ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.17, No.11, November 2006, pp.2341−2351
DOI: 10.1360/jos172341

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

服务质量感知的网格工作流调度*

王 勇[1+], 胡春明[2], 杜宗霞[2]

[1](北京工业大学 计算机学院,北京 100022)

[2](北京航空航天大学 计算机学院,北京 100083)

QoS-Awared Grid Workflow Schedule

WANG Yong[1+], HU Chun-Ming[2], DU Zong-Xia[2]

[1](College of Computer Science and Technology, Beijing University of Technology, Beijing 100022, China)

[2](School of Computer Science and Engineering, BeiHang University, Beijing 100083, China)

+ Corresponding author: Phn: +86-10-67392994, Fax: +86-10-67391744, E-mail: wangy@bjut.edu.cn, http://www.bjut.edu.cn

Wang Y, Hu CM, Du ZX. QoS-Awared grid workflow schedule. *Journal of Software*, 2006,17(11):2341−2351.
http://www.jos.org.cn/1000-9825/17/2341.htm

Abstract: QoS allows for the selection and execution of Grid workflow to better fulfill customer's expections. Introducing of QoS can provide means for schedule of component services and make the execution of workflow satisfy the requirements of users better. The estimate algorithms of QoS and QoS-awared workflow schedule are two basic problems of workflow QoS management. In this paper, the QoS parameters of grid workflow are discussed, the corresponding QoS estimate algorithms are presented based on a Grid workflow model, and the problem of QoS-awared gird workflow schedule is proposed. Some QoS-awared schedule algorithms are discussed and a solution based on genetic algorithms is proposed.

Key words: grid; workflow; quality of service; schedule algorithm; genetic algorithm

摘 要: 在网格工作流中引入服务质量,可以使网格中的资源更好地围绕用户的要求进行组织和分配,服务质量为工作流执行过程中选择成员服务提供了依据.工作流服务质量的估算和服务质量感知的工作流调度是实现服务质量感知的网格工作流的两个关键问题.基于一种网格工作流模型讨论了网格工作流的服务质量参数体系,提出了工作流服务质量的估算算法和网格工作流调度数学模型,并提出了基于遗传算法的调度方法.仿真实验表明,该调度算法具有较好的收敛性.

关键词: 网格;工作流;服务质量;调度算法;遗传算法

中图法分类号: TP393 文献标识码: A

随着 Web 服务技术规范 WSRF(Web service resource framework)[1]的出现,网格技术和 Web 服务技术进一步相融合,以服务为基本构成元素的服务网格已经成为网格构建的主流方向之一[2,3].网格工作流作为实现网格中

* Supported by the National Natural Science Foundation of China under Grant No.90412011 (国家自然科学基金); the National High-Tech Research and Development Plan of China under Grant No.2003AA119030 (国家高技术研究发展计划(863)); the ChinaGrid Project of the Ministry of Education of China (国家教育部中国教育科研网格计划)

Received 2006-04-17; Accepted 2006-08-07

资源协同工作的一项重要技术手段,把网格中多个服务包装成一个更大粒度的服务部署在网格中,供其他服务或上层的应用访问.

由于基于网络环境的各类应用的差异性、动态性,网络应用软件的集约化不仅极大地增加了对核心功能需求的复杂度,而且对软件的可靠性、可维护性、安全性、可控性等非功能性需求也越来越高;同时,网络环境中同时存在着数量众多、功能相同或相近、服务质量等非功能特性各异的服务,根据服务质量等应用需求动态组合服务,实现应用软件的“按需服务”机制,则成为网络应用软件设计与开发的一个核心问题.在网格工作流中引入服务质量,将应用所需的服务质量作为服务选择的依据,使基于工作流的组合服务的执行和调度围绕用户所提出的服务质量参数进行,使得工作流的执行更能满足用户的要求.

我们将网格工作流的服务质量管理分为协商和执行两个阶段,其中,协商阶段是根据工作流中各个活动执行的历史信息计算工作流的服务质量值,为工作流的执行提供服务质量参数;在工作流的执行过程中,工作流执行服务需要根据用户提出的服务质量的参数值对工作流中的各个活动进行调度,把对整个工作流的服务质量要求规划到对工作流中各个成员服务的服务质量要求上.在这两个阶段中,工作流服务质量的估算算法是实现这些功能的基础,它为在协商阶段工作流服务质量值的计算和执行阶段工作流执行的规划目标的确立提供了计算方法;而在工作流的运行过程中,成员服务的调度过程是工作流执行的关键步骤.

针对服务质量感知的网格工作流中的两个关键问题:工作流服务质量的估算和服务质量感知的工作流调度,本文提出了相应的估算算法和调度算法.本文第 1 节简要介绍服务质量研究所基于的网格工作流模型.第 2 节介绍网格工作流服务质量的估算算法.第 3 节重点介绍服务质量感知的网格工作流调度算法.第 4 节分析一个工作流实例,对基于遗传算法的调度算法进行仿真.

1 网格工作流模型 GPEL(grid process execution language)简介

BPEL4WS(business process execution language for Web service)[4]是服务组合的典型描述语言,但本身也存在一些缺点,例如在成员服务的绑定方式方面存在局限、过于简单的生命周期管理功能、不支持对等交互以及语言本身过于冗余复杂等,尤其是 BPEL4WS 不能直接应用在网格工作流的构造上,例如不支持最新的网格技术规范 WSRF,对网格中的大计算量任务和大数据量交互任务没有提供支持.基于 Web 服务组合描述语言 BPEL4WS,我们提出了一种网格工作流描述语言 GPEL[5,6],其主要特点是:

(1) 支持网格技术规范 WSRF.网格工作流支持对 WSRF 标准服务的组合,网格工作流本身也可以发布为一个 WSRF 标准服务.

(2) 支持大计算量类型的任务.在计算密集型网格应用中,工作流中的一个活动通常需要多个成员服务来共同完成,而 BPEL4WS 中的一个活动通常绑定到一个服务上来实现.通过增加一种称为 mInvoke 类型的原子活动,支持工作流中的一个活动在运行时绑定到多个服务上.

(3) 支持存在大数据量交互的任务.在成员服务之间存在大数据量交互的情况下,完成工作流管理功能的工作流执行服务,可能充当工作流执行过程中的瓶颈,通过 WSRF 提供的通知机制,使得工作流中需要进行大数据交互的成员服务之间直接进行数据交换,从而避开了工作流执行服务这个逻辑上的瓶颈.

(4) BPEL4WS 采用结构化和有向图两种构造方式,引入了很多冗余,许多工作流模式可以在 BPEL4WS 中找到若干种表示[7].GPEL 去除了基于有向图的工作流构造方式,增加了一种并行类型的编程结构 parallel,以支持并行计算类型的任务.

目前,e-Science 是网格工作流的主要应用领域之一,这些服务一次运行的代价通常较大(如运行时间较长、花费较大等),因此可以忽略数据的网络传输代价和工作流的内部处理代价,假设工作流中包含的成员服务运行的代价是工作流运行代价的关键.

GPEL 包含 9 种原子类型的活动(empty,invoke,receive,reply,assign,wait,throw,terminate,mInvoke)和 6 种结构化活动(sequence,switch,pick,parallel,while,scope).对于这些活动,需要考察它们对工作流服务质量的贡献情况.

在 GPEL 的 9 种原子活动中,只有 invoke 活动和 mInvoke 活动需要调用外界服务,其他活动的功能是接收来自外界的请求(receive 活动)、对外界的请求作出响应(reply 活动)或者进行工作流内部的处理活动(empty,assign,wait,throw 和 terminate),也就是说,这些活动对于整个工作流的服务质量不会有什么贡献.在 6 种结构化活动中:switch 类型活动的语义是根据工作流执行的内部条件选择其中的一个分支来执行;pick 类型活动的语义是根据外部请求消息的类型选择其中的一个分支来执行,都是选择执行控制结构,不同的是做出选择的条件的来源不同,一个来自于工作流执行的内部数据(switch 活动),而另一个来自于外界的选择(pick 活动),两者的服务质量计算模型没有分别;scope 为包含在其中的活动提供一个共同的执行上下文(共同的补偿处理、事件处理和错误处理),本身并不提供工作流的执行控制功能,由包含在其中的子活动决定.也就是说,在 GPEL 中只考虑顺序活动、选择活动、循环活动、并行活动、scope 活动 5 种结构化活动的服务质量计算问题.

2 网格工作流服务质量的估算算法

2.1 网格工作流服务质量的计算公式

网格工作流所支持的服务质量参数取决于成员服务所支持的服务质量参数.在 Web 服务的服务质量研究中,不同的研究工作提出了不同的服务质量参数体系[8,9],这些服务质量参数体系的内容大同小异,基本上涵盖了服务质量的不同方面,包括性能、可靠性、可提供性、正确性、完整性、费用、安全等,有的是定量的,有的是定性的.我们从服务质量参数的重要性程度和易于度量的角度出发,对 Web 服务的服务质量参数进行了选择,建立了包括 5 个服务质量参数(性能、花费、可靠性、可提供性和声誉)的参数体系:

(1) 性能:用来评测服务完成请求的速度[8],用响应时间作为性能的度量指标.响应时间是指工作流或者服务响应一次请求所需要的时间,可以进一步划分为请求等待时间和请求处理时间.

(2) 花费:用来衡量工作流或者服务一次执行所需要的费用.

(3) 可靠性:指服务或者工作流在一定条件下、在特定的时间内执行所需要的功能的能力.可靠性可以用执行的成功率来衡量,也就是服务或者工作流执行成功的次数与总的执行次数的比率.

(4) 可提供性:可提供性与可靠性相关,是指服务或者工作流在一定的条件下、在特定的时间内提供所需要的功能的能力.可提供性可以用服务或者工作流提供运行的时间与总的时间的比率来衡量.

(5) 声誉:声誉是用来描述一个服务的可信度的指标,它依赖于用户对服务使用的体验.不同的用户对相同的服务可能有不同的体验,也就是说,会给予服务不同的声誉值.声誉的值采用用户使用某一服务后给予该服务的平均等级来衡量,即 $R_{ep}=\sum_{i=1}^{N} Rank_i / N$.

基于第 1 节的分析,我们逐一给出 GPEL 中各个活动的服务质量计算公式.

invoke 类型的活动的执行语义是调用网格中的一个服务,所以 invoke 类型的活动的服务质量等同于该服务的服务质量,计算公式如下:

响应时间: $T_{invoke}=T_s$;

花费: $C_{invoke}=C_s$;

可靠性: $Rel_{invoke}=Rel_s$;

可提供性: $A_{invoke}=A_s$;

声誉: $Rep_{invoke}=Rep_s$.

其中,T_{invoke},C_{invoke},Rel_{invoke},A_{invoke},Rep_{invoke},T_s,C_s,Rel_s,A_s,Rep_s 分别是 invoke 类型活动和该服务的对应的服务质量参数.

mInvoke 类型的活动在执行时会绑定到网格中多个功能相同的服务上,这些服务的服务质量参数各不相同.服务的数量由配置文件中的参数和运行时查找网格信息服务的信息来确定,假定为 N.我们给出 mInvoke 类型的活动的服务质量计算公式如下:

响应时间: $T_{mInvoke}=\mathrm{MAX}_{i=1}^{N}\{T_i\}$, T_i 为第 i 个服务的响应时间;

花费: $C_{mInvoke}=\sum_{i=1}^{N}C_i$, C_i 为第 i 个服务的花费;

可靠性: $Rel_{mInvoke}=\prod_{i=1}^{N}Rel_i$, Rel_i 为第 i 个服务的可靠性值;

可提供性: $A_{mInvoke}=\prod_{i=1}^{N}A_i$, A_i 为第 i 个服务的可提供性值;

声誉: $Rep_{mInvoke}=\sum_{i=1}^{N}Rep_i/N$, Rep_i 为第 i 个服务的声誉值.

scope 类型的活动为包含在其中的活动提供了一个共同的执行上下文,子活动可以是原子活动,也可以是结构化活动.scope 类型活动的服务质量取决于包含在其中的子活动的服务质量,其服务质量计算公式如下:

响应时间: $T_{scope}=T(a)$, $T(a)$为活动 a 的响应时间;

花费: $C_{scope}=C(a)$, $C(a)$为活动 a 的花费;

可靠性: $Rel_{scope}=Rel(a)$, $Rel(a)$为活动 a 的可靠性值;

可提供性: $A_{scope}=A(a)$, $A(a)$为活动 a 的可提供性值;

声誉: $Rep_{scope}=Rep(a)$, $Rep(a)$为活动 a 的声誉值.

顺序活动(sequence)的执行语义是顺序执行包含在该结构化活动中的子活动,这些活动可以是结构化活动,也可以是原子活动,假定包含的活动的数量为 N.顺序活动的服务质量的计算公式如下:

响应时间: $T_{sequence}=\sum_{i=1}^{N}T(a_i)$, $T(a_i)$为活动 a_i 的响应时间;

花费: $C_{sequence}=\sum_{i=1}^{N}C(a_i)$, $C(a_i)$为活动 a_i 的花费;

可靠性: $Rel_{sequence}=\prod_{i=1}^{N}Rel(a_i)$, $Rel(a_i)$为活动 a_i 的可靠性值;

可提供性: $A_{sequence}=\prod_{i=1}^{N}A(a_i)$, $A(a_i)$为活动 a_i 的可提供性值;

声誉: $Rep_{sequence}=\sum_{i=1}^{N}Rep(a_i)$, $Rep(a_i)$为活动 a_i 的声誉值.

选择活动(switch,pick)的执行语义是多个分支的选择执行,也就是从多个执行分支中选择一个分支执行.我们假定分支的数目为 N,执行活动 a_i 的概率为 p_i,并且满足约束 $\sum_{i=1}^{N}p_i=1$.选择活动的服务质量计算公式如下:

响应时间: $T_{selection}=\sum_{i=1}^{N}p_i*T(a_i)$, $T(a_i)$为活动 a_i 的响应时间;

花费: $C_{selection}=\sum_{i=1}^{N}p_i*C(a_i)$, $C(a_i)$为活动 a_i 的花费;

可靠性: $Rel_{selection}=\sum_{i=1}^{N}p_i*Rel(a_i)$, $Rel(a_i)$为活动 a_i 的可靠性值;

可提供性: $A_{selection}=\sum_{i=1}^{N}p_i*A(a_i)$, $A(a_i)$为活动 a_i 的可提供性值;

声誉: $Rep_{selection}=\sum_{i=1}^{N}p_i*Rep(a_i)$, $Rep(a_i)$为活动 a_i 的声誉值.

循环活动(while)的执行语义是在一定条件下的某一活动的重复执行.我们假定在活动 a 执行结束以后进行循环的概率为 p,很明显,活动 a 执行结束后离开循环的概率为 $1-p$.给出循环活动的服务质量计算公式如下:

响应时间: $T_{while}=T(a)/(1-p)$, $T(a)$为活动 a 的响应时间;

花费: $C_{while}=C(a)/(1-p)$, $C(a)$为活动 a 的花费;

可靠性: $Rel_{while}=(1-p)*Rel(a)/(1-p*Rel(a))$, $Rel(a)$为活动 a 的可靠性值;

可提供性: $A_{while}=(1-p)\times A(a)/(1-p\times A(a))$, $A(a)$为活动 a 的可提供性值;

声誉: $Rep_{while}=Rep(a)$, $Rep(a)$为活动 a 的声誉值.

并行活动(parallel)的执行语义是所包含的多个活动的并行执行.我们假定并行活动中包含的活动的数量为 N,给出并行活动的服务质量计算公式如下:

响应时间: $T_{parallel}=\mathrm{MAX}_{i=1}^{N}\{T(a_i)\}$, $T(a_i)$为活动 a_i 的响应时间;

花费: $C_{parallel}=\sum_{i=1}^{N}C(a_i)$, $C(a_i)$为活动 a_i 的花费;

可靠性: $Rel_{parallel}=\prod_{i=1}^{N}Rel(a_i)$, $Rel(a_i)$为活动 a_i 的可靠性值;

可提供性: $A_{parallel}=\prod_{i=1}^{N}A(a_i)$, $A(a_i)$为活动 a_i 的可提供性值;

声誉: $Rep_{parallel}=\sum_{i=1}^{N}Rep(a_i)/N$, $Rep(a_i)$为活动 a_i 的声誉值.

需要指出的是,在选择活动和循环活动的服务质量模型中涉及到了转移概率.转移概率可以在工作流定义时由工作流的设计人员指出,但是,工作流定义人员在定义工作流时,往往很难准确地给出这些转移概率的值,所以,这种方法一般适用于工作流初次运行的时候.在工作流运行一定时间以后,转移概率通过工作流执行日志计算相应的转移频率,并通过转移频率代替转移概率来获得.工作流执行的次数越多,转移概率就越准确.

2.2 网格工作流服务质量的估算算法

基于 GPEL 中定义的活动的服务质量计算模型,下面我们讨论网格工作流的服务质量估算算法.由于 GPEL 提供了一种结构化工作流描述语言,在算法执行的过程中需要对网格工作流定义进行结构分析.下面,我们给出根据工作流中的成员服务的响应时间求解网格工作流响应时间的算法如下:

getResponseTime(*activity a*) //因为整个工作流通常用一个结构化活动来表示,所以,此算法可以用来求解一个活动的响应时间,也可以用来求解整个工作流的响应时间

{

responseTime=0; //赋初值

获得活动的类型;

如果活动是 invoke 类型的活动

responseTime=*getTimeOfSerivce*(*s*); //invoke 活动包含的服务的响应时间可以通过查询网格信息服务获得

如果是 mInvoke 类型的活动

responseTime=Max{*getTimeOfSerivce*(s_1),…,*getTimeOfSerivce*(s_N)}; //服务的响应时间为所包含的服务的响应时间的最大者

如果是 scope 类型的活动

获得直接子活动 *a*;

responseTime=*getResponseTime*(*a*); //取决于子活动的响应时间

如果是顺序类型的活动

获得所有的直接子活动 $a_1,a_2,\ldots,a_N$;

responseTime= $\sum${*getResponseTime*(a_1),..., *getResponseTime*(a_N)}; //递归调用各个直接子活动求解响应时间,并累加

如果是选择类型的活动

获得所有的直接子活动 $a_1,a_2,\ldots,a_N$ 以及相应的转移概率 $p_1,p_2,\ldots,p_N$;

$responseTime=\sum\{p_1\times getResponseTime(a_1),...,p_N\times getResponseTime(a_N)\}$; //递归调用各个直接子活动求解响应时间

如果是循环类型的活动

获得循环包含的活动 a'和循环执行概率 p;

$responseTime=getResponseTime(a')/(1-p)$; //递归调用活动 a'的响应时间

如果是并行类型的活动

获得所有的直接子活动 $a_1,a_2,\ldots,a_N$;

$responseTime=\mathrm{Max}\{getResponseTime(a_1),\ldots,getResponseTime(a_N)\}$; //递归调用各个直接子活动求解响应时间

返回活动 a 的响应时间 $responseTime$; //其他情况下,活动 a 的响应时间为 0

} //算法结束

该算法的复杂性与工作流中活动的嵌套层次有关,假设工作流活动的最大嵌套层次为 l,则该算法的复杂性为 $O(l)$.花费、可靠性、可提供性、声誉的求解算法与响应时间的求解算法类似,这里不再重复.

工作流的服务质量估算算法是实现工作流服务质量管理的基础,该算法可以用于在已知工作流中各个成员服务的服务质量参数时,求解整个工作流的服务质量,也可以在已知整个工作流服务质量参数时,用于求解其中各个成员服务的服务质量参数值,并把这些值作为工作流执行中选择成员服务的依据.

3 服务质量感知的网格工作流调度算法

3.1 分 析

网格工作流的执行调度过程,就是控制工作流中各个成员服务的执行次序,并从网格中存在的大量候选服务中选择合适的服务进行执行的过程,其核心功能就是服务的选择.服务质量为工作流执行过程中选择成员服务提供了一种依据,服务质量感知的工作流调度把对于整个工作流的执行要求规划到各个成员服务,并作为服务的非功能特性方面的要求.

要对工作流中的成员服务进行调度,首先需要建立工作流的总的服务质量要求与所包含的成员服务的服务质量要求之间的关系.为了方便,我们假定服务一次运行的代价通常比较大(体现在运行时间长、花费比较大等方面),假设工作流中成员服务的执行代价是整个工作流执行的代价的关键所在.基于这样的假设,第 3 节提出了一种网格工作流服务质量的估算算法,解决了在已知成员服务的服务质量参数的情况下,计算相应的工作流服务质量参数的问题,我们利用该算法可以建立工作流的服务质量要求与成员服务的服务质量要求之间的关系.因为该算法最终要递归到对网格中服务的相应服务质量参数的求解上来,所以在执行该算法时,每需要一个服务的相应的服务质量参数值,就以未知变量来替代,这样,在算法执行结束后,就可以建立起工作流的服务质量参数与成员服务的服务质量参数之间的关系.

我们假定一个网格工作流中包含了 n 个成员服务,就可以得到如下的时间 t、花费 c、可靠性 rel、可提供性 a 和声誉 rep 的关系:

$$\begin{cases} t=t(t_1,t_2,...,t_n) \\ c=c(c_1,c_2,...,c_n) \\ rel=rel(rel_1,rel_2,...,rel_n) \\ a=a(a_1,a_2,...,a_n) \\ rep=rep(rep_1,rep_2,...,rep_n) \end{cases} \tag{1}$$

其中:t_i,c_i,rel_i,a_i,rep_i 分别是第 i 个服务的响应时间、花费、可靠性、可提供性和声誉参数值;t,c,rel,a,rep 分别是

加和、连乘、Max 等运算的复合.

假定用户对工作流的总的执行要求如下:$t<T,c<C,rel>REL,a>A,rep>REP$(考虑到最终用户面对的是整个工作流的应用,并不关心应用的实际流程,也就是工作流的步骤,在这里,我们主要考虑对整个工作流的执行要求,对工作流中某一个服务的执行要求在下文中有讨论),结合上述的工作流与成员服务的服务质量参数之间的关系(式(1)),可以得到如下的不等式组:

$$\begin{cases} t = t(t_1, t_2, ..., t_n) < T \\ c = c(c_1, c_2, ..., c_n) < C \\ rel = rel(rel_1, rel_2, ..., rel_n) > REL \\ a = a(a_1, a_2, ..., a_n) > A \\ rep = rep(rep_1, rep_2, ..., rep_n) > REP \end{cases} \tag{2}$$

其中,T,C,REL,A,REP 为常数.

每一个成员服务在网格中都存在一定数量的候选服务,我们记第 i 个成员服务的候选服务的数量为 m_i,则工作流对成员服务的调度过程就是从所有成员服务的候选服务中选择合适的服务,使得能够满足式(2)所列出的执行要求.这样,服务质量感知的工作流调度机制就转变为式(2)的组合优化问题.

3.2 调度算法分析与设计

所有成员服务的候选服务的服务质量参数都注册在一个称为网格信息服务的信息表中,通过网格信息服务提供的查询接口,我们可以获得所有成员服务的服务质量信息.我们假定网格中的服务能够把自己承诺提供的服务质量及时地更新到网格信息服务中,也就是说,通过查询信息服务获得的网格服务的服务质量信息能够反映网格中服务的当前状况.工作流的调度过程就是基于从网格信息服务获得服务的服务质量信息进行的.

穷举法是一个很显然的解决问题的方法,从每一个成员服务的候选服务中选择一个服务进行逐个匹配,直到找到合适的匹配组合,使得能够满足式(2)的要求.假定第 i 个成员服务有 m_i 个候选服务,则共有 $\prod_{i=1}^{n} m_i$ 种匹配的可能,而 $(\mathrm{MIN}(m_1,...,m_i,...,m_n))^n < \prod_{i=1}^{n} m_i < (\mathrm{MAX}(m_1,...,m_i,...,m_n))^n$,所以,穷举法的时间复杂性是指数级的.也就是说,穷举法的时间复杂性是随成员服务的数量呈指数增长的.

穷举法的适用程度取决于工作流中包含的成员服务的数量,如果成员服务的数量很多,则穷举法是没有实用价值的;但是,如果成员服务的数量不多,则穷举法是一个简单而又实用的方法.在网格的主要应用领域 e-Science 中,复杂的流程并不多见.而且,在网格发展的比较长的一段时间内,网格中服务的数量也有待于进一步的成长,所以穷举法还是一种非常有效的方法.

随着工作流中所包含服务的数量的增加和网格中候选服务的增加,穷举法搜索的空间急剧扩大,采用穷举法已经很难或者不可能得到合适的解.这类非线性组合优化问题一般是 NP 难问题,对于这类复杂问题,人们已经认识到遗传算法是寻求满意解的最佳工具之一.遗传算法是一种最优化的技巧,其基本原理是仿效生物界中的“物竞天择、适者生存”的演化法则.遗传算法的做法是把问题参数编码为染色体,再利用迭代的方式进行选择、交配以及变异等运算来交换种群中染色体的信息,最终生成符合优化目标的染色体.

(1) 染色体编码

在遗传算法运算之前,需要针对问题设计染色体,包括基因字串的长度以及基因代表的含义,即对要搜索空间的可行解以编码的形式来呈现.一般的编码方式采用二进制编码,此外也有整数、实数、文字等编码方式.

我们采用二进制的编码方式,对于具有 n 个成员服务的工作流,染色体划分成 n 段,其中每一个分段为相应成员服务的候选服务预留位置,如××××|×××××|×××是具有 3 个成员服务的编码,而其中每一个分段为相应成员服务的候选服务的编码,如 1101|00111|101 代表第 1 个成员服务的第 13 个候选服务、第 2 个成员服务的第 7 个候选服务和第 3 个成员服务的第 5 个候选服务的一个编码.

(2) 种群初始化

在完成染色体编码以后,必须产生一个初始种群作为起始解,所以首先需要决定初始化种群的数目.初始化种群的数目一般根据经验得到,如果初始化种群的数目太大,则可能会消耗过多的计算时间,但是如果太小,则可能因难以达到预期的效果而导致过早收敛.

我们在种群初始化时采用随机方式产生,一般情况下种群的数量视搜索空间的大小($\prod_{i=1}^{n} m_i$)而确定,其取值在 10~160 之间浮动.

(3) 适应度函数的设定

我们首先把式(2)转变为一个单目标的组合优化问题,由式(2)可得:

$$\begin{cases} 1-t(t_1,t_2,...,t_n)/T>0 \\ 1-c(c_1,c_2,...,c_n)/C>0 \\ rel(rel_1,rel_2,...,rel_n)/REL-1>0 \\ a=a(a_1,a_2,...,a_n)/A-1>0 \\ rep(rep_1,rep_2,...,rep_n)/REP-1>0 \end{cases} \tag{3}$$

由式(3)可以得到优化目标函数:

$$\max obj=rel(rel_1,rel_2,...,rel_n)/REL+a(a_1,a_2,...,a_n)/A+rep(rep_1,rep_2,...,rep_n)/REP- t(t_1,t_2,...,t_n)/T-c(c_1,c_2,...,c_n)/C \tag{4}$$

一旦第 i 个成员服务的候选服务 j 确定以后,这里的 rel_i,a_i,rep_i,t_i,c_i 可以通过查表获得,所以可以看作成员服务 i 和候选服务 j 的函数.

在把目标函数(式(4))转化为适应度函数的时候,并不能保证目标函数处处都取非负值,为了不错过一些边缘情况($t(t_1,t_2,\ldots,t_n)/T\approx1$,$c(c_1,c_2,\ldots,c_n)/C\approx1$),我们取如下的适应度函数:

$$f=\begin{cases} 2+obj, & 2+obj>0 \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

(4) 终止条件设定

严格地讲,遗传算法的迭代终止条件目前尚无定论.在许多组合优化问题中,适应度最大值并不清楚,其本身就是搜索的对象,因此终止条件很难确定.

我们采用式(2)作为迭代的终止条件,也就是说,如果发现合适的染色体解码后能够满足式(2)的要求,则终止迭代过程.如果满意解并不存在,而搜索空间又非常庞大,在这种情况下,若发现群体中个体的进化已经趋于稳定状态,换句话说,如果发现群体中一定比例的个体已经是同一个体,则终止算法的迭代过程.

(5) 选　择

选择操作是从种群中选择优胜的个体,并淘汰劣质的个体.选择的目的是把优化的个体直接遗传到下一代,或者通过配对交叉产生新的个体再遗传到下一代.选择是建立在群体中个体的适应度评估的基础上的.常用的选择算子包括适应度比例方法、最佳个体保留方法、期望值方法、排序选择方法等.

我们选择最常用的适应度比例方法来对种群中的个体进行选择.

(6) 交　叉

交叉算子是遗传算法中起核心作用的遗传操作.所谓交叉是指两个父代个体的部分结构加以替换重组而生成新个体的操作.对于二进制编码来说,常用的交叉算子包括一点交叉、二点交叉、多点交叉、一致交叉等.

我们选用二点交叉算子,并选择交叉概率为 0.5.

(7) 变　异

变异算子的基本内容是对群体中个体串的某些基因座的基因值作变动.就字符集为{0,1}的二进制码串来说,编译操作就是把基因座上的某些基因值取反,即 1→0 或者 0→1.一般来说,变异算子操作的基本步骤如下:

1) 在群体的所有个体的码串范围内随机确定基因座;

2) 以事先确定的变异概率对这些基因座的基因值进行变异.

我们选用基本变异算子,并选定变异概率为 0.001.

4 实例分析

我们给出如图 1 所示的工作流实例,对基于遗传算法的调度算法进行仿真实验.该工作流实例共包含 7 个 invoke 类型的活动、3 个顺序活动、1 个选择活动、1 个并行活动和 1 个循环活动.该假想实例能够比较全面地反映 GPEL 的描述能力,具有一定的代表性.

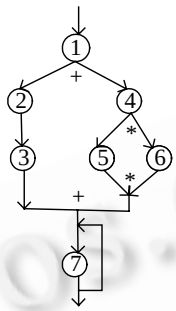

Fig.1 A workflow example

图 1 一个工作流的例子

假定其中每一个服务分别有 4,16,8,4,4,16,8 个候选服务,也就是说,搜索空间为 $4\times16\times8\times4\times4\times16\times8=1048576$.采用第 2.2 节中提出的方法可以建立式(6)所示的非线性不等式组(我们假定用户对工作流的服务质量要求为 $t<600$,$c<500$,$rel>0.8$,$a>0.7$,$rep>80$,且通过工作流的执行历史求得选择活动两个分支的转移概率分别为 0.4 和 0.6,循环活动的循环执行概率为 0.8):

$$\begin{cases} t_1+0.4\times t_2+0.4\times t_3+0.6\times t_4+0.6\times \mathrm{MAX}(t_5,t_6)+5\times t_7<600 \\ c_1+0.4\times c_2+0.4\times c_3+0.6\times c_4+0.6\times c_5+0.6\times c_6+5\times t_7<500 \\ rel_1\times(0.4\times rel_2\times rel_3+0.6\times rel_4\times rel_5\times rel_6)\times 0.8\times rel_7/(1-0.2\times rel_7)>0.8 \\ a_1\times(0.4\times a_2\times a_3+0.6\times a_4\times a_5\times a_6)\times 0.8\times a_7/(1-0.2\times a_7)>0.7 \\ (rep_1+(0.4\times(rep_2+rep_3)/2+0.6\times(rep_4+(rep_5+rep_6)/2)/)+rep_7)/3>80 \end{cases} \tag{6}$$

根据第 3.2 节中对基于遗传算法的调度算法的分析,可以建立该工作流调度问题的适应度函数为(在 Matlab 中的遗传算法工具包求解的是最小化问题,与第 3.2 节给出的适应度函数是等价的)

$$\min f=I_1+I_2+I_3-I_4-I_5,$$

其中:

$$I_1=(t_1+0.4\times t_2+0.4\times t_3+0.6\times t_4+0.6\times \mathrm{MAX}(t_5,t_6)+5\times t_7)/600,$$

$$I_2=(c_1+0.4\times c_2+0.4\times c_3+0.6\times c_4+0.6\times c_5+0.6\times c_6)+5\times c_7)/500,$$

$$I_3=(rep_1+(0.4\times(rep_2+rep_3)/2+0.6\times(rep_4+(rep_5+rep_6)/2)/)+rep_7)/3/80,$$

$$I_4=rel_1\times(0.4\times rel_2\times rel_3+0.6\times rel_4\times rel_5\times rel_6)\times 0.8\times rel_7/(1-0.2\times rel_7)/0.8,$$

$$I_5=a_1\times(0.4\times a_2\times a_3+0.6\times a_4\times a_5\times a_6)\times 0.8\times a_7/(1-0.2\times a_7)/0.7.$$

服务质量的格式为 $Q_{i,j}=(r_{i,j},a_{i,j},t_{i,j},c_{i,j})$的形式,分别表示该服务的可靠性值、可提供性值、响应时间和花费,参数值随机生成.

通过仿真,可以看到适应度函数的收敛情况,如图 2 所示,其中,仿真算法的终止条件为种群中的个体趋于稳定状态,也就是说,种群中个体的平均适应度值与最佳个体的适应度相同.从图中可以看出:在算法运行前期,最佳适应度函数值与最坏适应度函数值之间的差距较大,平均适应度函数值有较大的变化;随着算法的执行,最佳适应度函数值与最坏适应度函数值之间的差距迅速变小,平均适应度函数值的变化趋于平稳;在不到 20 代时,最佳适应度函数值与最坏适应度函数值之间的距离趋于 0,而平均适应度函数值稳定在–0.27 左右.

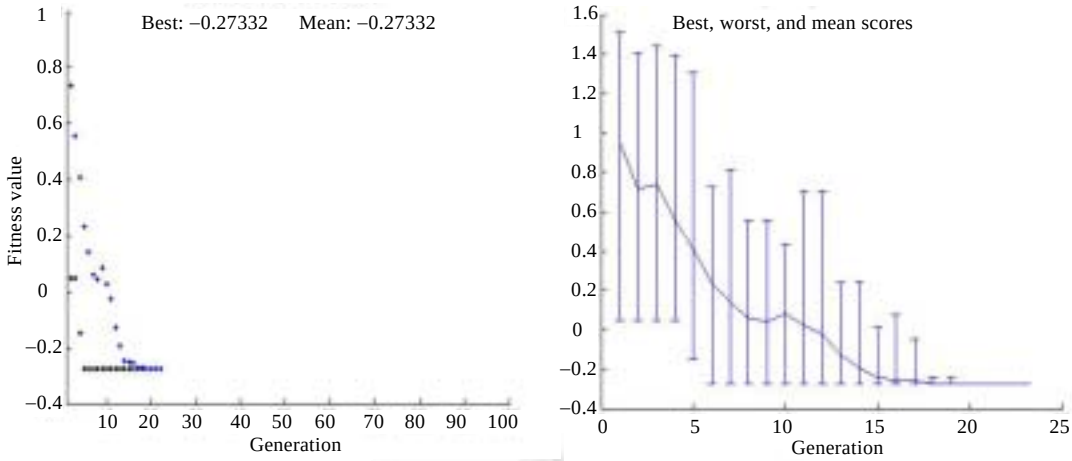

Fig.2 The convergence of fitness function

图 2 适应度函数的收敛

该仿真算法最终选择的候选服务序列分别为 3,9,5,4,4,10,3,计算可得工作流的 t=526<600,c=440<500, rel=0.85>0.8,a=0.82>0.7,rep=89>80,满足对工作流的服务质量要求.

需要说明的是,工作流调度过程中,并不需要找到满足式(2)的最优解,也就是说,如果采用第 3.2 节给出的寻找满意解的迭代终止条件,算法的执行时间还要大大缩短.为了便于实现仿真算法,在该实例中给出的成员服务的候选服务的个数都是 2^n 的形式,这样使得编码过程中不会出现不合法的染色体,在算法的实际实现过程中,只需加入染色体合法性检查环节即可.

此外,我们主要考虑对工作流的整体执行要求,如果用户对工作流中某一个活动有特殊的执行要求,存在两种处理的方式:一种是把这种执行要求作为一项约束加入到优化函数中;另一种是在算法的实现过程中把该要求作为染色体合法性的一个指标,在合法性检查中确保该活动的服务质量要求得到满足.

5 相关工作

服务质量最早出现于网格计算和实时计算系统中.早期的工作流服务质量管理方面的工作从方便工作流管理的角度入手,所支持的服务质量体系也仅仅包含时间一个维度[10].

自从服务的概念出现以后,服务已经成为构成工作流的基本要素.在服务工作流服务质量方面的研究工作中,具有代表性的是 CrossFlow[11,12]和文献[10]的工作.在 CrossFlow 中考虑了时间和花费两项服务质量参数,通过工作流执行的日志信息构建了一个连续时间的马尔可夫链模型,以对工作流服务质量参数进行估算,但是,由于连续时间马尔可夫链并不支持并行类型的活动,所以,CrossFlow 对并行活动的处理显得非常繁琐.文献[10]提供了对于时间、花费、可靠性等服务质量参数的支持,并为每一个服务质量维度提供了一种计算方法,但是并不能直接应用在网格工作流中,主要的原因是所基于的工作流模型不同,而且在网格工作流中,对于服务质量参数体系的建立有着特殊的要求.

文献[10]中给出了工作流服务质量的计算方法,但并没有给出相应的调度算法.在 CrossFlow 中考虑了时间和花费两项服务质量参数,把时间和花费合成为一项代价指标,并假定成员服务可以为工作流中的若干个活动提供实现,把工作流的调度问题转化为求解一个最小代价覆盖问题,给出了该问题的穷举搜索算法.首先,本文提出的服务质量感知的调度算法基于多个服务质量参数(响应时间、花费、可靠性、可提供性、声誉),并可以扩展到多个服务质量参数(只要该服务参数是可以度量的);其次,本文不同于 CrossFlow 提出的一个成员服务可以为工作流中的多个活动提供实现的假设,而认为工作流中的一个活动由网格中的一个服务(其中的一个操作)实现,并单独计算代价,并给出了求解该调度问题的穷举法和遗传算法.

6 结束语

本文在一种网格工作流描述语言GPEL的基础上建立了网格工作流服务质量的参数体系和计算模型,并给出了相应的计算算法.与文献[10]的工作相比,本文提出的服务质量参数体系和算法更加适应网格的环境和应用的要求.最后,本文提出了服务质量感知的网格工作流调度问题的数学模型,给出解决此问题的穷举法和遗传算法,并给出了一个工作流实例,对基于遗传算法的调度算法进行了仿真和分析.

下一步的工作包括两个方面:一方面是进一步考察网格工作流的服务质量要求,建立更为完善、更加符合网格环境要求的服务质量参数体系;另一方面是把本文提出的算法应用到网格工作流的服务质量管理中去,并在实际应用中得到验证和改进.

References:

[1] Czajkowski K. The WS-resource framework. 2004. http://www.globus.org/wsrf/specs/ws-wsrf.pdf

[2] Foster I, Kesselman C, Nick J, Tuecke S. The physiology of the grid: An open grid services architecture for distributed systems integration. 2002. http://www.globus.org/research/papers/ogsa.pdf

[3] Hu CM, Huai JP, Sun HL. WebSASE4G: A Web services-based grid architecture and its supporting environment. Journal of Software, 2004,15(7):1064−1073 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/15/1064.htm

[4] Curbera F, Goland Y, Klein J, Leymann F, Roller D, Thatte S, Weerawarana S. Business process execution language for Web services. 2002. http://msdn.microsoft.com/Webservices/default.asp?pull=/library/en-us/dnbiz2k2/html/bpel1-0.asp

[5] Wang Y, Hu CM, Huai JP. A new grid workflow description language. In: Proc. of the 2005 IEEE Int'l Conf. on Services Computing, Vol 2. Orlando: IEEE Computer Society, 2005. 257−260.

[6] Wang Y, Huai JP. Comparative analysis of BPEL4WS and a grid workflow language called GPEL. In: Proc. of the 2005 IEEE Int'l Conf. on Services Computing, Vol 2. Orlando: IEEE Computer Society, 2005. 253−254.

[7] Wohed P, van der Aalst WMP, Dumas M, ter Hofstede AHM. Pattern based analysis of BPEL4WS. 2004. http://www.citi.qut.edu.au/about/research_pubs/technical/pattern_based_analysis.pdf

[8] Ran SP. A model for Web services discovery with QoS. ACM SIGecom Exchanges, 2003,4(1):1−10.

[9] Mani A, Nagarajan A. Understanding quality of service for Web services. IBM, 2002. http://www-106.ibm.com/developerworks/library/ws-quality.html

[10] Cardoso AJS. Quality of service and semantic composition of workflows [Ph.D. Thesis]. Georgia: University of Georgia, 2002.

[11] Grefen P, Aberer K, Hoffner Y, Ludwig H. CrossFlow: Cross-organizational workflow management in dynamic virtual enterprises. Int'l Journal of Computer Systems Science & Engineering, 2000,15(5):227−290.

[12] Klingemann J, Wäsch J, Aberer K. Deriving service models in cross-organizational workflows. In: Proc. of the RIDE—Information Technology for Virtual Enterprises (RIDE-VE'99). Sydney, 1999. 100−107. http://ieeexplore.ieee.org/iel4/6091/16312/00758620.pdf?arnumber=758620

附中文参考文献:

[3] 胡春明,怀进鹏,孙海龙.基于Web服务的网格体系结构及其支撑环境研究.软件学报,2004,15(7):1064−1073. http://www.jos.org.cn/1000-9825/15/1064.htm

王勇(1974－),男,山东潍坊人,博士,讲师,主要研究领域为服务组合,工作流,网格计算.

杜宗霞(1975－),女,博士,讲师,主要研究领域为服务组合,工作流.

胡春明(1977－),男,博士,讲师,主要研究领域为分布式计算,中间件,网格计算.