一种考虑QoS动态变化的服务选择方法

孙若男[1] 张 斌[1] 赵秀涛[1] 李 飞[1,2]
(东北大学信息学院计算机应用研究所 沈阳 110819)[1] (东北大学秦皇岛分校 秦皇岛 066004)[2]

摘 要 随着Web服务相关技术的高度发展,网络上存在着多个完成相似功能的Web服务。如何选择适合的服务来生成满足用户要求的组合服务成为服务领域的重要研究课题之一。现有的基于服务质量QoS(Quality of Ser-vice)的服务选择方法通常都是假定服务提供者发布的QoS数值是真实且固定不变的,然而服务的质量往往在实际运行中发生变化。为此,针对该问题提出了一种考虑QoS数据动态变化的服务选择方法。该方法引入了QoS分时可靠的思想,将作为选择依据的QoS数值根据以往的服务执行情况给出不同时段上的不同表现数值,这样可以更加贴合地描述服务运行的实际情况。该方法按服务在不同时间段上的可靠性变化划分为不同的子服务,利用冗余的思想在不同的时间段下为用户提供多个满足要求的备选服务。最后通过一组模拟实验说明该方法的可用性和有效性。

关键词 服务选择算法,分时可靠模型,服务集预处理方法
中图法分类号 TP319 **文献标识码** A

Service Selection Approach Considering the Dynamic Change of QoS

SUN Ruo-nan[1] ZHANG Bin[1] ZHAO Xiu-tao[1] LI Fei[1,2]
(Computer Application Technology Research Institute, Northeast University, Shenyang 110819, China)[1]
(Computing Center, Northeastern University at Qinhuangdao, Qinhuangdao 066004, China)[2]

Abstract With the high development of Web services technology, there are several features similar Web service on Internet. How to select suitable services to meet the needs of customers and generate the service composition has become one of the major issues. Based on the existing QoS service selection method is usually assumed that service provider's value is real and fixed, but the quality of services often changes in the actual operation. Therefore, the paper put forward a dynamic change of data to consider QoS service selection method. The method introduces a reliable QoS sub-periods, which can be more relevant to describe the actual operation of service. The method according to service at different time changes on the reliability of the sub-divided into different services, using redundant, under a different time period to meet the requirements to provide users with multiple options for services. Finally, a simulation experiment shows the availability and effectiveness of the method.

Keywords Service selection algorithm, Reliable model of time-sharing, Service set pretreatment

1 引言

近年来,面向服务体系结构高度发展,Web服务(Web Service)技术作为其中重要的实现方式,得到了广泛的关注和长足的发展。当前,Internet上存在为数众多且内容丰富的服务。单一Web服务功能有限,而服务组合通过选择一系列服务并将其整合来提供更加完善的功能。在Web服务的相关研究中,如何选择出适合的组合服务是一个极具研究意义的问题[1-8]。

服务组合通过整合现有简单服务,来形成功能强大的复杂服务,有效地避免了服务软件的重复开发,充分体现了Web服务支持快速应用集成的优势。尽管服务组合具有相当多的优点,但是对于服务组合相关技术的研究与实施应用仍有大量的研究空间。

1.1 问题描述

Web服务处于高动态的网络环境中,随着服务的执行,其所处的网络环境可以存在多样的变化,比如服务器运行差错、网络中断、服务执行中挂起等多种不良情况。这些变化都有可能直接导致服务组合在执行上不满足于整体的约束条件。因此,如何在动态环境下满足组合服务的可靠性约束,逐渐成为了目前服务组合研究方向的热点问题,服务组合的可靠性直接决定了Web服务是否能够成功应用。当前针对QoS属性的服务选取问题中大部分直接参与选取的服务QoS属性值仅为注册服务时服务提供者提供给UDDI的相关参数数据,但是随着服务的执行,服务所处环境的变化,服务的QoS属性一直在发生着变化,统一使用一个固定值来选取服

到稿日期:2011-01-25 返修日期:2011-03-05 本文受核高基科技重大专项基金(2010ZX01045-001-008)资助。
孙若男(1984—),女,博士生,主要研究方向为Web服务、云计算,E-mail:sunruonan@126.com;张 斌(1964—),男,博士,博士生导师,主要研究方向为Web信息处理、数据挖掘;赵秀涛(1985—),男,博士生,主要研究方向为服务演进;李 飞(1975—),男,博士生,讲师,主要研究方向为Web服务、网络监测。

务有可能会不适应现在服务所处的动态状态。所以本文针对服务的动态 QoS 变化，为了更加适应服务所处环境，更准确地描述服务在不同时间段上的状态信息，提出了一个全新的 QoS 属性表达方式，亦即将服务的属性值细化到以小时为单位的时间轴上。这样可以更加细化服务的属性表示，从而在选取的过程中可以获得面向动态环境具有较好适应性的组合服务。分时评估服务的相关属性是本文的创新点之一。

1.2 相关研究

目前基于 QoS 的 Web 服务选取问题已经成为 Web 服务领域研究的热点之一。服务选择的结果不仅直接关系到服务能否成功组合，而且对组合服务的质量有着重要的影响[1,2]。但在服务选取的研究中，对于 QoS 中的参数评估多是将其设定为一个固定的值[3,4]，而由于环境及服务的动态性，如果对于每个服务仅仅是使用其注册在服务注册中心的提供属性值来进行服务选取，则有很大的可能在执行的过程中选取出的组合服务会出现问题，此时，再进行后续替换的依据仍然是基于 Web 服务的固定的某一属性值，而该数值在此时是否可用仍有待商榷[16]。所以，目前的选取过程存在的一个极大的问题就是适应性不强，选取的基础无法使选取出的服务很好地适应外部环境的变化。

目前服务运行在动态的网络中，网络状态多变，比如网络延迟、带宽变化以及服务网络间的可用等。同时服务也是随时变化的，服务的 QoS 可能出现剧烈波动的情况，服务本身出现持久性或挥发性的变化。所以服务选择方法必须尽量适应各种不可预知的服务变化以适应用户的约束条件。

2 CDCQ：一种考虑 QoS 动态变化的服务选择方法

针对上述相关服务选择方法的缺点，本文提出了一种考虑 QoS 动态变化的服务选择方法(Service Selection Approach Considering the Dynamic Changeof QoS，CDCQ)。它在借鉴目前提出的基于 QoS 的服务选择方法基础上，引入了一种分时可靠的思想，即将以往作为服务选择依据的固定的服务 QoS 数值按照其服务执行日志中的数据进行不同时段上的划分，在此基础上选择出的组合服务既满足了用户需求，又可以更好地适应实际情况。

本文中所涉及到的服务相关定义，比如服务、任务、组合服务等同于以往的定义，与其它服务选取方法区别的地方在于针对选取服务 QoS 的分时想法。故在此给出以下两个定义。

定义 1(子任务) 用以表示同一个任务细化在不同时间段上运行的部分。子任务仅是提出的一个概念，而非功能上拆分出的新任务。任务按时间段上的可靠性数值变化映射在时间轴上的节点划分出的子任务为 $T_{i,j}$，表示的是组合服务流程上的第 i 个任务中的第 j 个时间段上的任务。

子任务表示为 $T_i=\{T_{i,1},T_{i,2},\cdots,T_{i,j}\}$。

定义 2(子服务) 用以表示同一个服务细化在不同时间段上运行的部分。同子服务一样，子服务也非功能上拆分出的新服务。服务按时间段上的可靠性数值变化映射在时间轴上的节点划分出的子服务为 $WS_{i,j,k}$，表示的是组合服务流程上的第 i 个任务中的第 j 个时间段上的第 k 个服务实例。

子服务表示为 $WS_{i,j}=\{WS_{i,j,1},WS_{i,j,2},\cdots,WS_{i,j,k}\}$。

子任务与子服务的划分如图 1 所示。

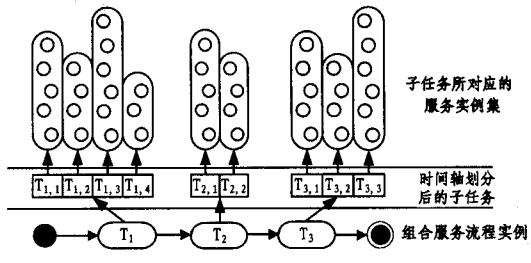

图 1 服务选取的分层示意图

2.1 组合服务选择过程描述

服务选择的目标即为满足用户对于服务组合的质量要求，并优化整个服务组合选择过程。

本文提出的 CDCQ 方法以服务组合的可靠性作为服务选择的目标，在满足服务组合执行时间约束的前提下，为组合服务流程上的各个任务选取出多个待执行的备选服务，这样每个任务都对应一系列可以选用的服务，并不是只找到单个服务去执行，形成例如图 2 所示的待执行方案。

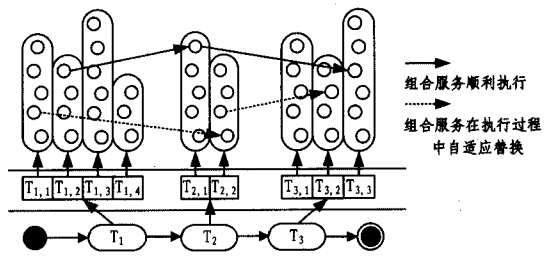

图 2 服务组合执行方案

图 2 中的每个任务都对应一系列待执行备选服务，分别用 S_1、S_2、S_3 表示任务 T_1、T_2、T_3 对应的备选服务集，S_1'、S_2'、S_3'表示对应的选择后的待执行备选服务集。可以看出 $S_1'\subseteq S_1$、$S_2'\subseteq S_2$、$S_3'\subseteq S_3$，即经过本文的服务选取方法使得每个任务对应一系列可以选用的服务，这些待执行备选服务是各个任务备选服务的子集。用 $\max(qf_{T_i})$表示第 i 个任务的待执行备选服务集中最大的约束值，则选取出的待执行方案满足 $\sum_{i=0}^{n}\max(qf_{T_i})\leqslant Q_c$。

在用户要执行服务组合时，只需要在各个任务的待执行备选服务集中进行局部选取即可。而当执行阶段某一服务出现问题时，可以在该任务的待执行备选服务中选择一个次优的服务进行替换，或许替换后的服务组合不一定是最优解，但它是用户可以接受的近优解，如图 2 所示。

在 CDCQ 方法中不需要考虑服务替换后所组成的新执行计划是否满足全局约束，因为在为每个任务选出一系列待执行备选服务时，都是在满足用户全局约束的前提下进行的，所以从各个任务中任选一个备选服务所组成的组合服务的总约束也一定满足用户约束。只有当该任务对应的所有待执行备选服务都不可用时，才需进行服务重选取，从而降低了服务重新选择的概率，这在很大程度上保证了组合服务的可靠性，降低了服务选择的代价。

根据以上小节中的描述，在满足服务组合整体执行时间约束的条件下，使组合服务可靠组合的服务流程的各个任务对应尽量多的待执行备选服务。

(1)组合服务执行时间不大于用户对于执行时间的约束；

(2)组合服务的全局可靠性最优；

(3)组合服务在时间分段上也具有高可靠性。

综上所述，可以把上面讨论的组合服务选取问题转化为下面的数学问题进行解决：

$$\begin{cases} s.t.\ ComputeEt(qws_{i,j,k}) \leqslant Q_{CS} \\ \mathrm{Max}(Rel_{CS}) \\ \mathrm{Max}(Rel_{CS_t}) \\ s.t.\ \forall WS_{i,j,k} \subseteq S_{i,j}' \subseteq S_i',\ i=1,2,\cdots,n; \\ \quad j=1,2,\cdots,m_i;k=1,2,\cdots,m_{i,j} \end{cases} \tag{1}$$

式中，Rel_{CS} 是指组合服务的可靠值，Q_{CS} 为用户对服务组合的全局约束条件，$WS_{i,j,k}$ 表示第 i 个任务的第 j 个时间段上的第 k 个服务实例，$qws_{i,j,k}$ 表示上述服务实例的属性值，$S_{i,j}'$ 表示第 i 个任务的第 j 个时间段上的备选服务集，S_i' 表示第 i 个任务的全部待执行备选服务集，n 是组合服务流程中的任务数，$m_{i,j}$ 表示第 i 任务的第 j 个时间段上的子任务所对应的待执行备选服务个数，m_i 表示第 i 任务对应的待执行备选服务个数，$WS_{i,j,k} \in S_{i,j}' \in S_i'$ 表示执行方案中的各个服务实例来自各自对应的服务集，$ComputeEt()$ 是用以计算当前组合服务流程的针对执行时间约束的函数值。

由上式可知，从各个任务的备选服务集中选出一个服务实例组成执行方案，该方案的约束值均不会大于用户提出的对于组合服务全局的约束。所以在进行服务替换时不需要考虑替换后组成的新的执行方案是否符合用户约束的情况。

2.2 QoS 模型

针对 CDCQ 方法中的分时可靠思想，在本小节中给出对应的服务、任务和组合服务的 QoS 模型。

2.2.1 组合服务流程

目前服务的相关操作中，多将服务组合流程看作一个基于服务的工作流[6]，其与 WFMC 定义的工作流[7]的 4 种基本模型对应。本小节给出了服务组合流程的基本模型：串行模型、并联模型、选择模型、循环模型，大部分服务组合流程都可以由这 4 种结构进行有限次的嵌套递归得到，组合服务流程的 QoS 参数也可以通过基本模型的 QoS 评价方法来获得。

图 3 为这 4 种结构的形式化表示。

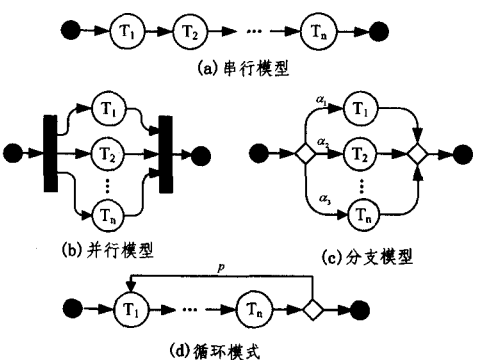

图 3 服务组合流程基本模型

表 1 复合任务类型的属性计算公式

复合任务类型	执行时间 Et(time)	可靠性 Rel(reliability)
串行模型 sf(sequence flow)	$T_{CS}=\sum_{i=1}^{n} Et_i$	$Rel_{CS}=\prod_{i=1}^{n} Rel_i$
并行模型 pf(parallel flow)	$T_{CS}=\mathrm{Max}(Et_1,Et_2,\cdots,Et_n)$	$Rel_{CS}=1-\prod_{i=1}^{n}(1-Rel_i)$
分支模型 cf(condition flow)	$T_{CS}=\sum_{i=1}^{n} Et_i\alpha_i$	$Rel_{CS}=\sum_{i=1}^{n} Rel_i\alpha_i$
循环模型 lf(loop flow)	$T_{CS}=p\sum_{i=1}^{n} Et_i$	$Rel_{CS}=\prod_{i=1}^{n} Rel_i p$

针对图 3 中的 4 种结构模型，可得到复合任务类型的属性计算公式，见表 1。其中，分支模型设中第 i 个分支被选中的概率是 α_i，$\sum_{i=1}^{n}\alpha_i=1$；循环模型中循环次数为 p。

要指出的是，对于分支任务来说具体选择哪一条分支路径执行，在服务组合执行前是一个未知的随机事件，在本文中假设每条路径被选中的概率相同，即整个分支任务的可靠性和执行时间均等于各支路径值的均值。并行任务的执行时间取决于执行时间最长的那一条并行执行路径。在此基础上按照在组合流程中嵌套的层次进行递归处理，就能求出任意嵌套层次中每个任务的可靠性取值。

从上面的各种流程结构可以看出，分支、并行以及循环任务中都嵌套了顺序任务，而顺序任务中又可以嵌套包含原子任务、分支任务、并行任务和循环任务。因此本文讨论的组合服务流程结构从整体上可以等效为一个由原子任务、分支任务、循环任务，或并行任务经过有限次递归嵌套而构成的顺序任务。

服务、任务、组合服务的 QoS 模型即在此结构下给出形式化定义。

2.2.2 服务 QoS 模型

本文将服务按不同时间段上的不同表现拆分为各时间段上的子服务，即在选取过程中划分的最小单位为子服务。服务 WS 的 QoS 属性模型可以由子服务的属性模型集合来描述，如式(2)所示。

$$qws=\{qws_1,qws_2,\cdots,qws_n\} \tag{2}$$

式中，n 表示该服务在时间轴上划分的子服务个数，此时服务的属性由被划分为由 n 个小时间段上的子服务属性来描述。而在第 i 个时间段上的子服务的属性模型可描述为式(3)。

$$qws_i=\{Et_{WS_i},Rel_{WS_i}\} \tag{3}$$

式中，qws_i 对应第 i 个时间段上子服务的属性值，Et_{WS_i} 表示子服务的执行时间，Rel_{WS_i} 表示子服务的可靠性，即服务可以被正确发送、执行并且返回结果的比率，它可以根据服务日志中的历史记录来计算：$Rel_{WS_i}=\frac{m}{n}$，m 为服务在时间段内所成功调用的次数，n 为服务在该时间段内执行的总次数。

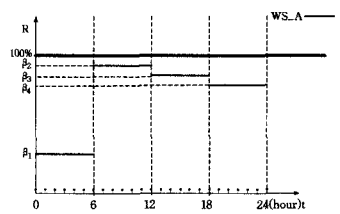

图 4 一个服务的分时 QoS 示意图

从图 4 中可以明显看出服务的 QoS 属性是分段的，不同的子服务的执行时间和可靠性的数值是不同的。同一服务中，不同的子服务间的关系呈分支结构，由 2.2.1 节中可知，将服务的执行时间取值为各子服务的执行时间值的均值，即服务的执行时间为 $\frac{\sum_{a=1}^{n} Et_{WS_i}}{n}$，代入式(3)可得到式(4)，即子服务的 QoS 计算公式。

$$qws_i=\left\{\frac{\sum_{a=1}^{n} Et_{WS_i}}{n},Rel_{WS_i}\right\} \tag{4}$$

按照上述描述模型，将服务 WS_A 的相关参数予以图形化说明(见图 4)。

服务 WS_A 被分割为 4 个子服务的属性，在时间段 $t_1(0,6)$、$t_2(6,12)$、$t_3(12,18)$、$t_4(18,24)$上的执行时间分别为 ρ_1、ρ_2、ρ_3、ρ_4，可靠度为 β_1、β_2、β_3、β_4。按照上述公式可得如下描述。

子服务的描述模型为 $qws_i=\left\{\frac{\rho_1+\rho_2+\rho_3+\rho_4}{4},Relws_{t_i}\right\}$，其中，$Rel_{t_1}(S)=\beta_1$，$Rel_{t_2}(S)=\beta_2$，$Rel_{t_3}(S)=\beta_3$，$Rel_{t_4}(S)=\beta_4$。

服务的描述模型为 $qws_i=\{qws_{s_1},qws_{s_2},qws_{s_3},qws_{s_4}\}$。

上述公式描述出了“分时可靠”这个本文研究的理论基础。当然，此 QoS 模型还可以根据用户的需求进行相应的扩展，比如增加服务价格、可用性等评价因素，但这些目前不在本文的讨论范围内。

2.2.3 任务的 QoS 模型

假设任务 T_i 已经在数据预处理阶段后得到了时间段上的分割，如图 5 所示。T_i 已经在时间轴上分割为 n 段，t_j 时间段上的服务为 $WS_{i,j,1}$ 到 WS_{i,j,k_j}，即在服务集预处理完成这个时间段上有 k_j 个子服务来负责该时间段上任务的可靠性保障。

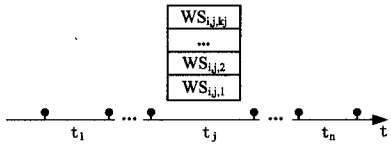

图 5 任务的 QoS 模型

由于服务的 QoS 属性是分段的，因此其对应任务的执行时间和可靠性属性在不同的时间段上的数值也必然是不同的。

子任务执行时间与服务执行时间的处理相同，均是取均值。针对可靠性来说，在预处理阶段结束后每个子任务都对应适量的子服务。此处相当于为子任务 $T_{i,j}$ 使用 k_j 个子服务来保证其可靠性。所以在任务 T_i 的第 j 个时间段上的子任务 $T_{i,j}$ 的执行时间和可靠性数值可量化为式(5)。

$$q_{T_{i,j}}=\left\{\frac{\sum_{l=1}^{k_j}Et_{WS_{i,j,l}}}{k_j},1-\prod_{l=1}^{k_j}(1-Relws_{i,j,l})\right\} \tag{5}$$

由于同一个任务在不同的时间分段上的子任务为分支关系，因此仍将任务的执行时间取子任务执行时间的均值，则有式(6)。

$$q_{T_i}=\left\{\frac{\sum_{j=1}^{n}Et_{T_{i,j}}}{n},1-\prod_{l=1}^{k_j}(1-Relws_{i,j,l})\right\} \tag{6}$$

所以，再将式(3)中的子任务执行时间计算公式代入式(5)中即可得到本文所需的任务 T_i 的 QoS 模型中执行时间和可靠性的计算模型，见式(7)。

$$q_{T_i}=\left\{\frac{1}{n}*\sum_{j=1}^{n}\frac{\sum_{l=1}^{k_j}Et_{WS_{i,j,l}}}{k_j},1-\prod_{l=1}^{k_j}(1-Relws_{i,j,l})\right\} \tag{7}$$

2.2.4 复合任务的 QoS 模型

在本文中组合服务被等效为一个顺序执行的复合任务，故可以直接用复合任务的 QoS 模型来计算服务组合的相关组合。

假设各复合任务中的任务数为 m。

(1)复合任务的可靠性计算公式见表 2。

(2)复合任务的执行时间计算公式见表 3。

表 2 冗余机制下的复合任务可靠性计算公式

复合任务类型	可靠性取值(Rel_{CS})
串行模型 sf(sequence flow)	$Rel_{CS}=\frac{1}{n}*\prod_{i=1}^{m}\{\sum_{j=1}^{n}[1-\prod_{l=1}^{k_j}(1-Relws_{i,j,l})]\}$
并行模型 pf(parallel flow)	$Rel_{CS}=\frac{1}{n}*\mathrm{Min}\{\sum_{j=1}^{n}[1-\prod_{l=1}^{k_j}(1-Relws_{i,j,l})]\}$，其中 $i\in(1,m)$
分支模型 cf(condition flow)	$Rel_{CS}=\frac{1}{n^2}*\sum_{i=1}^{m}\sum_{j=1}^{n}[1-\prod_{l=1}^{k_j}(1-Relws_{i,j,l})]$
循环模型 lf(loop flow)	$Rel_{CS}=\frac{1}{n^p}*\sum_{i=1}^{m}\sum_{j=1}^{n}[1-\prod_{l=1}^{k_j}(1-Relws_{i,j,l})]$，其中 p 为循环次数

表 3 冗余机制下的复合任务执行时间计算公式

复合任务类型	执行时间取值(Et_{CS})
串行模型 sf(sequence flow)	$Et_{CS}=\frac{1}{n}*\frac{1}{k_j}\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{l=1}^{k_j}Et_{WS_{i,j,l}}$
并行模型 pf(parallel flow)	$Et_{CS}=\frac{1}{n}*\frac{1}{k_j}\mathrm{Max}(\sum_{j=1}^{n}\sum_{l=1}^{k_j}Et_{WS_{i,j,l}})$，其中 $i\in(1,m)$
分支模型 cf(condition flow)	$Et_{CS}=\frac{1}{m}*\frac{1}{n}*\frac{1}{k_j}\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{l=1}^{k_j}Et_{WS_{i,j,l}}$
循环模型 lf(loop flow)	$Ec_S=\frac{1}{n^p}\left(\sum_{j=1}^{n}\frac{\sum_{l=1}^{k_j}Et_{WS_{i,j,l}}}{k_j}\right)^p$，其中 p 为循环次数

3 备选服务集预处理

根据上文的服务分时可靠模型，可知在不同时间段上的服务质量是不同的，这就有可能存在某个时间段内服务 QoS 过低，或服务质量不稳定等多种情况。通过备选服务集预处理的方法可以有效减少无效的 QoS 服务，减轻后续选择过程中的压力。服务集预处理方法的目的就是为了提前将完成某一功能的一组候选服务按照某种规则规整，以便于后续的选取工作。

3.1 问题描述

目前的组合服务可靠性保障大多在服务选取或者运行阶段采取某些机制来保证，鲜有在服务选取之前进行服务选取集预处理的步骤。

在 CDCQ 方法中，首先将备选服务集进行适当的删减来降低后续服务选择工作的压力。

服务集预处理的聚类操作中所依靠的数据是通过解析系统中的执行日志库得来的。

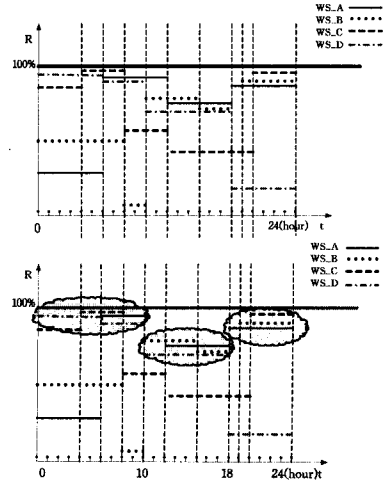

图 6 可靠性密度值聚类结果示意图

在服务执行日志库中可以完整地记录服务执行过程中的

相关状态，比如服务所在主机是否正常运行，部分线路的通断，每次调用服务的执行时间以及服务在格式端上的不同可靠性表现。在服务及预处理阶段通过解析服务器上的执行日志，可以得到每个服务在时间轴分布上的不同的可靠性数值，如图 6 所示。

3.2 子服务间的可靠性密度

基于密度的聚类算法的核心思想是“寻找被低密度区域分离的高密度区域”，套用到本文的情景环境中即为通过逐渐将高可靠且高密度分布的子服务聚类在一起而聚簇成子任务。通过一个任务中的多个子任务来保证任务的高可靠性。

服务的 QoS 的属性值相关分布是独立的，所以本文可以采取聚类的方法从中找出可靠性映射到时间轴上的分段。预处理阶段的目标是将时间轴上邻接且可靠度数值相近的子服务聚类为一个整体，所以相对来说使用密度聚类算法来进行时间轴分段比较适合。

基于服务可靠性密度的聚类算法通过计算同一任务中不同的子服务间的可靠性差值，将该值的倒数设为密度，利用邻近的可靠性差值密度变化趋势，将满足聚类条件的子服务逐步加入到当前簇中，不断地循环此步骤，直到达到聚类终止条件。由于算法利用密度的变化情况来聚类，因此对于本文中的这种可靠性密度分布不均匀的 Web 服务 QoS 集合等具有良好的应用效果。

定义 3(时间段关系) 把时间段从绝对的或瞬时的参考体系中分离出来，则两个区间之间只存在相对的关系，有如下 7 种关系组成。除了第一种 before 关系中两个时间段之间没有交集，其他的时间段上都存在相交的关系。这就对应于不同的时间段上的相交关系。在后续可靠性密度考察中需要利用此规则来判定两时间段的相交关系。

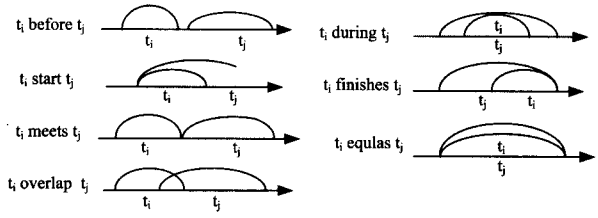

图 7 时间段间关系

定义 4(可靠性密度) 亦即时间点相邻的两个子服务上的可靠性数值差值绝对值的倒数。

设子服务 WS_{i,j,k_1} 所在时间段为 $t_{k_1}=\langle t_{i_1},t_{j_1}\rangle$，可靠性数值为 r_{k_1}，子服务 WS_{i,j,k_2} 所在时间段为 $t_{k_2}=\langle t_{i_2},t_{j_2}\rangle$，可靠性数值为 r_{k_2}。由图 7 可以判断两时间段间的关系，若存在相交关系，此时两服务间的可靠性差值为 $Diff_r=|r_{k_1}-r_{k_2}|$。因为两个子服务间的相对密度值是无向性的，故有子服务密度计算式(8)。

$$Den_{(WS_{i,j,k_1},WS_{i,j,k_2})}=Den_{(WS_{i,j,k_2},WS_{i,j,k_1})}=\frac{1}{Diff_r}=\frac{1}{|r_{k_1}-r_{k_2}|} \tag{8}$$

由上式可知，两个服务间的可靠性差异越小，则两者间的密度越大。通过不断地迭代已有的子服务，将密度相近的子服务逐渐聚合在一起，完成时间轴划分要求。在本文后续的算法描述中，将可靠性密度简称为密度。

基于密度的聚类算法的思想是：首先在数据集 D 中找到任意一个核心对象 p，求出 p 的核心集合，得到初始类 C；然后由初始类 C 开始进行类的扩展，直到没有任何对象可以归入该类；重新在 D 中寻找任意一个未归类的核心对象 q，重复上述过程，直到没有任何对象可以归入任何类，算法结束。初始类 C 的扩展过程分两步进行：首先，对 p 的核心集合进行扩展，得到类 C 的扩展核心集合；然后，根据关于扩展核心集合中核心对象的密度可达这一条件，对类 C 进行扩展。

3.3 基于子服务可靠性密度的时间段划分算法

基于密度的聚类算法是针对独立任务所包含的相关子服务来进行聚类的。其工作原理是寻找被低密度区域分离的高密度区域。它的核心思想是通过扩展核心点，将密度相连的所有数据识别出来，作为一个簇。

聚类结果示意图如图 6 所示，该图中将完成同一个任务的 4 个服务按照其在时间轴上不同时间段内的子服务可靠性密度聚类为 3 个部分。

算法 基于子服务可靠性密度的聚类算法

输入：同一个任务下的服务 QoS 属性
输出：任务在时间轴上聚类出的子服务簇
BEGIN
初始化子服务的标志位 $flag=0$
DO
在 $flag=0$ 的子服务中查找中心点 WS_ω
IF(WS_ω 唯一)
return WS_ω
ELSE
return 左侧第一个 WS_ω
END IF $WS_\omega.flag=1$
IF($Den>\gamma$)
$WS_\omega.flag=-1$
ELSE
$WS_\omega.flag=1$
以中心点将簇中的子服务划分为左右两个部分
IF(左侧簇中有子服务)
FOR(WS_ω=左侧密度最大的子服务；左侧子服务数；WS_ω—)
密度聚类操作
END FOR
END IF
IF(右侧簇中有子服务)
FOR(WS_ω=右侧密度最大的子服务；左侧子服务数；WS_ω——)
密度聚类操作
END FOR
END IF
IF(无法继续聚类)
调整阈 γ
WHILE(时间轴上完整的覆盖)
END DO WHILE
END

本文算法的聚类依据是子服务间可靠性的密度模式。

服务预处理阶段除了为后续的选取工作提供了初始的可靠性数值较高、差值较小的初始候选服务实例之外，也很好地保证了组合服务中每个独立任务的局部的可靠性。

4 模拟实验

在这一节中，对本文提出的考虑 QoS 分时可靠的服务选择方法的可用性和有效性进行实验考量。

4.1 实验场景设置

实验计算机为 Intel® Core(TM)2 CPU 6300@1.86GHz，

RAM4GB,硬盘 250G。服务选择的实际编码部分使用粒子群算法,采用 Java 语言实现。

实验所用的组合服务模型中每一个节点代表组合服务中的一个任务,如图 8 所示,在该组合服务流程中共包含 5 个任务,其中已包含了顺序、并行和分支结构。

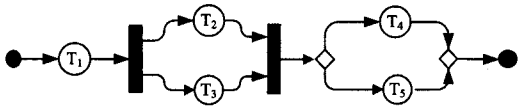

图 8 实验用组合流程实例

目前在网络中并没有足够多可免费调用的满足功能和非功能需求的服务实例,缺乏统一的 Web 服务的测试样本集。本文通过随机模拟方式生成了一些数据作为单个服务的实验依据,利用 Java 语言编写了一个随机函数用以在一定范围内生成 Web 服务中的相关参数,参数取值范围设置为执行时间 $0 \leqslant Et \leqslant 10s$,可靠性 $0 \leqslant Rel \leqslant 1$;同时随机地给出服务在时间轴上的分段信息,每个服务的分段数量 $1 \leqslant Sec \leqslant 5$,将服务在不同时间范围内的不同的执行时间和可靠性数值写入 MySQL 数据库中,用于后续实验。候选服务相关信息示例见表 4。

表 4 候选服务相关信息

任务	服务	子服务	t_{begin}	t_{end}	Et	Rel
T_1	$WS_{1,1}$	$WS_{1,1,1}$	0	11	2.6987	0.8563
T_1	$WS_{1,1}$	$WS_{1,1,2}$	11	20	3.2486	0.7895
T_1	$WS_{1,1}$	$WS_{1,1,3}$	20	24	2.4896	0.4793

注意,此处的 $WS_{i,j,k}$ 表示的是任务服务 T_i 所对应的服务 $WS_{i,j}$ 在第 k 个时间段的子服务。

4.2 有效性实验

该实验的目的是用于验证本文算法找到 QoS 全局最优组合服务的有效性。该实验通过算法执行的 CPU 开销来验证本文选取算法解决实际问题的有效性。实验分别从以下两个角度进行了对比。

为实验用组合服务流程中的 5 个任务随机生成多组初始数据集,对其进行服务的预处理操作。由于使用随机函数对其进行赋值,而服务的初始属性值对预处理中的聚类算法的影响极大,因此得到的服务集预处理的结果也大不一样。

在本实验中生成了 8 组不同的初始数据,聚类后的粒子编码长度(不包含粒子编码开始和结束处的两个虚服务)分别为 10、13、17、19、21、23、25、31。按照文中的算法对其进行选取操作,得到的可靠性数值在图 9 标注出来。

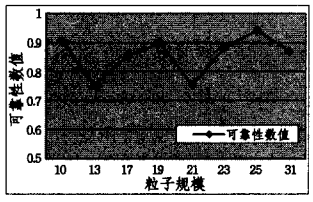

图 9 不同粒子规模下的可靠性数值

经过上述实验可以得出在粒子编码规模不同的情况下,均可以使用本文中的面向组合服务自适应执行的服务选取方法来选取出满足约束条件的备选服务集。但是由于实验结果中可靠性的分布很随机,组合服务的可靠性随着粒子规模的增大并没有表现出有规律的变化,此处并不能总结出相关规律。

本实验的主要目的是验证所提选取算法针对服务选取工作的有效性。对于选出的组合服务是否可行、算法是否可行将在下一小节中使用可行性实验来论证。

4.3 可行性实验

针对本服务选取方法的可行性分别对从全局可靠性与穷举算法得出的全局最优解可靠性求比率来验证可行性。

本实验的目的是验证面向组合服务自适应的服务选取方法,找到高可靠的全局最优解的可行性。作为参考比较的另一种方法是穷举法,即逐个地在不同的时间段上穷举计算出满足约束条件的可执行组合服务的高可靠最优解。然后分别在不同的条件下,将本文算法与之进行比较。

图 10 中显示的是,在粒子规模分别为 10、13、17、19、21、23、25、31 时,迭代 200 代所获得的可靠性指标的全局最优结果的百分率,此百分率指的是在相同服务结点和服务规模的数据集条件下算法运行 20 次所得的非劣解集与按可靠性穷举出的最优解的统计概率。

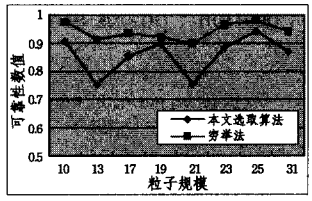

图 10 两种算法的可靠性数值对比图

从图 11 可以看出,实验结果显示,在有限次迭代进化(200 次)条件下本文的服务选取方法所选取出的不同长度的子任务划分下组合服务的可靠性与穷举法选出的组合服务的最优解可靠性比值在 90%左右。因此,包含服务集预处理阶段的服务选取算法解决约束条件下的可靠性最优解问题是可行的。

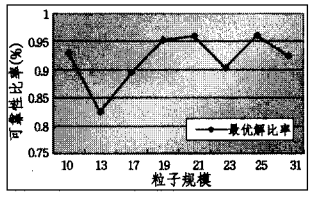

图 11 最优解比率

结束语 服务选择是服务相关领域研究的关键问题之一,针对目前全局最优服务选择算法并未考虑在动态网络环境中的服务 QoS 变化实时性,在引入分时可靠思想的基础上,设计了一种分时段的 QoS 可靠性评价模型。同时针对该模型,提出了一种考虑 QoS 动态变化的服务选择方法。

虽然按照本文的方法可以选择出满足要求的可靠性较高的组合服务,但文中的方法仍有较明显的问题尚待解决。比如在服务集预处理阶段,依据可靠性数值的相对密度在时间轴上进行聚类。针对聚类问题有多种解决方法,本文选择的是基于密度的聚类算法。文中方法虽然可以达到聚类的目的,但其效率并不是很高。同时本文在初始数据过于分散的情况下存在不可聚或聚类结果过多的可能;而且本文的算法中只考虑了执行时间和可靠性这两个因素,在服务的 QoS 中还有很多其他的考察指标需要引入。在后续的研究工作中一是可针对时间段划分找出具有更高执行效率的算法,二是考虑多约束下如何选取出适合的服务组合,以便按用户要求更

(下转第 117 页)

够准确定位水印嵌入的起始位置进而提取水印信息。

(2) 本文压缩域水印算法对 MP3 压缩攻击具有较高的鲁棒性。

(3) 算法能在保证听觉效果的同时抵抗多种常见的音频攻击,具有较高的鲁棒性。

(4) 算法对同步攻击具有一定的鲁棒性。

另外,该 MP3 水印算法应更好地利用音频信号的重要特征并结合 HAS 来提高抵抗去同步攻击的能力,这将在以后做进一步的深入研究。

参 考 文 献

[1] Cox I, Killian J, Leighton T, et al. Secure spread spectrum watermarking for multimedia [J]. IEEE Trans. Image Process, 1997, 6(12): 1673-1687

[2] 李伟,袁一群,李晓强,等. 数字音频水印技术综述[J]. 通信学报,2005,26(2):100-111

[3] 孙圣和,陆哲明,牛夏牧. 数字水印技术及应用[M]. 北京:科学出版社,2004

[4] Bassia P, Pitas L, Nikolaidis N. Robust audio watermarking in the time domain [J]. IEEE Transactions on Multimedia, 2001, 3 (2): 232-242

[5] Cveji C, Keskinarkaus A, Seppanen T. Audio watermarking using m-sequences and temporal masking [A] // Proceedings of the 7th IEEE Workshop on Applications of Signal Processing to Audio and Acoustics[C]. New York, NY, 2001: 227-230

[6] Lie W N, Chang L C. Robust and high-quality time-domain audio watermarking subject to psychoacoustic masking [J]. IEEE International Symposium on Circuits and Systems, 2001, 2: 45-48

[7] 王向阳,杨红颖,赵红,等. 自适应小波域数字音频水印嵌入算法研究[J]. 小型微型计算机系统,2006,27(7):1353-1357

[8] Wang C T, Chen T S, Chao W H. A new audio watermarking based on modified discrete cosine transform of MPEG/Audio layer ÓM[C] // Proceedings of the 2004 IEEE International Conference on Networking, Sensing & Control. Taipei, 2004: 984

[9] Koukopoulos D, Stamatiou Y. A watermarking scheme for MP3 audio files [J/OL]. Int J Signal Process, 2006, 2/3: 206. http://www. waset. org/ijsp/v2/v223230. pdf, 2006205201

[10] 刘亚多,李伟,李晓强,等. 压缩域鲁棒音乐指纹算法研究[J]. 电子学报,2010,38(5):1172-1176

[11] Chang T Y. Research and implementation of MP3 encoding algorithm [D]. Taiwan: National Chiao Tung University, 2002

[12] Cox I J, Miller M L, Bloom J A. Digital Watermarking [M]. Morgan Kaufman Publishers, 2002

[13] 吴绍权,黄继武,黄达人. 基于小波变换的自同步音频水印算法[J]. 计算机学报,2004,27(3):366-370

[14] 马天笑,王向阳,杨红颖. 一种基于均值量化的抗去同步攻击数字水印算法[J]. 计算机科学,2009,36(4):257-259

[15] Wang Xiang-yang, Zhao Hong. A novel synchronization invariant audio watermarking scheme based on DWT and DCT [J]. IEEE Trans Signal Processing, 2006, 54(12): 4835-4840

[16] 高海英,钮心忻,杨义先. 基于量化的小波域自同步数字音频水印算法[J]. 北京邮电大学学报,2005,28(6):103-105

[17] 彭宏,王珣,王卫星,等. 基于音频特征的多小波域水印算法[J]. 计算机研究与发展,2010,47(2):216-222

[18] 鲍德旺,杨红颖,祁薇,等. 基于音频特征的抗去同步攻击数字水印算法[J]. 中国图象图形学报,2009,14(12):2620-2622

(上接第 105 页)

高效地完成服务选择工作。

参 考 文 献

[1] Zeng L, Benatallah B, Ngu A H H, et al. Qos-aware middleware for Web services compositon[J]. IEEE TransSoftwEng, 2004, 30 (5): 311-327

[2] Ran S. A modle for Web services discovery with QoS[J]. ACM SIGEcom Exchanges, 2003, 4(1): 1-10

[3] Liu Y, Ngu A, Zeng L. QoS computation and policing in dynamic Web service selection[C] // Proc. Of the WWW2004. New York: ACM Press, 2004: 66-73

[4] Zeng L, Benatallah B, Dumas M, et al. Quality driven Web services composition [C] // Proc. of the 12thInt'l Conf. on WWW2003. 2003: 411-421

[5] Tian M, Gramm A, Ritter H, et al. Efficient selection and monitoring of QoS-aware web services with the WS-QoS Framework [A] // IEEE/WIC/ACM International Conference on Web Intelligence(WI'04)[C]. 2004

[6] BPEL4WS. Business Process Execution Language for Web Services [EB/OL]. http://www-128. ibm. com/developerworks/library/specification/ws-bpel/, February 2005

[7] http://wiki. huihoo. com/index. php? title=Workflow

[8] Schuster H, Georgakopoulos D, Cichocki A, et al. Modeling and composing service-based and reference process-based multi-enterprise processes [A] // Proceeding of 12th International Conference on Advanced Information Systems Engineering (CaiSE) [C]. Stockholm, Sweden, Springer Verlag, June2002

[9] Soydan B A, Munindar S P. A DAML-based repository for QoS-aware semantic Web service selection[A] // Proceedings of the IEEE International Conference on Web Services (ICWS' 04) [C]. 2004

[10] Zhou Chen, Chai Liang-tien, Lee B-S. DAML-QoS ontology for web services [A] // IEEE International Conference on Web Services(ICWS' 04) [C]. 2004

[11] Zhang Liang-jie, Li Bing, Chao Tian, et al. QoS-aware middleware for web services-based business process composition[A] // IEEE[C]. 2003: 4057-4064

[12] Canfora G, Penta M D, Esposito R, et al. A lightweight approach for QoS-aware service composition [A] // ICSOC[C]. 2004

[13] Jaegar M C, Muhl G, Golze S. QoS-Aware Composition of Web Service: A Look at Selection Algorithms [A] // ICWS2005[C]. Orlando, Florida USA, 2005

[14] Benatallah B, Dumas M, Sheng Q Z, et al. Declarative composition and peer-to-peer provisioning of dynamic Web services[A] // Proc. of the 18th Int'l Conf. on Data Engineering. San Jose: IEEE Computer Society[C]. 2002: 297-308

[15] 赵俊峰,谢冰,张路,等. 一种支持领域特性的 Web 服务组装方法[J]. 计算机学报,2005,28(4):731-738

[16] Kennedy J, Eberhart R C. Particle Swarm Optimization [A] // Proc. of the IEEE Conf. on Nerual Networks[C]. Perth: IEEE Press, 1995: 1942-1948

[17] 刘书雷,刘云翔,张帆,等. 一种服务聚合中的 QoS 全局最优服务动态选择算法[J]. 软件学报,2007,18(3):646-656

[18] Han Jia-wei, Kamber M, et al. Data Mining Concepts and Techniques[M]. 北京:机械工业出版社,2001: 149-151