ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.20, No.3, March 2009, pp.546−556
doi: 10.3724/SP.J.1001.2009.03339

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

随机 QoS 感知的可靠 Web 服务组合*

范小芹[1,2,3+], 蒋昌俊[1,2], 王俊丽[1], 庞善臣[1,4]

[1](同济大学 电子与信息工程学院,上海 201804)

[2](同济大学 嵌入式系统与服务计算教育部重点实验室,上海 201804)

[3](山西大学 计算机与信息技术学院,山西 太原 030006)

[4](山东科技大学 信息科学与工程学院,山东 青岛 266510)

Random-QoS-Aware Reliable Web Service Composition

FAN Xiao-Qin[1,2,3+], JIANG Chang-Jun[1,2], WANG Jun-Li[1], PANG Shan-Chen[1,4]

[1](Electronics and Information Engineering School, Tongji University, Shanghai 201804, China)

[2](Key Laboratory of Embedded System and Service Computing, Ministry of Education, Tongji University, Shanghai 200092, China)

[3](Computer and Information Technology School, Shanxi University, Taiyuan 030006, China)

[4](College of Information Science and Engineer, Shandong University of Science and Technology, Qingdao 266510, China)

+ Corresponding author: E-mail: fxq0917@hotmail.com

Fan XQ, Jiang CJ, Wang JL, Pang SC. Random-QoS-Aware reliable Web service composition. *Journal of Software*, 2009,20(3):546−556. http://www.jos.org.cn/1000-9825/3339.htm

Abstract: In the service-oriented environment, a single Web service can hardly satisfy the given request, so the composition of multiple Web services is required to fulfill the goal. Without considering the inherent stochastic and dynamic nature of Web service, the existing composition methods mostly generate static plans. As a result, Web service composition often terminates with failure inevitably. In this paper, metrical methods of several random QoS dimensions and QoS Management Architecture are presented, and one reliable Web service composition algorithm is also designed based on markov decision process (MDP)—only dynamic controlling method of stochastic discrete event system (SDES). Experimental results demonstrate the success rate of Web service composition has been improved greatly.

Key words: Web service composition; Markov decision process; random QoS; Web service; reliable composition

摘 要: 在面向服务的环境下,单个 Web 服务往往不能满足用户的要求,这时就需将已有的单个 Web 服务进行组合,以便产生满足用户需求的、增值的组合服务.已有的服务组合方法都很少考虑 Web 服务的随机性和 Internet 环境的动态性,从而在服务选择过程中产生的规划都是静态规划,结果导致在服务组合时都以较大概率出现组合失败.针对上述问题,提出了 Web 服务各随机 QoS 指标的度量方法和自适应 QoS 管理体系结构,并利用随机型离散事件系统唯一的动态控制方法——马尔可夫决策过程(MDP),设计出随机 QoS 感知的可靠 Web 服务组合算法.实验结果

* Supported by the National Natural Science Foundation of China under Grant No.90718012 (国家自然科学基金); the National High-Tech Research and Development Plan of China under Grant No.2007AA01Z136 (国家高技术研究发展计划(863)); the National Basic Research Program of China under Grant No.2003CB316902 (国家重点基础研究发展计划(973))

Received 2008-01-16; Accepted 2008-03-31

表明,考虑随机性的 QoS 度量方法和 QoS 管理体系结构,以及平衡了“风险”与“报酬”的 MDP 有效地提高了服务组合成功率.

关键词: Web 服务组合;马尔可夫决策过程(MDP);QoS 随机性;Web 服务;可靠组合

中图法分类号: TP301 **文献标识码**: A

随着Web服务技术的发展,通过服务组合来满足用户需求已经成为必然趋势.但由于Internet环境的开放性和动态性以及 Web 服务的随机性,导致组件服务的 QoS 具有很强的不确定性,从而影响了服务组合成功率和组合服务的质量.所以,如何准确地度量Web服务的QoS以及动态地对其进行自适应管理,对服务的成功组合有着重要的意义;另一方面,已有的服务组合方法在规划过程中都假设 Web 服务具有确定行为,而这个假设往往导致服务组合过程中出现约束冲突.所以,在考虑真实环境的不确定性和动态性的情况下,研究服务的可靠组合方法有着重要的现实意义.

现有的 QoS 度量方法基本都是用历史日志的期望近似实际的 QoS 值[1],这明显忽略了很多不确定的因素,结果不可避免地导致估计值与真实值之间的偏差较大,从而在服务组合时都以较大概率出现组合失败.针对上述问题,本文将 Web 服务各 QoS 指标定义为一系列随机变量,并利用随机变量的数学期望和方差来度量各指标的取值,这样就克服了以往仅用历史日志的算术平均或比值来衡量指标的弊端,将具有不同波动性的 Web 服务区分开来;同时,考虑到在节假日或使用高峰期间,爆炸式的请求可能会阻塞 Web 服务主机,进而影响 Web 服务的性能,本文提出一项新的 QoS 指标——负载,用来描述 Web 服务的实时利用率,并作为服务选择的一项指标.

最初的 Web 服务体系结构模型仅由 UDDI、服务请求者和服务提供商组成,后来,为了便于 Web 服务选择,有研究者对其进行了扩充,引入了 QoS 指标描述部分,但是,各 QoS 指标值是作为确定值发布的.实际上,由于 Web 服务的随机性,各 QoS 指标值都在随机变化.因此,用户从 UDDI 得到的查询值与实际值相差较大,结果降低了服务组合成功率.为了解决上述问题,有研究者对 Web 服务的 QoS 值进行动态更新,方法是只考虑最近 N 次的日志,然后根据这 N 次的结果来更新 Web 服务 QoS,这种更新办法导致系统计算量过大,管理代价过高.因此,本文扩展了 Web 服务体系结构,提出了一种新的 Web 服务 QoS 管理体系结构,该体系结构可以根据用户的反馈和系统的检测信息,对 UDDI 中的 QoS 值进行自适应更新,从而缩小查询值与实际值的差距,有助于 Web 服务的成功组合.

服务组合是指对现有的一组服务按照一定的业务逻辑进行集成,从而更好地满足用户的需求.服务组合广义上可以分为手工组合、半自动组合和自动组合[2].由于 Internet 上有大量具有不确定性的 Web 服务可用,手工分析这些服务并生成合成服务规划已经不能满足实际的应用需求.对于自动组合,首先需要利用匹配算法生成状态图或信息图,然后利用回溯算法确定具体的组合规划[3−6].这种组合方法不仅过程复杂,而且在组合执行过程中,如果存在一个 Web 服务调用异常(例如:调用失败,或与约束冲突),则整个规划失败.因此,很多关于服务组合的应用和研究工作都侧重于半自动方式[7,8].半自动组合方式的实现,首先需要业务人员建立组合流程模型,然后对模型中各抽象任务自动地绑定调用实例.基于全局优化的服务绑定方法主要有两类:以整数规划为代表的穷尽优化方法[1]和以进化算法为代表的近似优化方法[9].但这两种方法存在一个共同的缺陷:产生的规划都是静态规划,从而在组合过程中如果出现调用异常,则只能进行重规划[1],或者为了提高组合效率而选择一个次优规划,或者采取协商机制[10],这些方法将不可避免地降低组合服务的性能.实际上,Internet 环境下预定义流程的Web服务选择问题,是对一个随机型离散事件系统进行动态寻找最优规划的过程,而MDP是这类系统唯一的动态控制方法[11],同时已经证明其复杂度是 P 完全的[12].因此,本文利用 MDP 进行服务组合,该方法在弱化其他规划中许多假设的同时又可以形式化决策过程中的一些不确定性情况(例如:服务调用失败),最终产生一个平衡了“风险”和“报酬”的动态最优规划[12].文献[13]虽然也利用 MDP 进行服务组合,但主要考虑组合服务的流程结构特点,给出不同结构下的 MDP 模型,对系统各状态下采取的决策次数并没有进行讨论,而这对服务组合成功率有重要的影响.

本文第 1 节给出 Web 服务的 QoS 模型以及各随机 QoS 指标的度量方法.第 2 节提出一种具有动态自适应

功能的 Web 服务 QoS 管理体系结构.第 3 节基于 MDP 设计随机 QoS 感知的可靠 Web 服务组合算法.第 4 节利用仿真实验说明该方法的有效性.第 5 节对本文进行总结.

1 Web 服务的 QoS 模型及各随机 QoS 指标度量方法

在服务组合过程中,查询代理(如图 1 所示)对一个请求反馈回的结果是一类功能属性相同、非功能属性有差异的 Web 服务.为了对这类服务进行区分,我们借助 Web 服务的质量指标建立了一个 Web 服务 QoS 模型.为了方便讨论,本文的 QoS 模型只包含了有限的几项指标,其实这个模型是可扩展的,当有新指标加进来时,不需要对建立在该模型上的服务选择技术进行修改.

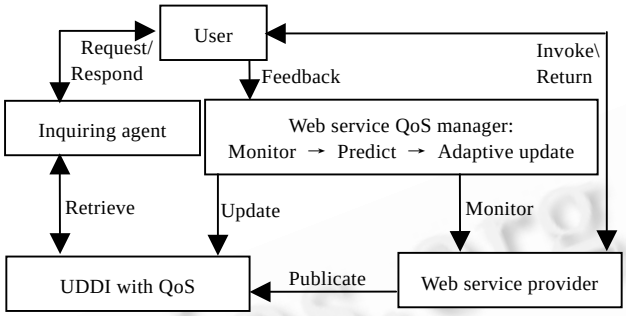


Fig.1 Web service architecture with QoS manager

图 1 Web 服务 QoS 管理体系结构

下面给出组件 Web 服务的各指标的定义及其度量方法.

• 执行代价:Web 服务 *ws* 的执行代价 $C(ws)$是指调用服务 *ws* 所必须支付的费用.它由 Web 服务提供商来提供,并且在本文中 $C(ws)$是一个确定的值,不因环境的动态性而改变.

• 执行时间:任务 t_i 在 Web 服务 *ws* 上的执行时间 $T(ws)$是指:从服务 *ws* 收到任务 t_i 的执行请求(例如:服务执行所需参数)到输出执行结果的延时.

对 $T(ws)$,其取值结果不仅取决于 *ws* 本身对任务的处理速率,同时也与调用 *ws* 的任务数(例如:调用任务数越多,则 $T(ws)$越大)以及 *ws* 对所有任务的处理规则(例如:先来先服务,随机服务,优先强占服务等)有关.因此,$T(ws)$是一个与上述因素有关的随机变量.本文采用具有 k 个服务台的批服务 $M/M/k$ 排队模型来对 $T(ws)$进行预测.对 *ws*:假设所有调用任务按参数为λ的泊松分布到达,*ws* 对任务的处理时间服从参数为μ的指数分布,并且对所有调用任务采取先来先服务(FCFS)的排队规则,并约定 *ws* 同时最多给 k 个任务提供服务,则 $T(ws)$的数学期望和方差分别为

$$E(T(ws))=\frac{\rho\pi_k}{(1-\rho)^2\lambda}+\frac{1}{\mu},$$

$$D(T(ws))=\frac{\pi_k[2(1-\rho)-\pi_k]\rho^2}{\lambda^2(1-\rho)^4}+\frac{1}{\mu^2},\text{其中 }\rho=\frac{\lambda}{k\mu},\ \pi_k=\frac{(k\rho)^k\pi_0}{k!},\text{且 }\pi_0=\left[\sum_{i=0}^{k-1}\frac{(k\rho)^i}{i!}+\frac{(k\rho)^k}{k!(1-\rho)}\right]^{-1}.$$

• 负载:Web 服务 *ws* 的负载 $L(ws)$用来衡量该服务的实时利用率.其取值取决于调用 *ws* 的任务的到达率λ(单位时间内任务到达个数)以及 *ws* 的服务率μ(单位时间内处理的任务数),并且有 $L(ws)=\lambda/\mu$.

$L(ws)\to 0$ 表明该时刻 *ws* 的利用率较低,其服务质量能以较大概率得到保证;$L(ws)\to 1$ 表明 *ws* 的利用率已经达到极限,一旦λ发生变化,则其性能以较大概率得不到保证;$L(ws)\to 1$ 表明 *ws* 已经超负荷运行,调用 *ws* 的任务的执行时间可能无限延长,其余 QoS 也很难得到保证.

• 可靠性:Web 服务 *ws* 的可靠性 $A(ws)$反映其可靠程度.$A(ws)$取值越大,表明 *ws* 出现失效的频率越低,也即平均失效时间越短.在 Web 环境下,导致 *ws* 失效的因素很多,例如:Web 服务固有的 bugs、硬件环境以及网络环

境等.因此,准确地说,$A(ws)$是一个随机变量.但由于 $A(ws)$的分布未知,所以采用样本的性质来对总体进行估计是一种比较有效的办法,而且矩估计所得一阶、二阶估计值都是总体数学期望和方差的无偏估计.所以,对 ws 的最近 n 次可靠性执行日志:$a_1,a_2,\ldots,a_n(a_i(1\le i\le n)$可由图 1 中的 QoS 信息管理器监测得到),其中,a_i 为 ws 的第 i 次故障与第 i+1 次故障之间的时间间隔,则 $A(ws)$的数学期望和方差分别为

$$E(A(ws))=\sum_{i=1}^{n}a_i\Big/n,$$

$$D(A(ws))=\frac{1}{n}\times\sum_{i=1}^{n}(a_i-E(A(ws)))^2.$$

• 信誉度:Web 服务 ws 的信誉度 $R(ws)$反映其可信程度.$R(ws)$取值越大,表明 ws 的可信程度越高,即 ws 的实际执行结果与用户的期望越接近.由于 Web 环境的开放性和动态性,以及不同用户对 ws 的同一执行结果的满意程度也可能不同,从而导致 ws 的信誉度 $R(ws)$在随机变化.因此,通过取样对 $R(ws)$的数学期望和方差做出估计.为了准确而公平地估计出 ws 的信誉度 $R(ws)$,抽样不仅要考虑用户的反馈,同时也要结合 QoS 信息管理器的监测数据.根据不同 QoS 指标的取值对服务性能的影响效果不同,分为正指标和负指标,其中正指标是指该指标取值越大,则 ws 的性能越好,例如可靠性和信誉;负指标是指该指标取值越大,表明 ws 的性能越差,例如执行时间、执行代价和负载,则对 ws 的近 n 次执行和查询日志:

$$\begin{Bmatrix}a_1^{j(+)/j(-)} & a_2^{j(+)/j(-)} & \ldots & a_n^{j(+)/j(-)}\\ a_1^{j(query)} & a_2^{j(query)} & \ldots & a_n^{j(query)}\end{Bmatrix},$$

其中 $a_i^{j(+)}/a_i^{j(-)}(1\le i\le n)$ 表示第 i 次执行 ws 所得的关于正/负指标 j 的结果,$a_i^{j(query)}(1\le i\le n)$ 表示第 i 次调用 ws 前关于指标 j 的查询值.在本文中,j 从执行代价、执行时间、可靠性以及负载中取值.记

$$l_i^{j(-)}=\begin{cases}1, & a_i^{j(-)}\le a_i^{j(query)}\\ \varepsilon^{j(-)}\Big/\left(a_i^{j(-)}-a_i^{j(query)}\right), & a_i^{j(-)}>a_i^{j(query)}\end{cases},$$

其中 $\varepsilon^{j(-)}$为用户确定的、对负指标 j 的、可接受的偏差,且 $\varepsilon^{j(-)}\le a_i^{j(-)}-a_i^{j(query)}$;类似地,记

$$l_i^{j(+)}=\begin{cases}1, & a_i^{j(query)}\le a_i^{j(+)}\\ \varepsilon^{j(+)}\Big/\left(a_i^{j(query)}-a_i^{j(+)}\right), & a_i^{j(query)}>a_i^{j(+)}\end{cases},$$

其中 $\varepsilon^{j(+)}$为用户确定的、对正指标 j 的、可接受的偏差,且 $\varepsilon^{j(+)}\le a_i^{j(query)}-a_i^{j(+)}$,则 $R(ws)$的数学期望和方差分别为

$$E(R(ws))=\frac{1}{m}\times\sum_{k=1}^{m}w^{j(-)}l_k^{j(-)}+\frac{1}{n}\times\sum_{k=1}^{n}w^{j(+)}l_k^{j(+)},$$

$$D(R(ws))=\frac{1}{m}\times\sum_{k=1}^{m}\left|a_k^{j(-)}-a_k^{j(query)}\right|+\frac{1}{n}\times\sum_{k=1}^{n}\left|a_k^{j(+)}-a_k^{j(query)}\right|,$$

其中 $w^{j(-)}/w^{j(+)}$ 是与负/正指标 j 对应的权值,反映出用户对不同指标的重视程度.

2 Web 服务 QoS 管理体系结构

近年来,Web 服务技术在 B2B 集成中已经发挥了重要的作用,然而 Web 服务的使用率仍然不高,部分原因在于:(1) 对提供者和请求者都无法提供一种公平、准确、透明的方式来计算和监测 Web 服务的 QoS;(2) 在服务发现过程中无法考虑 QoS 参数,从而使得请求者无法基于 QoS 参数来定义查询标准;也正是这些原因降低了服务组合成功率和组合服务的质量[14].为此,本文提出了一种 Web 服务 QoS 管理体系结构,如图 1 所示.

Web 服务 QoS 管理器首先对任务 t_i 和 Web 服务 ws 之间的通信数据进行监测,并记录与 ws 的 QoS 有关的信息,例如:单位时间内到达的任务数和处理完毕的任务数、故障发生时间和故障修复时间以及周期性监测到的一些实际执行信息等等;然后根据监测结果对 ws 的各 QoS 指标进行预测.令 X 为 ws 的某个 QoS 指标的实际执行结果,$E(X)$和 $D(X)$分别为调用 ws 前查询所得的该指标的数学期望和方差,则对于置信度α,Web 服务 QoS

管理器将根据预测结果对 ws 的相应的 QoS 指标值进行自适应更新.鉴于各指标对应的随机变量所服从的分布未知,则根据契比雪夫不等式,有:X 的置信度为 α 的置信区间为 $\{E(X)-\sqrt{D(X)/(1-\alpha)},E(X)+\sqrt{D(X)/(1-\alpha)}\}$,如果某次调用服务 ws 的执行结果 $x\notin\{E(X)-\sqrt{D(X)/(1-\alpha)},(E(X)+\sqrt{D(X)/(1-\alpha)})\}$,则根据最近 n 次的执行结果重新估计 $E(X)$和 $D(X)$,从而使得服务 ws 的 QoS 指标值随着各因素的变化能够动态地自适应更新.这种自适应机制与其他方法(例如:只有当 ws 的 QoS 与约束发生冲突,或每次都根据最新日志更新 QoS 指标时)相比,不仅大大减少了计算量,降低了管理代价,更重要的是使得查询所得服务的各 QoS 值与实际执行结果偏差减小,从而使得基于该体系结构的 Web 服务发现的准确率大为提高,有助于 Web 服务的成功组合.

3 基于 MDP 的 Web 服务组合

3.1 MDP的基本概念[15]

MDP 是动态规划与马尔可夫过程相结合的产物,适合于对随机型离散事件系统进行动态控制,且在对决策问题建模时考虑其随机性和有序性.本文采用离散决策时刻,有限阶段马尔可夫决策模型.

定义 1(马尔可夫决策模型). 五元组 $\{T,S,A(i),P(\bullet|i,a),r(i,a)\}$ 称作是马尔可夫决策模型,其中

T 是系统的决策时刻组成的集合,对有限阶段决策时刻记作 $T=\{0,1,2,\ldots,N\}$;

S 是系统所有可达状态组成的非空集合,即系统的状态空间;

$A(i)$是在状态 i 下所有可执行动作组成的集合;

$P(\bullet|i,a)$ 是转移概率函数.表示在当前状态 i 下,执行动作 $a\in A(i)$后所有可达状态的概率分布,且有

$$\sum_{j\in S}P(j|i,a)=1;$$

$r(i,a)$是报酬函数.表示在当前状态 i 下执行动作 $a\in A(i)$后所得报酬.一般来讲,报酬还依赖于下一个决策时刻的状态 j,且有

$$r(i,a)=\sum_{j\in S}r(i,a,j)\times P(j|i,a).$$

定义 2(马氏决策函数). f 是状态空间 S 上的决策函数,当且仅当

对$\forall_i\in S$ 有 $f(i)\in A(i)$,即 $f:S\to A(i)$.全体决策函数组成的集合记作 F.

定义 3(马氏策略). 决策函数序列 $\pi=(f_0,f_1,f_2,\ldots)$ 称为马氏策略,其中 $f_t\in F,t\in T$.

定义 4(最优方程). 对决策时刻 t 及状态 s_t,定义如下方程为最优方程,且对 $t=N$,有边界条件 $u_N(s_N)=r(s_N)$.

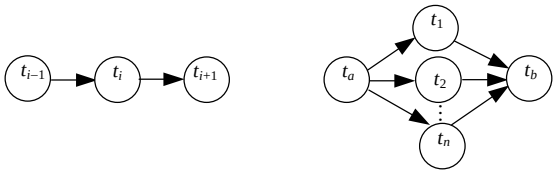
$$u_t(s_t)=\max_{a\in A(s_t)}\{r(s_t,a)+\sum_{j\in S}p_t(j|s_t,a)u_{t+1}(j)\}.$$

定义 5(最优策略). 使最优方程取得最大值的策略为最优策略.

3.2 Web 服务组合的 MDP 问题

Web 服务组合方法从组合方案生成方式可以分为两大类:静态组合和动态组合.静态组合意味着请求者应在组合计划实施前创建一个抽象的流程模型,该模型包括抽象任务以及任务间的依赖关系,且通常用图来描述.而动态组合不仅自动地选择、绑定 Web 服务,同时更重要的是自动地创建流程模型,这需要请求者指定一些约束关系,包括 Web 服务间的依赖关系、用户的偏爱等.本文是在预定义流程的基础上研究服务组合,因此属于静态组合.对于预定义的流程模型,一般只包含串行、并行、选择和循环 4 种结构.根据文献[10],可以将选择结构分解为不同的执行路径,将循环按分支概率展开,从而使得化简所得执行路径的流程图成为一个只包含串行和并行的 DAG 图.因此,本文研究的各执行路径的流程图中只考虑图 2 所示的两种结构.

决策时刻:由于组合服务流程图中含有有限个任务节点,因此,本文采用有限阶段 MDP,并且对图 2(a),在决策时刻 t、系统状态 s_t 下,执行动作 t_i 并成功后,系统状态将转移到 s_{t+1},同时处于决策时刻 t+1;如果 t_i 执行失败,则系统状态和决策时刻均不变.对图 2(b),认为并发的动作具有相同的决策时刻.



(a) Sequential structure (b) Parallel structure

(a) 顺序结构 (b) 并行结构

Fig.2 Composite service structures

图 2 组合服务结构

状态:对包含 n 个任务 $t_1,t_2,\ldots,t_n$ 的流程图,系统状态 s_t 用$(t_1,t_2,\ldots,t_n)$表示,其中 t_i 取 1 或 0,分别表示任务已绑定服务和未绑定服务.

动作:在决策时刻 t、系统状态 $s_t\in S$ 下,可执行的任务集为 $\{t_1,t_2,...t_n,n\ge 0\}\triangleq Execute(s_t)$,并且对 $t_i\in Execute(s_t)$,其可调用服务集为 $Invoke(t_i)$,则当前状态 s_t 下的动作集 $A(s_t)$为 $\bigcup_{i=1}^{n} Invoke(t_i)$,系统的动作集为 $\bigcup_{s_t\in S} A(s_t)$.

转移概率:在系统某一状态下,对于不同的可调用 Web 服务,转移概率的定义遵循:性能较好者分配较大的成功转移概率,而且对于执行失败的 Web 服务,其成功转移概率的定义遵循递减的原则.具体定义如下:在决策时刻 t,系统状态 s_t 及其 $Execute(s_t)$下,若:

(1) $|Execute(s_t)|=1$,不妨假设 $Execute(s_t)=\{t_i\}$且 $Invoke(t_i)=\{ws_1,ws_2,...,ws_n,n\ge 1\}$,则状态 s_t 下,第 k(其中 $k=2,3,\ldots$)次调用 ws_i 并成功执行转移到 s_{t+1} 的概率为

$$P^k(s_{t+1}/s_t,ws_i)=\frac{P^{k-1}(s_{t+1}/s_t,ws_i)}{k},\ P^k(s_t/s_t,ws_i)=1-P^k(s_{t+1}/s_t,ws_i),$$

且

$$P^1(s_{t+1}/s_t,ws_i)=\begin{cases}\dfrac{Quality(ws_i)}{\sum_{i=1}^{n}Quality(ws_i)}, & n\ge 2\\ Quality(ws_i), & n=1\end{cases},\ P^1(s_t/s_t,ws_i)=1-P^1(s_{t+1}/s_t,ws_i).$$

(2) $|Execute(s_t)|\ge 2$,不妨假设 $Execute(s_t)=\{t_1,t_2,...,t_n,n\ge 2\}$,且对 $t_i\in Execute(s_t)$有 $Invoke(t_i)=\{ws_{t_i}^1,ws_{t_i}^2,...,ws_{t_i}^{m_i},m_i\ge 1\}$,则状态 s_t 下,第 k(其中 $k=2,3,\ldots$)次调用 $ws_{t_1}^{j_1},ws_{t_2}^{j_2},...,ws_{t_n}^{j_n}$ 并成功执行转移到 s_{t+1} 的概率为

$$P^k(s_{t+1}/s_t,ws_{t_1}^{j_1},ws_{t_2}^{j_2},...,ws_{t_n}^{j_n})=P^{k-1}(s_{t+1}/s_t,ws_{t_1}^{j_1},ws_{t_2}^{j_2},...,ws_{t_n}^{j_n})\Big/k,$$

$$P^k(s_t/s_t,ws_{t_1}^{j_1},ws_{t_2}^{j_2},...,ws_{t_n}^{j_n})=1-P^k(s_{t+1}/s_t,ws_{t_1}^{j_1},ws_{t_2}^{j_2},...,ws_{t_n}^{j_n}),$$

且

$$P^1(s_{t+1}/s_t,ws_{t_1}^{j_1},ws_{t_2}^{j_2},...,ws_{t_n}^{j_n})=\prod_{i=1}^{n}P_i,\ P^1(s_t/s_t,ws_{t_1}^{j_1},ws_{t_2}^{j_2},...,ws_{t_n}^{j_n})=1-P^1(s_{t+1}/s_t,ws_{t_1}^{j_1},ws_{t_2}^{j_2},...,ws_{t_n}^{j_n}),$$

其中,

$$P_i=\begin{cases}Quality(ws_{t_i}^{j_i})\Big/\sum_{k=1}^{m_i}Quality(ws_{t_i}^{k}), & m_i\ge 2\\ Quality(ws_{t_i}^{j_i}), & m_i=1\end{cases},$$

且 $Quality(ws_i)$为 Web 服务 ws_i 的性能值,具体定义如下:

对任务 t_i 及其 $Invoke(t_i)$,一方面由于 Web 服务的正、负指标值增大对其性能影响不一致;另一方面,对于具有相同 QoS 指标期望值的不同 Web 服务,由于具有不同的方差,其性能可能相差较大.因此,首先将各 QoS 指标归一,且 ws_i 的负(正)指标 j 及方差的值 $Q^{j(-)}(ws_i)(Q^{j(+)}(ws_i))$归一后为 $\overline{Q^{j(-)}(ws_i)}$ ($\overline{Q^{j(+)}(ws_i)}$):

$$\overline{Q^{j(-)}(ws_i)}=\begin{cases}\dfrac{\max\{Q^{j(-)}(ws_i)\}-Q^{j(-)}(ws_i)}{\max\{Q^{j(-)}(ws_i)\}-\min\{Q^{j(-)}(ws_i)\}}, & \text{若}\max\{Q^{j(-)}(ws_i)\}\neq\min\{Q^{j(-)}(ws_i)\}\text{,其中,}1\le i\le n\\ 1, & \text{否则}\end{cases},$$

$$\overline{Q^{j(+)}(ws_i)}=\begin{cases}\dfrac{Q^{j(+)}(ws_i)-\min\{Q^{j(+)}(ws_i)\}}{\max\{Q^{j(+)}(ws_i)\}-\min\{Q^{j(+)}(ws_i)\}}, & \text{若}\max\{Q^{j(+)}(ws_i)\}\neq\min\{Q^{j(+)}(ws_i)\}\text{, 其中,}1\le i\le n\\ 1, & \text{否则}\end{cases},$$

则 $Quality(ws_i)=\sum_{j=1}^{7}w_j\times\overline{Q^{j(-)/(+)}(ws_i)}$,其中,$w_j\in[0,1]$且$\sum_{j=1}^{7}w_j=1$.

效益:在决策时刻 t,对状态 s_t 及其 $Execute(s_t)$,若:

(1) $|Execute(s_t)|=1$,不妨假设 $Execute(s_t)=\{t_i\}$ 且 $Invoke(t_i)=\{ws_1,ws_2,...,ws_n,n\ge1\}$,则

$$r(s_t,ws_i)=\sum_{j\in\{s_t,s_{t+1}\}}p_t(j\mid s_t,ws_i)r(s_t,ws_i,j),$$

其中 $r(s_t,ws_i,s_{t+1})=quality(ws_i)$ 且 $r(s_t,ws_i,s_t)=0$.

(2) $|Execute(s_t)|\ge2$,不妨假设 $Execute(s_t)=\{t_1,t_2,...,t_n,n\ge2\}$,且对 $t_i\in Execute(s_t)$有

$$Invoke(t_i)=\{ws_{t_i}^1,ws_{t_i}^2,...,ws_{t_i}^{m_i},m_i\ge1\},$$

则

$$r(s_t,ws_{t_1}^{j_1},ws_{t_2}^{j_2},...,ws_{t_n}^{j_n})=\sum_{j\in\{s_t,s_{t+1}\}}p_t(j\mid s_t,ws_{t_1}^{j_1},ws_{t_2}^{j_2},...,ws_{t_n}^{j_n})r(s_t,ws_{t_1}^{j_1},ws_{t_2}^{j_2},...,ws_{t_n}^{j_n},j),$$

其中

$$r(s_t,ws_{t_1}^{j_1},ws_{t_2}^{j_2},...,ws_{t_n}^{j_n},s_{t+1})=\sum_{i=1}^{n}Quality(ws_{t_i}^{j_i}),$$

且 $r(s_t,ws_{t_1}^{j_1},ws_{t_2}^{j_2},...,ws_{t_n}^{j_n},s_t)=0$.

3.3 Web服务组合过程中MDP问题的最优方程

在服务组合过程中,如果每个被绑定服务都第 1 次执行成功,则所得报酬最大.反之,调用服务执行失败次数越多,报酬越低.因此,在 Web 服务组合过程中,在决策时刻 $t(t=0,1,\ldots,N)$,第 k 次对状态 s_t 选择实例服务的最优方程为

$$u_t^k(s_t)=\begin{cases}\max\limits_{a\in A(s_t)}\{r(s_t,a)+p_t(s_{t+1}\mid s_t,a)u_{t+1}^1(s_{t+1})+p_t(s_t\mid s_t,a)u_t^{k+1}(s_t)\}, & 1\le k<N(s_t)\\ 0, & k=N(s_t)\end{cases},$$

其中,$N(s_t)$是与状态 s_t 相关的、平衡了服务组合成功率与执行代价的常数;$N(s_t)$越大,则组合成功率越高,随之计算复杂度越大;$N(s_t)$的取值与 Web 服务的随机性和环境的动态性密切相关,具体可以根据实验来确定.

3.4 随机 QoS 感知的可靠 Web 服务组合算法

步骤 1.令决策时刻 $t=N$ 且对 $s_N\in S,u_N^1(s_N)=r_N(s_N)=0$.

步骤 2.如果 $t=0$,则 $\pi=(f_0^*,f_1^*,...,f_{N-1}^*)$ 为最优马氏策略,而 $u_0^1(s_0)$ 为最优值函数,算法停止.

否则,令 $t-1\Rightarrow t,k=1$ 后,进入步骤 3.

步骤 3.对任意 $s_t\in S$,有 $u_t^{N(s_t)}(s_t)=0$.当 $k<N(s_t)$计算

$$u_t^k(s_t)=\max_{a\in A(i_t)}\{r_t(s_t,a)+p_t(s_{t+1}\mid s_t,a)u_{t+1}^1(s_{t+1})+p_t(s_t\mid s_t,a)u_t^{k+1}(s_t)\},t+1\Rightarrow k.$$

步骤 4.记 $A_t^*(s_t)=\arg\max_{a\in A(s_t)}\{u_t^1(s_t)\}$,并任意取定 $f_t^*(s_t)\in A_t^*(s_t)$,从而确定了时刻 t 的决策规则 f_t^*.

步骤 5.返回到步骤 2.

3.5 Web服务组合过程中多执行路径的选择

对于包含 n 条执行路径 $ep_k(k=1,2,\ldots,n)$的 Web 服务组合 wsc,记 $u^*_{0,k}(s_0)$ 为第 k 条执行路径 ep_k 的最高报酬,且对应最优马氏策略为 $\pi^*_k=(f^*_0, f^*_1,..., f^*_{N-1})$,则该服务组合 wsc 所得最高报酬为 $u^*(wsc)=\max\limits_{1\le k\le n}\{u^*_{0,k}(s_t)\}$,最优策略为 $\pi^*(wsc)=\arg\max\limits_{1\le k\le n}\{u^*_{0,k}(s_0)\}$ 中任意之一.

4 性能模拟

为了验证 Web 服务 QoS 管理器对服务组合成功率的影响,以及服务组合方法的有效性,我们利用一系列随机产生的流程实例进行了大量实验.对流程图中各任务的候选服务,是在给定各 QoS 指标的数学期望和方差的前提下,根据实验规模按照正态分布随机产生,同时各性能指标的权值也是随机产生并保证其和为 1.实验环境为:Pentium Dual CPU 1.6GHz,1G RAM, Windows XP,matlab6.5.

4.1 Web服务QoS管理器仿真分析

为了比较体系结构中有无 Web 服务 QoS 管理器对动态环境中具有随机性的 Web 服务组合成功率的影响,分别对包含 10 个、20 个以及最多 80 个任务的顺序流程图进行了实验,其中每个任务均有 30 个候选服务,同时,对系统中每个状态只做一次决策,并且对每种情况都进行了 20 次实验,组合成功率为 20 次结果的平均值.从而得出体系结构中有无 Web 服务 QoS 管理器,以及随着流程中任务数的增加,服务组合成功率的变化,如图 3 所示.

在这里,服务能否成功组合与组件服务的动态变化密切相关.从图 3 中可以看出,对于具有相同任务数的流程模型,体系结构中如果具有 Web 服务 QoS 管理器模块,则组合成功率平均提高了 50%以上.例如,当流程中包含 10 个任务且不具有 Web 服务 QoS 管理器模块时,服务成功组合率为 0.5,反之,组合成功率提高到 0.8.由于实验中流程结构采用顺序结构,所以当流程图中任务数增加时,服务组合成功率都随之下降;但图 3 表明,具有 Web 服务 QoS 管理器模块时,服务组合成功率变化相对平缓,这是因为自适应模块降低了组件服务的真实 QoS 值与 UDDI 中注册值之间的差异.

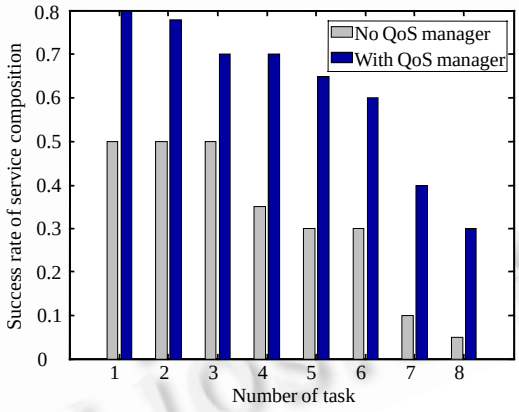


Fig.3 Relation between QoS manager and success rate of service composition

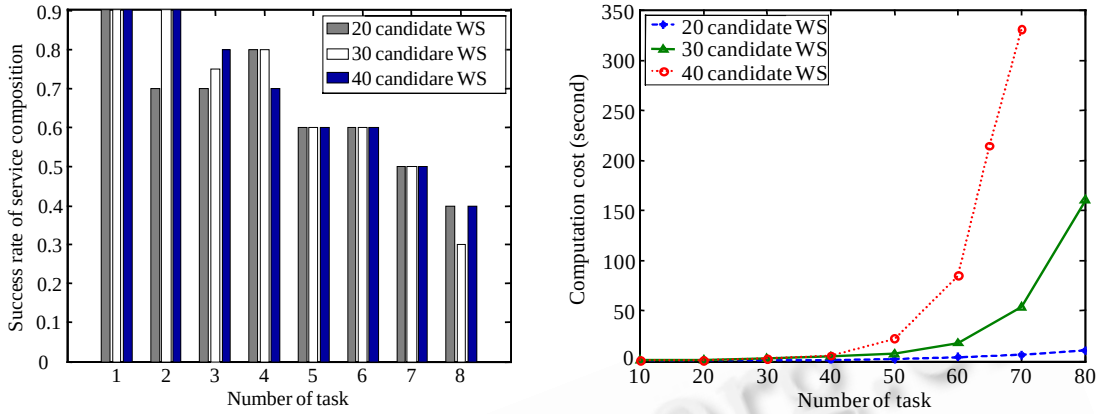
图 3 Web 服务 QoS 管理器对服务组合成功率的影响

4.2 不同任务数、不同候选服务数仿真对比

为了分析流程中任务数、每个任务的候选服务数以及系统每个状态下需要做出的决策次数与服务组合成功率和计算代价的关系,文中对顺序结构且分别包含 10、20 以及最多 80 个任务的流程图进行了实验,其中每个任务又分别有 20、30 以及 40 个候选服务,且对系统中每个状态只作一次决策,关于组合成功率和计算代价的变化如图 4 所示.

图 4(a)表明:对于任务数确定的流程图,当候选服务数在 20~40 之间变化时,对服务组合成功率的影响不很

明显;但随着任务数的增加,组合成功概率随之降低;候选服务数较多时,组合成功率变化平缓,说明当候选服务较多时,组合服务性能相对稳定.图 4(b)表明:当流程图中任务数小于 50 个时,候选服务在 20~40 之间变化对计算代价的影响不很明显,但是当任务数多于 50 个时,对于 40 个候选服务,其计算量巨幅增加.因此,在实际组合过程中,应该根据流程图中的任务数来控制候选服务数.



(a) Variation in composition success rate
(a) 组合成功率的变化

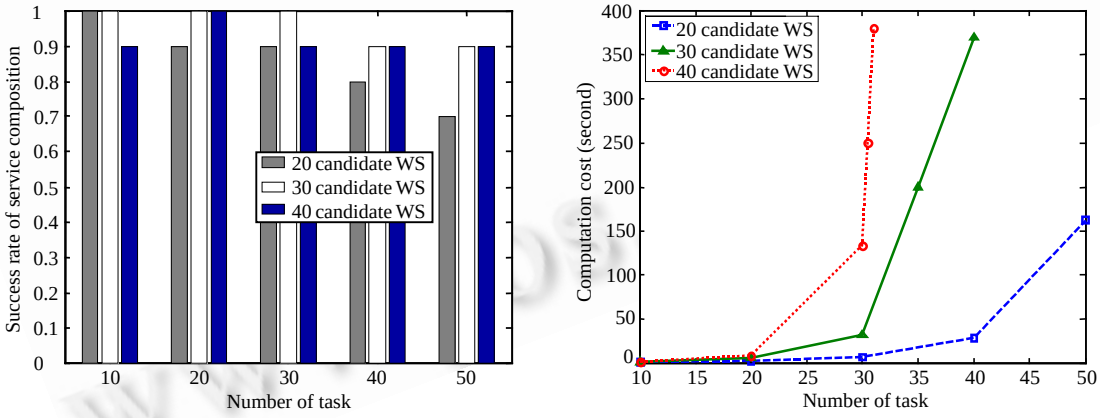
(b) Variation in computation cost
(b) 计算代价的变化

Fig.4 Influence on Composition success rate and computation cost under different number of tasks and candidates (here, each state with one decision-making)

图 4 任务数、候选服务数对组合成功率及计算代价的影响(其中,系统每个状态的决策次数为 1)

4.3 决策次数仿真分析

由于组合环境的动态性以及 Web 服务的随机性,对于系统的每个状态只作一次决策不可避免地要降低服务组合成功率.因此,为了提高组合成功率就必须增加每个状态的决策次数,但计算量也随之增加.为了分析系统每个状态的决策次数与服务组合成功率和计算代价的关系,对顺序结构且分别包含 10、20 以及最多 50 个任务的流程图进行了实验,其中每个任务又分别有 20,30 以及 40 个候选服务,且对系统的每个状态均作 3 次决策,关于组合成功率和计算代价的变化如图 5 所示.



(a) Variation in composition success rate
(a) 组合成功率的变化

(b)Variation in computation cost
(b) 计算代价的变化

Fig.5 Influence on Composition success rate and computation cost under different number of tasks and candidates (here, each state with three decision-makings)

图 5 任务数、候选服务数对组合成功率及计算代价的影响(其中,系统每个状态的决策次数为 3)

通过图 4(a)与图 5(a)的比较可以看出,如果对系统每个状态均作 3 次决策,服务组合的成功率将明显提高,平均可达 90%.但图 5(b)表明,随着决策次数的增加,当任务数多于 30 个时,计算量也大幅增加.因此,在实际组合过程中,应根据流程图中的任务数和候选服务数,以及要求的组合成功率来决定每个状态的决策次数.

5 结 论

本文针对 Internet 环境的动态性和 Web 服务的随机性,提出了 Web 服务各随机 QoS 指标的度量方法,其度量结果与其他方法相比,能够较准确地反映 Web 服务的真实 QoS;同时扩展了 Web 服务体系结构模型,使 Web 服务 QoS 值能够自适应地动态更新;基于 MDP 设计了随机 QoS 感知的可靠 Web 服务组合算法.最后通过实验说明了方法的有效性.

由算法描述和实验结果可知,本文的创新点以及优势在于:1) 将 Web 服务组合问题转化为随机型离散事件系统动态寻优的过程,并基于 MDP 设计了随机 QoS 感知的可靠 Web 服务组合算法;2) 将 Web 服务各 QoS 指标看作是随机变量,并给出了 QoS 模型中各指标的度量方法;3) 扩展了 Web 服务体系结构模型,使 Web 服务 QoS 值能够自适应地动态更新;4) 与其他静态规划组合方法相比,服务组合成功率以及组合服务的可靠性都有明显提高.进一步需要研究的问题是针对模糊、不确定约束研究可靠的服务组合方法及其验证技术.

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.

[2] Majithia S, Walker DW, Gray WA. A framework for automated service composition in service-oriented architectures. In: Bussler C, ed. Proc. of the European Semantic Web Symp. 2004. Berlin: Springer-Verlag, 2004. 269−283.

[3] Oh SC, Lee D, Kumara SRT. Web service Planner (WsPr): An effective and scalable Web service Web composition algorithm. Int'l Journal of Web Services Research, 2007,4(1):1−23.

[4] Ponnekanti SR, Fox A. SWORD: A developer toolkit t for Web service composition. In: Proc. of the 11th World Wide Web. New York: ACM Press, 2002. 83−107.

[5] Rao JH, Su XM. A survey of automated Web service composition methods. In: Cardoso J, Sheth AP, eds. Proc. of the 1st Int'l Workshop on Semantic Web Services and Web Process Composition. LNCS 3387, Berlin, Heidelberg: Springer-Verlag, 2005. 43−54.

[6] Blum AL, Furst ML. Fast planning through planning graph analysis. Artificial Intelligence, 1997,90:281−300.

[7] Benatallah B, Dumas M, Sheng QZ, Ngu AHH. Declarative composition and peer-to-peer provisioning of dynamic Web services. In: Agrawal R, Dittrich K, Ngu AH, eds. Proc. of the 18th Int'l Conf. on Data Engineering. San Jose: IEEE Computer Society, 2002. 297−308.

[8] Zeng LZ, Benatallah B, Dumas M. Quality driven Web service composition. In: Ellis A, Hagino T, eds. Proc. of the World Wide Web. Budapest: ACM, 2003. 411−421.

[9] Zhang CW, Su S, Chen JL. Genetic algorithm on Web services selection supporting QoS. Chinese Journal of Computers, 2006,29(7):1029−1037 (in Chinese with English abstract).

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

[11] Hu QY, Liu JY. An Introduction to Markov Decision Processes. Xi'an: Xi'an Electronic Science Technology Press, 2000 (in Chinese).

[12] Doshi P, Goodwin R, Akkiraju R, Verma K. Dynamic workflow composition using Markov decision processes. Int'l Journal of Web Services Research, 2005,2(1):1−17 .

[13] Gao AQ, Yang DQ, Tang SW, Zhang M. Web service composition using Markov decision processes. In: Fan WF, Wu ZH, Yang J, eds. Proc. of the 6th Int'l Conf. on Web-Age Information Management. LNCS 3739, 2005. 308−319.

[14] Eyhab AM, Mahmoud QH. Discovering the best Web service. In: Proc. of the World Wide Web. ACM, 2007. 1257−1258.

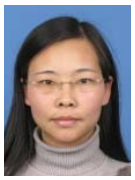
[15] Liu K. Applied Markov Decision Processes. Beijing: Tsinghua Press, 2004 (in Chinese).

附中文参考文献:

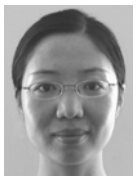
[9] 张成文,苏森,陈俊亮.基于遗传算法的 QoS 感知的服务选择.计算机学报,2006,29(7):1029−1037.

[11] 胡奇英,刘建庸.马尔可夫决策过程引论.西安:西安电子科技出版社,2000.

[15] 刘克.应用马尔可夫决策过程.北京:清华大学出版社,2004.



范小芹(1977−),女,山西岚县人,博士生,讲师,主要研究领域为 Petri 应用,服务计算.



王俊丽(1978−),女,博士,讲师,主要研究领域为本体理论及服务计算.

蒋昌俊(1962−),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为 Petri 理论及应用,网格计算,服务计算.

庞善臣(1974−),男,博士,副教授,CCF 高级会员,主要研究领域为 Petri 理论及应用.