai GP, Hou YR. Dynamic methods of trust-aware composite service selection. Chinese Journal of Computers, 2009, 32(8):1668−1675 (in Chinese with English abstract). [doi: 10.3724%2fSP.J.1016.2009.01668]

[16] Ardagna D, Mirandola R. Per-Flow optimal service selection for Web services based processes. The Journal of Systems and Software, 2010,83:1512−1523. [doi: 10.1016/j.jss.2010.03.045]

[17] Kang GS, Liu JX, Tang MD, Liu XQ, Fletcher KK. Web service selection for resolving conflicting service requests. In: Proc. of the IEEE Int'l Conf. on Web Services (ICWS 2011). 2011. 387−394. [doi: 10.1109/ICWS.2011.37]

[18] Li HF, Zhu Q, Ouyang YQ. Non-Cooperative game based QoS-aware Web services composition approach for concurrent tasks. In: Proc. of the IEEE Int'l Conf. on Web Services (ICWS 2011). 2011. 444−451. [doi: 10.1109/ICWS.2011.45]

[19] Boone B, Van Hoecke S, Van Seghbroeck G, Joncheere N, Jonckers V, De Turck F, Develder C, Dhoedt B. SALSA: QoS-aware load balancing for autonomous service brokering. The Journal of Systems and Software, 2010,83(3):446−456. [doi: 10.1016/j.jss.2009.09.033]

[20] Li WZ, Guo S, Xu P, Lu SL, Chen DX. An adaptive load balancing algorithm for service composition. Ruan Jian Xue Bao/Journal of Software, 2006,17(5):1068−1077 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/17/1068.htm [doi: 10.1360aosl71068]

[21] Jaeger MC, Rojec-Goldmann G, Mhl G. QoS aggregation for Web service composition using workflow patterns. In: Proc. of the 8th IEEE Int'l Enterprise Distributed Object Computing Conf. Monterey, 2004. 149−159. [doi: 10.1109/EDOC.2004.1342512]

[22] Korkmaz T, Krunz M. Multi-Constrained optimal path selection. In: Proc. of the 20th Annual Joint Conf. of the IEEE Computer and Communications Societies (INFOCOM 2001). 2001. 834−843. [doi: 10.1109/INFCOM.2001.916274]

[23] Al-Masri E, Mahmoud QH. Investigating Web services on the World Wide Web. In: Proc. of the 17th Int'l Conf. on World Wide Web (WWW 2008). 2008. 795−804. [doi: 10.1145/1367497.1367605]

附中文参考文献:

[6] 王显志,王忠杰,徐晓飞,刘英.面向多需求可满足性折中的服务组合方法.计算机研究与发展,2011,48(4):627−637.

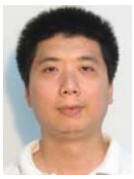
[9] 吴健,陈亮,邓水光,李莹,邝砾.基于 skyline 的 QoS 感知的动态服务选择.计算机学报,2010,33(11):2136−2146. [doi: 10.3724/SP.J.1016.2010.02136]

[11] 王尚广,孙其博,杨放春.基于全局 QoS 约束分解的 Web 服务动态选择.软件学报,2011,22(7):1426−1439. http://www.jos.org.cn/1000-9825/3842.htm [doi: 10.3724%2fSP.J.1001.2011.03842]

[12] 刘书雷,刘云翔,张帆,唐桂芬,景宁.一种服务聚合中 QoS 全局最优服务动态选择算法.软件学报,2007,18(3):646−656. http://www.jos.org.cn/1000-9825/18/646.htm [doi: 10.1360/jos180646]

[15] 王勇,代桂平,侯亚荣.信任感知的组合服务动态选择方法.计算机学报,2009,32(8):1668−1675. [doi: 10.3724%2fSP.J.1016.2009.01668]

[20] 李文中,郭胜,许平,陆桑璐,陈道蓄.服务组合中一种自适应的负载均衡算法.软件学报,2006,17(5):1068−1077. http://www.jos.org.cn/1000-9825/17/1068.htm [doi: 10.1360aosl71068]



朱勇(1977－),男,江苏南京人,博士生,主要研究领域为服务计算.
E-mail: zhuyong@seu.edu.cn

李伟(1978－),男,博士,副教授,CCF 高级会员,主要研究领域为下一代网络体系结构,服务计算.
E-mail: xchlw@seu.edu.cn

罗军舟(1960－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为下一代网络体系结构,云计算,服务计算.
E-mail: jluo@seu.edu.cn