ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.18, No.8, August 2007, pp.1870−1882
DOI: 10.1360/jos181870

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

一种基于业务生成图的Web服务工作流构造方法*

胡春华[1,2+], 吴 敏[1], 刘国平[1,3], 徐德智[1]

[1](中南大学 信息科学与工程学院,湖南 长沙 410083)

[2](湖南商学院 计算机与电子工程系,湖南 长沙 410205)

[3](School of Electronics, University of Glamorgan, Pontypridd CF37 1DL, UK)

An Approach to Constructing Web Service Workflow Based on Business Spanning Graph

HU Chun-Hua[1,2+], WU Min[1], LIU Guo-Ping[1,3], XU De-Zhi[1]

[1](School of Information Science and Engineering, Central South University, Changsha 410083, China)

[2](Department of Computer and Electronic Engineering, Hu'nan Business College, Changsha 410205, China)

[3](School of Electronics, University of Glamorgan, Pontypridd CF37 1DL, UK)

+ Corresponding author: Phn: +86-731-8832649, Fax: +86-731-8689238, E-mail: huchunhua777@163.com, http://cee.hnbc.com.cn

Hu CH, Wu M, Liu GP, Xu DZ. An approach to constructing Web service workflow based on business spanning graph. *Journal of Software*, 2007,18(8):1870−1882. http://www.jos.org.cn/1000-9825/18/1870.htm

Abstract: Based on the fact that Web service dynamically changes and rapidly increases in the Internet, a user-oriented service workflow constructing model is proposed. The same or similar function services are accumulated into a kind of service set in this model, which is organized by the spanning tree, and the business spanning graph is formed according to workflow's business logic relation. At the same time, on the basis of redefining the position, velocity, addition, subtraction and multiplication of particle swarm algorithm, combining with the cross and mutation operations in genetic algorithm, the QoS (quality of service)scheduling algorithm based on hybrid particle swarm optimization algorithm (HPSOA) is designed to satisfy different multi-QoS demands when Web service rapidly increases. Experimental results demonstrate that the constructing model could effectively shield the physical varieties and differences of Web service, and preferably combine the Web service resource in the Internet. It is suitable for the application demands in the virtual computing environment.

Key words: Web service; workflow; business spanning graph; QoS (quality of service) scheduling; HPSOA (hybrid particle swarm optimization algorithm)

摘 要: 针对互联网中Web服务具有动态变化且迅速增长的特点,提出了一种面向用户需求的服务工作流构造模型.该模型将功能相同或相似的服务聚集成一类服务集合,每类服务集合采用生成树的方式组织,并依据工作流的业务逻辑关系形成业务生成图;同时,在重定义粒子群算法的位置、速度、加/减法和乘法的基础上,结合遗传算法中的

* Supported by the National Natural Science Foundation of China under Grant No.60674016 (国家自然科学基金); the National High-Tech Research and Development Plan of China under Grant No.2006AA04Z172 (国家高技术研究发展计划(863)); the National Science Fund for Distinguished Young Scholars of China under Grant No.60425310 (国家杰出青年科学基金); the Natural Science Foundation of Hu'nan Province of China under Grant No.05JJ40118 (湖南省自然科学基金)

Received 2007-02-28; Accepted 2007-04-26

交叉、变异操作,设计了基于混合粒子群的QoS(quality of service)调度方法,保证在可选服务不断增长时能够满足用户的个性化需求.实验结果表明,该模型能够有效地屏蔽组成工作流的Web服务物理上的变化与差异,较好地组合了Internet中的Web服务资源,适合于虚拟计算环境的应用要求.

关键词: Web服务;工作流;业务生成图;QoS(quality of service)调度;混合粒子群算法

中图法分类号: TP311 **文献标识码:** A

Web服务技术通过采用WSDL,UDDI和SOAP等基于XML的标准和协议,解决了异构分布式计算以及代码与数据重用等问题,具有高度的互操作性、跨平台性和松耦合的特点[1].但在实际应用中,单项Web服务一般仅提供某些单一功能,通常无法满足复杂的应用需求.因此,将单一Web服务组合成功能更强大的服务以满足不同用户的应用需求已成为一个新的研究热点.

Web服务工作流(又称为服务组合)是实现互联网中服务资源协同工作的一个重要手段,将多项服务组装成一个具有更大粒度的增值服务或系统,满足上层的应用访问或其他需要[2].服务工作流是一种半自动服务组合方式,首先要由用户根据需求建立适合具体应用需求的工作流业务逻辑模型.该模型由多个服务节点组成,各服务节点包含具体的功能需求描述[3].开放的互联网环境中同时存在着数量众多、功能相同或相近、服务质量(quality of service,简称QoS)等非功能特性各异的服务.如何从这些异常丰富且动态变化的Web服务中选择满足各服务节点功能需求的具体服务,形成一个切实可行的工作流实例来完成用户的个性化需求,成为服务工作流研究的关键问题.本文称其为服务工作流路径选择问题.对该问题的研究主要包括两个方面:1) 从功能需求的角度在不断变化的Web服务中找出符合要求的Web服务集;2) 功能相同或相似的Web服务其服务质量差异很大,需要从符合要求的服务集中找出满足用户QoS需求的服务实例.

目前,使用工作流作为一个分布式活动的协调引擎,或作为一个服务合成建模和定义的工具对Web服务进行组合的研究已有不少.其中,Cross-Flow[4]和Sword[5]等研究项目致力于为企业提供一个动态的服务查找、发现与运行时动态绑定的服务合成平台,通过WSFL,LANG与BPEL4WS等服务合成语言为服务组合提供了一个统一建模和定义的工具.SciDAC-SDM项目[6,7]将用业务术语描述的抽象工作流与底层的由服务组成的执行工作流严格加以区分,并借用与数据库协调技术相类似的方法将抽象工作流转换成执行工作流.文献[8]利用事务的概念扩展了DAML-S本体中的过程模型,并提出了一个事务工作流本体概念的服务组织模型(transaction workflow ontology,简称TWFO).在以上这些研究中,从工作流业务领域到服务环境的映射只能进行静态的转换,不能根据服务的实时QoS状况,在执行时选取令用户满意的服务实例.

eFlow[9]等服务工作流系统对组合服务的选择过程采用基于QoS局部最优的调度方法.这些方法中,各服务节点对应的具体Web服务实例选择是相互独立的,并不能解决服务工作流的QoS全局优化问题.文献[10,11]将服务的多个QoS约束参数通过线性加权转化为一个单目标函数,利用线性规划的基本原理来解决服务选择的QoS全局最优化调度问题.文献[2,3]利用遗传算法等非线性优化的仿生算法求解QoS全局最优化调度问题,取得的效果不错.但这些研究存在两点不足:1) 未能将Web服务的QoS全局最优化调度求解与服务动态变化的实际结合起来;2) 当网络中可选的Web服务迅速增加时,QoS调度的时间收敛性很难满足用户的实时性需求.

针对以上问题,本文在课题组前期工作的基础上[12]提出了一种基于业务生成图的工作流构造模型.该模型聚集了多个功能相同或者相似的Web服务集合,每个服务集合采用生成树的方式进行组织,并依据工作流间的“上下游”逻辑关系组成业务工作流图;同时,用户可以依据自身的个性化需求对业务逻辑进行裁剪与限定、形成个性化定制模型;最后通过设计基于混合粒子群的QoS调度算法,消除互联网中可选服务迅速增加的影响,找出满足应用需求的工作流执行路径.实验与理论分析表明:该模型能够对动态的Web服务进行有效组合,对工作流执行路径的查找具有良好的时间收敛性,适合于虚拟计算环境的应用要求.

1 服务定义及逻辑工作流模型描述

本节首先给出 Web 服务和用户逻辑工作流结构的严格定义,而用户逻辑工作流定义后的实际运行是将其

映射到实际网络和服务实例的执行过程,因此在第 2 节中给出基于业务生成图的工作流构造模型组织结构,最后在第 3 节给出基于 QoS 的工作流调度算法.

工作流由一系列 Web 服务的调用来完成,下面首先给出 Web 服务的定义.

定义 1. Web 服务(WS)可以定义为一个三元组:$WS=(D,F,Q)$.

1) D 是服务的基本定义与描述,即服务 ID、服务名称、服务的商业实体以及服务的文本描述;

2) F 是服务功能描述,包括服务的接口参数 Parameter、前置条件 Precondition 和后置条件 PostCondition;

3) Q 是对服务非功能属性的描述.它的质量模型可以定义为

$$QoS(WS)=\{QoS(tim),QoS(cos),QoS(cap),QoS(ava),QoS(rep)\},$$

其中,$QoS(tim)$为所用的时间,$QoS(cos)$为所需的费用,$QoS(cap)$为计算能力,$QoS(ava)$表示服务的可用性,$QoS(rep)$表示服务的声誉度.对于 QoS 可能有不同的定义,但不影响本文的主要结论,而且目前的大多数 Web 服务描述语言都自发地遵循(D,F,Q),比如 OWL-S 等.

定义 2. 工作流业务逻辑模型(workflow business logic model,简称 WBLM)是指组成工作流的各类服务相互关联而形成的逻辑结构图.需要指出的是,模型中的服务是指一种服务类型,而不是具体的服务实例,属于此类型的服务实例均可以完成此类型所代表的活动.图 1 所示为某工作流实例.

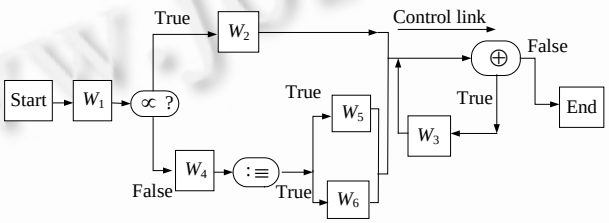


Fig.1 A workflow business logic model

图 1 工作流业务逻辑模型

工作流业务逻辑模型描述为 $WBLM=(MODELS,CL,DD)$,其中:

1) $MODELS=\{W_1,W_2,\ldots,W_n\}$为服务类型的集合,其中,$W_1,\ldots,W_n$表示工作流的$n$个参与服务类型;

2) $CL\subseteq MODELS\times MODELS\times\varphi$为控制连接(control link)的集合.对于$\langle W_i,W_j,\varphi\rangle\in CL$,控制结构运算符$\varphi$定义为从$W_i$到$W_j$的控制转换关系.

基本服务通过若干控制结构可组合成服务工作流.工作流使用的控制结构运算符依据 BPEL(business process executable language)的定义共有 5 个,分别用 sequence,switch,while,flow 和 pick 来表示顺序、选择、循环、并行及事件选择这些逻辑关系.

3) $DD\subseteq W\times W\times\psi$为数据依赖(data dependency)的集合.对于$\langle W_i,W_j,\psi\rangle\in DD$,数据依赖函数$\psi$定义为从$W_j.input$到$W_i.output$的函数依赖关系,并称$W_j$关于$\psi$数据依赖于$W_i$,记作$W_i\,\psi W_j$.

目前,支持对服务流程进行描述和定义的协议标准包括BPEL4WS,BPML和OWL-S等,它们的目标是将服务流程定义和具体的服务实现相分离,但本身并不具备依据多样化、个性化和动态业务需求匹配服务的功能[1].本文采用BPEL4WS来描述工作流业务逻辑模型,主要基于如下两方面原因:(1) 它遵循了通用的标准,采用标准的XML语言,能为其他系统所用;(2) 以其为基础,具体的个性化定制在调度模型中实现,以弥补其个性化定制能力较弱的不足.

2 基于业务生成图的工作流模型构造

本节主要论述基于业务生成图的工作流模型构造.服务工作流组成部分中有关 QoS 调度策略是本文研究的重点,在第 3 节中单独加以论述.

2.1 业务生成图的形成场景

下面给出 Web 服务组织模型的几个基本定义.

定义 3. 服务可替代关系:若某应用能在服务WS_i上满足性能要求,又能在服务WS_j上得到性能满足,则称服务WS_i可替代服务WS_j,记为$Sim(WS_i,WS_j)$.目前,相似服务的匹配大多采用基于本体的语义匹配方法.

定义 4. Web服务生成树(Web services spanning tree,简称WSST)是一类性能相同或者相似Web服务的有效集合.一棵服务生成树用无权无向图来表示,即$G=(V,E)$,其中,$V=n$,是树中服务节点数,节点能提供功能相同的服务;$E=m$,即树的边数.对于服务生成树$WSST_k$的任意两服务(WS_i,WS_j),都有$Sim(WS_i,WS_j)$.

定义 5. Web业务生成图(Web business spanning graph,简称WBSP)是依据工作流的业务逻辑关系,由多棵生成树组合成的图,可以用$G=(T,E)$表示,其中,$T=WSST$(Web服务生成树的集合),是生成图中构成工作流的各个Web服务生成树;$E=\langle T_i,T_j\rangle$,即生成图中两棵生成树之间依据接口和业务逻辑的流程关系形成的边,T_i树中服务的输出是T_j树中服务的输入,即在工作流路径中,T_i服务的"下游"是T_j.

互联网中服务资源异常丰富,同时存在大量的功能相同或相近的服务.根据Web服务在网络中发布的信息,可以采用文献[10,13]提供的策略对其属性进行计算,将某类功能相同(相似)的服务找出,限于篇幅,本文不再赘述;然后组织网络中某类相同(相似)服务$Sim(WS_i,WS_j)$形成一棵生成树,每一棵树设定一个令牌节点[14]来负责树中所有信息的维护.如网络中产生新的服务后,也可以通过如下的简单策略申请加入到服务生成树(WSST)中:

Step 1. 服务提供者以服务的参数向 SRS 发出加入申请报文(*Join_Probe*),申请加入 WSST.

Step 2. SRS 接收到申请报文(*Join_Probe*)后,根据服务的统一编址方法确定此服务所属的权威负责解析 SRS 节点,此 SRS 在表中查找此服务所属的 WSST 的地址,返回给申请服务加入者(joiner)相关 WSST 中的服务节点地址,或者是此类服务还没有服务注册.

Step 3. 如果 joiner 得到的是 WSST 地址,则继续 Step 4;否则,他自己成为此类 WSST 的第 1 个节点,也是令牌节点,则加入过程结束.

Step 4. joiner 向 WSST 连接加入,它首先发出报文(还是 *Join_Probe*),当 WSST 中某节点 v 接收到探测加入报文 *Join_Probe* 时,则返回给发起者肯定应答报文(*Ack_For_Probe*),表示通过此节点加入到 WSST 中.

Step 5. WSST 中的节点接收了新服务的加入,接收节点 v 向此类树中拥有令牌的节点发起通告报文,表示树中增加了一个服务.

Step 6. 在与树内节点的交互过程中,将树内邻居节点 Semi-adjacent 项以及 QoS 相关项填充.

Step 7. 令牌的节点在下一次广播报文中通告树中所有节点.

业务生成图的形成需要在工作流语义环境下进行.基本思想是,在工作流业务逻辑的支持下,工作流起始服务对应的生成树发起业务生成图的形成请求,它首先依据工作流的语义关系得知其"下游"的服务类,然后向 SRS 查询得到其"下游"的生成树地址,进而将"下游"服务联系起来.经过反复的查询与连接过程,从而形成业务生成图,具体构造业务生成图的算法如下:

算法 1. 业务生成图的构造算法.

Step 1. 工作流起始服务通过工作流模式得知"下游"服务的参数,以此向 SRS 发出查询此服务的查询申请报文(*Query_Probe*).

Step 2. SRS 接收到查询申请报文(*Query_Probe*)后,根据服务的统一编址方法确定此服务所属的权威负责解析 SRS 节点,此节点在表中查找该服务所属的 WSST 地址,返回给查询服务者 WSST 中的服务节点地址;或者是空,表示还没有此服务.

Step 3. 如果发起者得到的是 WSST 地址,则继续 Step 4;否则,过程结束.

Step 4. 发起者向 WSST 连接加入,它首先发出报文(还是 *Join_Probe*),当 WSST 中某节点 v 接收到探测加入报文 *Join_Probe* 时;向令牌节点转发,由令牌节点返回给发起者肯定应答报文(*Ack_For_Probe*).

Step 5. 发起者根据得到的信息填充如下几个字段信息:next-adjacent;link-band[],int link-capac[].这时,"上下游"关系已经建立.

Step 6. 新加入 WSST 中的令牌节点接收上游传过来的工作流模式,决定是否继续重复以上步骤,直至工作流关系建立.

2.2 业务生成图的工作流模型

服务工作流构造模型(service workflow constructing model,简称 SWCM)如图 2 所示,由两个有机部分组成:服务注册系统(service registrations system,简称 SRS,见图 2 左上)和 Web 服务工作流生成图(service workflow spanning graph,简称 SWSG,见图 2 中右).

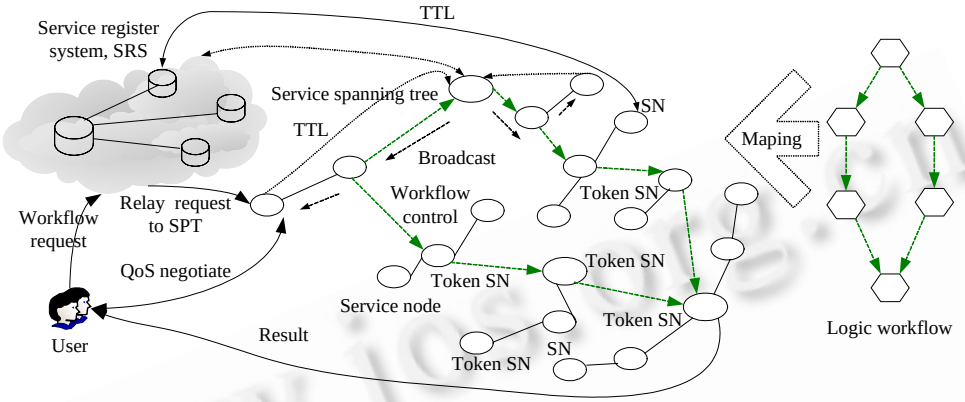


Fig.2 An availability model of Web service workflow

图 2 服务工作流可用性模型

1) 服务注册系统(SRS):主要起到索引服务作用,类似于 DNS 系统,存储的是某类服务对应的服务生成树地址.与文献[15]中提出的主动分布式 Web 服务注册系统的区别是:在文献[15]中主要是利用与 DNS 类似的服务注册系统(SRS),服务提供者直接向 SRS 注册.由于本文采用生成图的方式来组织工作流所对应的服务,因此在服务的组织上,并不需要每一个服务提供者直接向 SRS 注册,而是对于一棵生成树而言,选中其中一个作为令牌节点,由其来负责整棵树与 SRS 的服务信息维护,这样,对于广域网中的一类服务只需要一个节点与 SRS 交互,大幅度减少了 SRS 中服务信息维护的负载压力,而且避免了由于服务的动态产生,直接注册方法产生的系统维护与通信压力过大的问题.

2) Web服务工作流生成图:组织网络中某类相同或相似服务$Sim(WS_i,WS_j)$形成一棵生成树,设定一个令牌节点来负责树中所有信息的维护.为了保持生成树的鲁棒性,对令牌节点采用冗余策略;同时,使令牌节点与SRS间存在定时的交互动作TTL(time to live),目的是让SRS知道树还“活着”,它由令牌节点在TTL交互时将树内信息报告给SRS;同时,依据工作流业务逻辑关系,将服务间的“上下游”对应的生成树之间关联起来(图 2 中的灰色箭头边所示),形成服务工作流生成图.图 2 中右侧所示的用户逻辑工作流到实际服务工作流路的映射由本文的QoS调度算法来完成,而逻辑工作流的描述用上一节的定义 2 来实现.

整个工作流交互过程如图 2 所示.首先,用户向服务注册系统(SRS)发起服务工作流请求,SRS 接收到请求后,依据用户 XML 描述(见第 3.3 节)的工作流起始服务类型转发到对应的服务生成树,再依据第 4 节提出的混合粒子群调度算法依次遍历业务生成图,返回给用户一条最佳的工作流执行路径,执行工作流;如果未找到合适的路径,则用户可以降低 QoS 级别与服务生成树进行 QoS 协商,再进行查找,直到协商成功或者失败为止.

采用业务生成图的原因有以下几点:(1) 采用生成树来保持动态信息维护,能够有效减少服务直接与 SRS 交互所带来的负载压力,虽然内部服务动态性很大,从而使系统的稳定性大大提高;(2) 可通过生成树中节点的个数表示提供此类服务的数目;(3) 组织成生成树后,工作流逻辑到服务实例的映射变得方便且迅速,只需查询 SRS 中生成树的地址,就能找到实际的服务集合;(4) 生成树是一种分布式且自我维护的组织结构,只要存在类似的服务就不会出现因服务动态变化而使服务工作流不可用的现象.

2.3 工作流个性化定制

个性化定制模型是提供给用户一种定制个性化工作流的方法,采用 XML 语言来定制个性化的工作流.这是

一种普遍采用的方法,如WSFLOW.它主要包含3个方面的定制工作,在XML中可以采用3个节来表示,分别是业务模式定制、执行模式定制、QoS参数定制.

1) 业务模式定制是指对业务逻辑模式进行裁剪.在采用XML描述时,首先描述控制符(在service-control节中),然后描述服务类(在service-model节中),这时,可以对业务逻辑进行定制.其中,defaultmodel表示服务类型,如果有多个业务逻辑流程,则可以用allow为真或者假来选择或不选择此业务流程.

2) 执行模式定制是指对服务类到服务实例映射关系的定制与限定.在〈instance〉节中描述,用〈allow instance="*"/〉表示可以映射到任意的服务实例(即服务提供商),用〈deny instance="."/〉表示不能映射的服务实例,如工作流不希望经过某些特定区域,则在此描述.

3) QoS参数定制是指对映射的服务实例,如定义1中所定义的其他非功能性QoS属性的限定,表示将来调度时选择的服务实例必须满足的QoS要求,在〈QoS〉节表示,如用〈time<10/〉表示要求将来选择的服务实例的执行时间小于10.另外,在全局定制节部分〈globalization〉也可以定义整个工作流的QoS要求,用户还可以定制对不同QoS的偏向性,这是在〈Weight〉节中,对不同的QoS指标设置不同的权重来表示的.

下面给出采用XML定制的例子.

```
〈service-control〉〈!--定义业务逻辑的控制符--〉
    〈mode="flow"/〉〈!--定制服务类部分--〉
    〈service-model〉
          〈defaultmodel="A"/〉〈!--服务模式为A类服务--〉
          〈allow="true"/〉〈!--允许此服务模式--〉
     〈instance〉〈!--对服务实例的定制部分--〉
        〈allow instance="*"/〉〈!--允许映射的服务实例(服务提供商)--〉
        〈deny instance="."/〉〈!--不允许映射的服务实例--〉
     〈/instance〉
     〈QoS〉
   …    〈!--对QoS参数的定制部分--〉
〈/QoS〉
/service-model〉
…    〈!--定制其他服务类部分--〉
/service-control〉
…    〈!--其他定制部分--〉
    〈globalization〉〈!--工作流全局定制部分--〉
 〈QoS〉          〈!--对全局QoS参数的定制部分--〉
    〈time<110/〉〈!--工作流总的执行时间小于110--〉
    〈price<350/〉〈!--工作流总的执行费用小于350--〉
    〈capacity<75/〉〈!--工作流总的计算能力大于75--〉
    〈availability<0.7/〉〈!--工作流总的可靠性大于0.7--〉
    〈reputation<80/〉〈!--工作流总的信誉大于80--〉
〈/QoS〉
〈Weight〉          〈!--对不同QoS指标的权重定制--〉
    〈W_time=0.3/〉〈!--工作流执行时间权重为0.3--〉
    〈W_price=0.2/〉〈!--工作流执行费用权重为0.2--〉
〈/weight〉
…          〈!--对其他全局定制部分--〉
```

由上例可知:用户对工作流服务质量的要求为 tim<110,pri<350,cap>75,ava>0.7,rep>80.

3 基于混合粒子群算法的 QoS 调度

互联网中同时存在大量功能相同(相近)、服务质量(QoS)各异的 Web 服务,经过如图 2 所示的服务组织模型,可以将功能相同(相近)的服务进行有效组织,完成用户所需要的基本功能;但功能相同(相近)的 Web 服务,其 QoS 相差很大,如何从中找出“最佳”QoS 的服务实例来构造服务工作流满足用户的需求,将是本节需要解决的问题.QoS 调度实际上就是依据用户个性化定制的工作流,在业务生成图内,从服务质量各异的服务候选中选取“最佳”QoS 服务实例的过程,也即服务工作流中“最佳”执行路径的形成过程.

3.1 混合粒子群算法

随着Internet中Web服务的迅猛增长,服务注册系统中心(SRS)和服务生成树的可选服务都将迅速增加.对于QoS调度,穷举法因候选服务众多已难以满足需要,遗传算法同样也显得力不从心.例如,图 1 所示的某服务工作流系统,6 个服务类($W_1,W_2,...,W_6$)中,若每个服务类有 10 个候选服务,则穷举法的搜索空间为 10^6,穷举法显然不再适合,遗传算法的时间收敛性不强的弱点也逐渐显现出来.这种非线性组合的优化问题是一类NP难问题,而粒子群算法(PSOA)在多目标优化中已广泛应用,且收敛性强,所以,本文利用粒子群算法对构造模型进行QoS调度,以满足用户的相关要求.

粒子群算法[16]首先由Kennedy和Eberhart于1995年提出,是一种基于迭代的进化计算方法,其本质是利用本身信息、局部较优信息和全局较优信息来指导粒子下一步迭代的位置.对于求解“最佳”QoS路径问题,其当前位置是基本路径,若按基本粒子群算法,其速度难以表达.因此,通过重定义粒子群算法中的位置、速度、加、减和乘法操作,并将遗传算法的交叉、变异思想引入粒子群算法中,形成混合粒子群算法(hybrid particle swarm optimization algorithm,简称HPSOA)进行求解.在每次迭代中,既考虑粒子群算法的本身信息、局部较优信息和全局较优信息,又进行类似于遗传算法中的交叉、变异操作,以获得“最佳”QoS执行路径.下面给出重新定义后混合粒子算法的速度与位置操作符.

设 $X=\{x_1^m,x_2^m,...,x_n^m\}$ 表示服务工作流路径的一个解,其中,m 为某生成树中 Web 服务个数,n 为网络中生成树的棵数.将SRS中各点按从左到右、从上到下的顺序自然排序,然后按此顺序将每个待选点作为数组的一个元素,当元素x_i为 1 时,表示相应的点被选入该条路径中;否则,该点不在该条路径中.

粒子位置的重新定义:

$$X=(x_1,x_2,\ldots,x_m) \tag{1}$$

这里,粒子位置 X 表示问题的一个解,包括 m 个元素,每个元素的取值有两个:0 和 1,1 表示相应的点被选入该条路径中,0 则相反.例如,X(1,0,0,1,0,1,0,1)表示一条经过第 1,4,6,8 个节点的路径.

粒子速度的重新定义:

$$V=(v_1,v_2,\ldots,v_m) \tag{2}$$

粒子速度 V 表示路径的改进方向,即根据速度提示的位置和元素对原来路径进行替换,其元素取值有 3 个:–1,0,1.–1 指示要替换的链路的起始位置和结束位置,0 和 1 表示新的路径选择的点.例如:V(0,0,–1,0,1,0,–1,0)表示原路径中从第 3 个点到第 7 个点间的链路进行替换,因为第 3 个元素和第 7 个元素的值为–1,3 和 7 分别表示要替换路径的起点位置和结束位置.第 4,5,6 个元素的值表示新的路径把原来的路径中相应的一段链路替换为经过第 5 个点的路径(第 5 个元素为 1).

“加法”(⊕)的重新定义:$A⊕B$,假设 B 中第 i 个元素和第 j 个元素为–1,则把 A 中第 i+1 到第 j–1 的元素替换为 B 中相应位置的数值.例如,有 X(1,1,1,1,1,1,0,1)和 V(0,0,–1,0,1,0,–1,0),这样,$X⊕V$=(1,1,1,0,1,0,0,1).“加法”实际上是实现了遗传算法中的“交叉”操作,把原路径的一段链路替换成新的链路,而替换的方向是由局部较优信息和全局较优信息指导的.

“减法”(e)的重新定义:A e B,把A和B中数值相同的那个位置标记为–1,如果有多个–1,随机选取两个,结果只保留两个–1,两个–1 间的数值为A相应位置的数值,其他位置为 0.如有X_1(1,0,0,1,0,1,0,0)和X_2(1,1,1,0,1,1,0,0),

则X_1 e X_2=(−1,0,0,1,0,−1,0,0).“减法”实际上是实现“寻向”的操作,通过当前路径与局部较优路径的比较以及当前路径与全局较优路径的比较寻找差值,找到路径替换的方向.

“乘法”(⊗)的重新定义:与减法一起用才有意义,比如 $C1\otimes Rand()\otimes(P(t,i)\ \mathrm{e}\ X(t,i))$中,$P(t,i)$ e $X(t,i)$刚开始可能有多个−1,假设为 n 个,则保留第 $C1\%n$ 个和第 $Rand()\%n$ 个.当前路径与局部较优路径以及全局较优路径比较后,可能会发现多段应该替换的链路,根据 $C1$ 和 $Rand()$给出的概率选择一段路径进行替换.

最后,我们得到新的粒子算法的基本公式:

$$V(t+1,i)=V(t,i)\oplus C1\otimes Rand()\otimes(P(t,i)\ \mathrm{e}\ X(t,i))\oplus C2\otimes Rand()\otimes(P(t,g)\ \mathrm{e}\ X(t,i)) \tag{3}$$

$$X(t+1,i)=X(t,i)\oplus V(t+1,i) \tag{4}$$

经过进行式(3)和式(4)的进化迭代后,可以得到一条新的工作流路径.该路径是在原路径基础上增加或删除某些点得到的.通过检查新路径是否在服务生成树中来判断新路径是否合法,如果是合法的,再计算其性能是否好于原有路径,若好于原有路径,则使用新路径.经过多次迭代,可以得到一定范围内的较优路径.如果增加迭代的次数,则可以保证得到整个网络中的最优路径.

定理 1. 网络中初始路径按式(3)和式(4)多次迭代后,可以找到较优服务工作流执行路径.

证明:假设网络中某服务工作流的初始路径为 $X(0,i)$,其初始速度为 $V(0,i)$,路径的质量指标函数值为 $f(0,i)$,路径的质量指标函数用来评判路径性能的好坏,即其值较高则表示该路径为较优服务工作流路径,其中,i=0,1,…,m.

根据公式(4),经过一次迭代后得到新的路径为

$$X(t+1,i)=X(t,i)\oplus V(t+1,i) \tag{5}$$

其新路径的质量指标函数值为$f(t+1,i)$,显然,只有$f(t+1,i)\geq f(t,i)$才会保留新的路径;进而可以推导出经过多次迭代后,其质量指标函数值$f(t_m,i)\geq f(t_{m-1},i)$,其中,$t_m\geq t_{m-1}$.也就是说,多次迭代后得到的执行路径会比以前的执行路径更优(至少一样).

而根据公式(3),新路径的搜索吸收了前面搜索过程中的有益信息,搜索的速度方向参考了以前的速度方向以及当前较优路径的方向,从而可以得到

$$C1\otimes Rand()\otimes(P(t,i)\ \mathrm{e}\ X(t,i))\geq C1\otimes Rand()\otimes(P(t-1,i)\ \mathrm{e}\ X(t-1,i)) \tag{6}$$

$$C2\otimes Rand()\otimes(P(t,g)\ \mathrm{e}\ X(t,i))\geq C2\otimes Rand()\otimes(P(t-1,g)\ \mathrm{e}\ X(t-1,i)) \tag{7}$$

则 $V(t+1,i)=V(t,i)\oplus C1\otimes Rand()\otimes(P(t,i)\ \mathrm{e}\ X(t,i))\oplus C2\otimes Rand()\otimes(P(t,g)\ \mathrm{e}\ X(t,i))\geq V(t,i)$,从而可以更快地向更优目标接近.故使用公式(3)和公式(4)可以求出较优服务工作流路径. □

3.2 QoS调度算法的设计与分析

构造模型的 QoS 调度过程,其实质就是从如图 2 所示的服务组织模型中选择合适的候选服务进行执行的过程,其核心功能就是服务的选择.因为服务生成树中同时存在大量功能相同(相近)、服务质量各异的候选服务,所以工作流执行过程中还需选择 QoS 合适的成员服务来满足用户的个性化需求.由于不同的服务质量指标在工作流路径中具有不同的性质,有的具有相加性,如执行时间、执行费用等,而有些指标具有相乘性,如可用性指标等,因此对于具体的计算规则,本文参照文献[2]提出的工作流服务质量计算方法.

采用$f_1(x),\ldots,f_p(x)$函数分别代表QoS调度的考虑因素$QoS(tim),QoS(cos),QoS(cap),QoS(ava),QoS(rep)$等.通常,路径要满足计算能力尽量大、开销费用尽量小、执行时间尽量小、服务可用性尽量高、服务可靠性尽量大,也就是希望在QoS上得到满足以下目标的优化目标:

$$f(x)=F(\max f_1(x),\min f_2(x),\min f_3(x),\max f_4(x),\max f_5(x)) \tag{8}$$

式(8)中的 max 函数可以转化为 min 函数,因此,用加权法求得每条链路的综合服务质量函数为

$$g(x)=\min\sum_{j=1}^{5}w_j f_j(x)\geq\partial_j,\ x\in X \tag{9}$$

其中,w_j为根据调度目标给定的一组权值,反映了请求对每个优化目标的权重.为适合不同用户的QoS偏好,本文要求用户向构造模型提交工作流逻辑结构时也要确定不同QoS指标的权重.

而对于服务工作流的每条路径,假设它包括 n 段链路,它的综合服务质量由路径上的每段链路决定,即综合服务质量函数值等于路径上各链路综合质量函数值之和,

$$G(x)=\sum_{i=1}^{n} f_i(x) \tag{10}$$

若要求出 QoS 较优的服务工作流执行路径,首先就要根据服务组织模型和服务工作流逻辑求出服务工作流生成树.对新产生的每条路径,先判断是否为生成树中的路径,只有生成树中的路径才需要进一步判断是否为较优路径,否则需要重新产生路径.在给出的生成树的初始路径中,经过模型中的若干节点,包含若干条链路,每条链路都有相应的服务质量指标值,即 $QoS(tim)$,$QoS(cos)$,$QoS(cap)$,$QoS(ava)$,$QoS(rep)$等.先判断每条链路是否满足多目标条件,即是否满足公式(8),若不满足则要换一条路径,并根据公式(9)求出每条链路的综合服务质量值,然后根据公式(10)求出每条路径的综合服务质量函数值.下面我们给出求较优服务工作流路径的粒子算法实现.

算法 2. (HPSOA)使用混合粒子算法求出较优服务工作流路径.

Step 1. 设定群中的粒子数(Max-Num)以及最大的迭代次数(Max-Iteration).给每个粒子一个随机的初始位置 $X(0,i)$以及一个随机的初始速度 $V(0,i)$,(i=1,2,…,Max-Num),$X(0,i)$表示第 i 条初始路径,$V(0,i)$表示第 i 条初始路径的替换链路方向.局部较优路径 $Pi=X(0,i)$,全局较优路径 Pg 是所有 Pi 中最好的路径.

Step 2. 如果当前迭代的次数等于最大迭代次数 Max-Iteration,则 goto Step 5.

Step 3. 对粒子群中的每一个粒子,根据公式(3)计算新速度,根据公式(4)计算新位置.判断新路径是否符合逻辑工作流的限定条件,如果符合,则再判断路径上的每条链路是否满足公式(8)给出的多目标限制条件,如果满足,则根据公式(10)求得的新路径的适应度函数值比当前局部较优路径的适应度函数值还要小,于是用新的位置更新当前的局部较优路径,goto Step 4;否则,保留原来的位置,goto Step 2.

Step 4. 如果某个粒子的局部较优路径优于当前的全局较优路径,则用这个局部较优路径更新当前的全局较优路径,goto Step 2.

Step 5. 输出求得的最好解路径及其适应度函数值.

在算法 2 中,与普通的随机进化策略相比,增加了进化方向的引导,因而可以更快地向最优目标靠近.在 Step 3 中,首先保证了路径在生成树中,从而满足服务工作流逻辑模型,并且路径的每条链路都满足多目标约束,即服务质量指标,如 $QoS(tim)$,$QoS(cos)$,$QoS(cap)$,$QoS(ava)$,$QoS(rep)$等,都达到规定的要求.经过多次迭代,得到从这些满足多目标约束的路径中选择出来的具有较优 QoS 的路径.

定理 2. 算法 2 可以找到“最优”服务工作流执行路径.

证明:设 ∀ Max-Num 条初始路径 $X(0,i)$,据式(3)和式(4)可知,一次迭代后,其速度和位置分别为

$$V(1,i)=V(0,i)\oplus C1\otimes Rand()\otimes(P(0,i)\ominus X(0,i))\oplus C2\otimes Rand()\otimes(P(0,g)\ominus X(0,i)) \tag{11}$$

$$X(1,i)=X(0,i)\oplus V(1,i) \tag{12}$$

并对初始路径 $X(0,i)$中的相关链路进行替换.由定理 1 可知,每条初始路径经迭代后得到一条新路径 $X(t+1,i)=X(t,i)\oplus V(t+1,i)$,且路径 $X(t+1,i)$具有一定的随机性,并受到原路径和局部较优路径和全局较优路径的影响,其搜索空间为(Max-Iteration–i);又因局部较优路径和全局较优路径的引导,算法将朝着整体最优路径收敛.显然,这些新产生的路径不一定都符合工作流逻辑规范和 QoS 函数要求,只有产生符合工作流逻辑规范且 QoS 函数 $f(t+1,i)\geq f(t,i)$更优的路径才会保存下来.因此,经(Max-Iteration)次迭代后,算法 2 可以找到“最优”服务工作流执行路径. □

算法 2 和遗传算法都随机初始化种群,都使用适应度函数来衡量个体的优劣程度,并根据由适应度函数得到的适应值来进行一定的随机搜索.但算法 2 并不是真正的随机搜索,它的每一次迭代都考虑到了本次路径与各个进化分支的局部最优路径以及全局最优路径之间的差别,以一定的概率向局部最优路径和全局最优路径靠近,因为在进化方向上增加了引导,速度会更快,在搜索性能上好于遗传算法.粒子具有“记忆”的特性,可以通过对本分支的信息和兄弟分支的信息进行分析,从而继承更多的信息,故能在较短的时间内找到所求解.

4 实验结果与分析

4.1 实验环境配置

仿真实验的硬件系统选用联想万全服务器 T100 为 SRS 中心;5 台 PC 机(CPU 为 Intel Pentium IV 2.4GHz,内存为 512MB),其中 3 台 PC 机安装 Windows XP,为 Java 编程模拟网络节点;2 台 Win 2000 微机模拟用户提交申请,全部机器通过交换机 Fast Ethernet ES-3124 RL-24-port 10/100Mbps 组成一个局域网.

网络拓扑图中采用了 Java 编程开发的组件 WorkflowGen,共设计有 500 个服务节点,协作实体编号为 1~500,节点间物理连接度的平均值设为 3,每个节点的边数(连接度)采用 random(1~6)之间的随机函数产生.第 i 个节点的相邻节点的确定:假设 random(1~6)后的连接度为 m,然后从 random(i–km,i+km)号节点中随机选取 m 个节点作为其邻居.该拓扑图符合随机网络实际,且易于编程实现.

拓扑图产生后,我们设计了组件 TaskGen 在网络节点上布置服务,以便用户申请时为其提供服务.一个完整的服务工作流需由多个 Web 服务组成,因此,产生的服务要能构成很多不同模式的工作流,以满足用户的需要.鉴于目前没有相关服务工作流的标准平台和标准测试数据集,本文采用模拟服务数据作为测试用例.首先产生 50 种不同的工作流模板,然后根据此模板产生所需的各个服务,其产生要素包括:模板类别,输入、出接口,功能要求,服务布置所在的网络节点号(从 1~500 中选取)等.

4.2 实验结果及分析

本节从执行成功率、失效率、执行时间以及时间的收敛性等指标来对上述服务组织模型及 QoS 调度算法进行性能评价,具体分为两部分实验.

第 1 部分实验:从服务组织方面来对本文的模型进行评价,比较的对象是Korhonen J[8]利用语义代理构造的模型(TWFO)以及Ludaacher B[6]和Altintas I[7]设计的服务工作流系统(SciDAC-SDM).

实验 1. 服务工作流执行成功率对比.

成功率是衡量 Web 服务工作流性能的重要指标,从图 3 容易看出,随着 Web 服务的增加,本文提出的服务工作流构造模型(SWCM)的执行成功率明显优于其他构造方法.实验同时设计了一种 Web 服务产生与消失的场景,让服务动态产生的同时从系统中“死亡”一定数量的服务;以变化服务数量占总服务数量的比例(%)不断增加为实验的观测点,若构造模型不能对服务进行很好的组织,将导致工作流的执行成功率下降.从图 4 可以看出,在服务动态产生与消失比较频繁的情况下,SWCM 的优势较为明显,说明该模型能够有效地屏蔽 Web 服务的动态变化及影响.

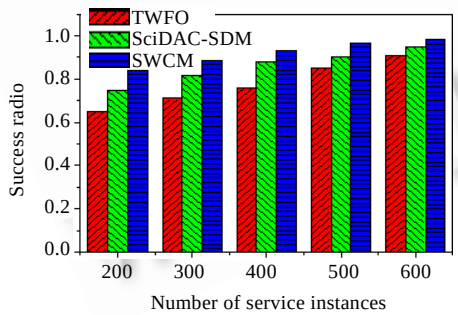


Fig.3 Comparison of success radio

图 3 成功率的比较

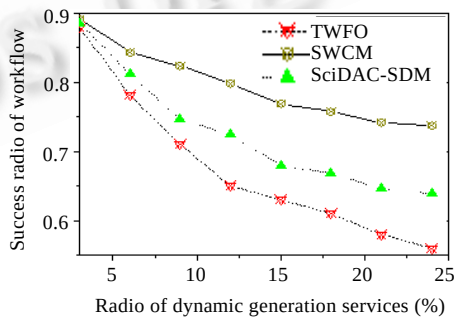


Fig.4 Comparison of success radio under dynamic services

图 4 动态服务下成功率的比较

实验 2. 模型的失效性对比以及新 Web 服务加入的影响.

在 Web 服务动态变化的网络环境中,工作流构造模型很容易失效.由图 5 易知,构造模型 SWCM 随着服务动态变化率的提高,其失效率的增幅慢慢减小,且失效率明显优于其他构造模型.我们同时对新 Web 服务加入构造模型的影响进行了实验.为了分析问题简单,将服务跳数(hop)作为执行时间的衡量参数;新服务刚加入系统

时,负载较轻,其处理能力较大,因此,服务路径选取的机会也较多.图 6 所示为当不同比例的新服务实例加入模型时,经过一段时间后,工作流执行路径中新服务实例达到总服务实例的 50%所花费的时间.该时间越小,表示模型 QoS 优化能力越强.图 6 中以每批次为一个时间单位,每批次将发出 10 个工作流申请.从图中可以看出,大约在第 5 批次后,新服务比例达到 50%.

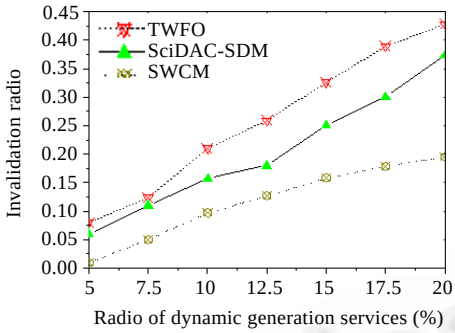


Fig.5 Comparison of invalidation radio under dynamic services

图 5 动态服务下成功率的比较

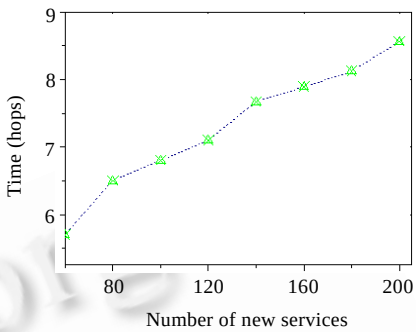
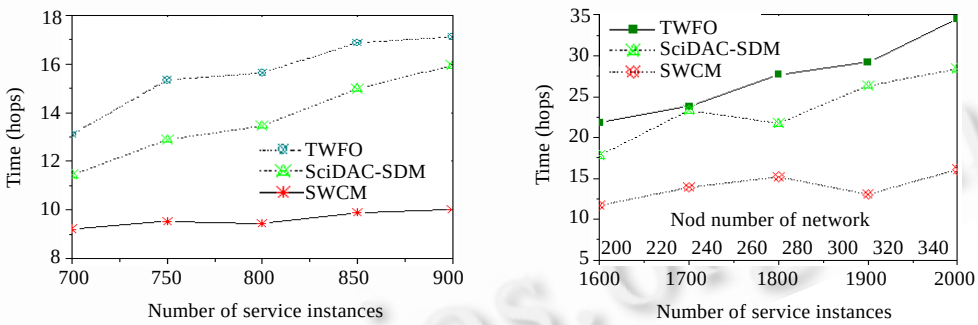


Fig.6 Comparison of execution time

图 6 执行时间的对比

实验 3. 服务实例及网络规模变化后执行时间的对比.

图 7(a)表示了在网络规模一定的情况下,单纯增加服务的数目对模型执行的时间几乎没有影响,而构造模型 TWFO 和 SciDAC-SDM 随服务的增加,执行时间也不断增长.从图 7(b)可以看出,在增大网络规模的同时增加服务实例数,本文提出的 SWCM 模型的执行时间只有小幅增加,网络的规模同样对构造模型的影响也不大,说明本文提出的模型具有很好的可扩展性.



(a) (b)

Fig.7 Comparison of execution time

图 7 执行时间的对比

第 2 部分实验:服务迅速增长时,本文调度算法 2 与遗传算法、eFlow 中局部寻优算法的性能评价.

本实验部分采用如图 1 所示的工作流实例,假定用户对工作流服务质量的要求与第 2.3 节的例子相同,服务质量指标 $QoS(tim)$,$QoS(cos)$,$QoS(cap)$,$QoS(ava)$,$QoS(rep)$分别用 t,co,a,ca,r 表示,则可以得到该工作流 QoS 调度的适应度函数为

$$g(x)=\sum_{j=1}^{5} w_j f_j(x),\ x \in X \tag{13}$$

其中,w_j=1,j=1,2,…,5;$f_1(x)$,$f_2(x)$,$f_3(x)$,$f_4(x)$,$f_5(x)$的计算根据图 1 有两种情况:

(a) 如果在选择活动中选择W_2,那么,路径经过活动W_1,W_2,W_3,计算公式为式(14);

(b) 若在选择活动中选择W_4,则路径经过活动W_1,W_4,W_5,W_6,计算公式为式(15),公式中,m为循环次数.

$$\begin{cases} f_1(x)=t_1+t_2+m\times t_3 \\ f_2(x)=co_1+co_2+m\times co_3 \\ f_3(x)=a_1\times a_2\times a_3 \\ f_4(x)=\min(ca_1,ca_2,ca_3) \\ f_5(x)=r_1\times r_2\times r_3 \end{cases} \tag{14}$$

$$\begin{cases} f_1(x)=t_1+m\times t_3+t_4+\max(t_5,t_6) \\ f_2(x)=co_1+m\times co_3+co_4+co_5+co_6 \\ f_3(x)=a_1\times a_3\times a_4\times \min(a_5,a_6) \\ f_4(x)=\min(ca_1,\min(ca_4,\min(ca_5,ca_6)),ca_3) \\ f_5(x)=r_1\times r_3\times r_4\times \min(r_5,r_6) \end{cases} \tag{15}$$

通过实验可以看到适应度函数的收敛情况,如图 8 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,即种群中个体的适应度值变化将趋于 0.该仿真算法最终选择的候选服务序列分别为 2,7,5,4,6,10,8,适应度函数值为−0.135.

实验还对 eFlow 系统中的局部寻优算法、遗传算法以及本文提出的调度算法 2 在执行时间上进行了比较.分别给定具有 100 个服务、200 个服务、300 个服务、400 个服务、500 个服务的网格模型,它们求得最优解所需要的时间如图 9 所示.从图 9 中容易看出,本文提出的算法其时间收敛性要优于其他算法.

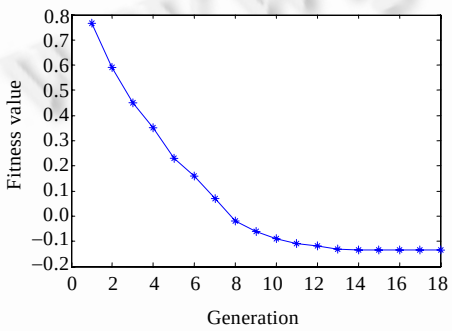


Fig.8 Convergence of fitness function

图 8 适应度函数的收敛

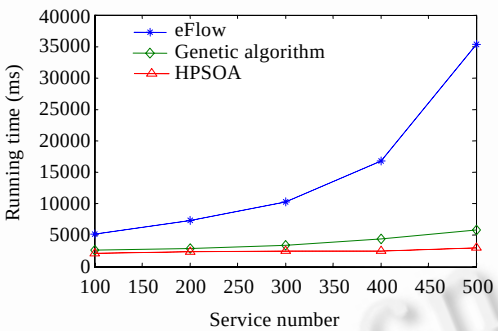


Fig.9 Comparison of executing time

图 9 执行时间的比较

5 结 论

本文提出了一种支持动态服务组织的工作流构造模型.该模型主要将功能相同或者相似的 Web 服务聚合成服务生成树,依据工作流的逻辑关系形成业务生成图的方式来组织服务;由于对服务资源依据工作流的逻辑关系进行了较好的组织,形成了与用户自定义工作流逻辑结构相对应的物理服务实例结构,提供了用户可定制的工作方式,可以充分利用网络上第三方提供的服务;最后给出了构造模型的 QoS 调度方法,利用混合粒子群算法从模型中选取满足用户服务质量要求的“最佳”候选服务并执行.从实验结果和理论分析来看,工作流构造模型较好地解决了因 Web 服务动态变化及迅猛增长引起的难题,有效地屏蔽了组成工作流的 Web 服务在物理上的变化与差异,适合于虚拟计算环境的应用要求.

References:

[1] Papazoglou MP. Service-Oriented computing: Concepts, characteristics and directions. In: Massimo M, ed. Proc. of the 4th Int’l Conf. on Web Information Systems Engineering (WISE 2003). IEEE Computer Society, 2003. 3−10.

[2] Wang Y, Hu CM, Du ZX. QoS-Aware grid workflow schedule. Journal of Software, 2006,17(11):2341−2351 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/17/2341.htm

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

646.htm
[4] Grafen P, Aberer K, Hoffner Y, Ludwig H. Cross-Flow: Cross-Organizational workflow management in dynamic virtual enterprises. Int'l Journal of Computer Systems Science and Engineering, 2000,15(5):277−290.
[5] Ponnekanti SR, Armando F. SWORD: A developer toolkit for building composite Web services [EB/OL]. Stanford: Stanford University, 2002. http://www2002.org/CDROM/alternate/786/
[6] Ludäscher B, Altintas I, Gupta A. Compiling abstract scientific workflows into Web service workflows. In: Proc. of the 15th Int'l Conf. on Scientific and Statistical Database Management (SSDBM). 2003. 251−255. http://kbis.sdsc.edu/SciDAC-SDM/ludaescher-compiling.pdf
[7] Altintas I, Memon A, Ludaacher B. Design and execution of scientific workflows using Web services. In: Proc. of the Presentation, San Diego Supercomputer Center and San Diego Software Industry Council to Host Web Services Conf. 2004. http://daks.ucdavis.edu/~ludaesch/Paper/SDSIC-01-04.pdf
[8] Korhonen J, Pajunen L, Puustjärvi J. Automatic composition of Web service workflows using a semantic Agent. In: Proc. of the IEEE/WIC Int'l Conf. on Web Intelligence (WI 2003). 2003. 566−572. http://csdl.computer.org/comp/proceedings/wi/2003/1932/00/19320566abs.htm
[9] Casati F, Ilnicki S, Jin LJ, Krishnamoorthy V, Shan MC. eFlow: A platform for developing and managing composition e-services. Technical Report, HPL-2000-36, Palo Alto: HP Laboratories, 2000.
[10] Liu YT, Ngu AHH, Zeng LZ. QoS computation and policing in dynamic Web service selection. In: Feldman SI, Uretsky M, Najork M, Wills CE, eds. Proc. of the WWW 2004. New York: ACM Press, 2004. 66−73.
[11] Cardoso J, Sheth A, Miller J, Arnold J, Kochut K. Quality of service for workflows and Web service processes. Journal of Web Semantics, 2004,1(3):281−308.
[12] Hu CH, Wu M, Liu GP, Wang SC. Research on schedule model and algorithm oriented client demand for service workflow. Journal of Chinese Computer Systems, 2007,28(6):1008−1014 (in Chinese with English abstract).
[13] Zeng LZ, Benatallah B, Anne HH, Dumas M, Kalagnanam J, Chang H. QoS-Aware middleware for Web services composition. IEEE Trans. on Software Engineering, 2004,30(5):311−327.
[14] Hong SH, Kim YC. Implementation of a bandwidth allocation scheme in a token-passing fieldbus network. IEEE Trans. on Instrumentation and Measurement, 2002,51(2):246−251.
[15] Du ZX, Huai JP. Research and implementation of an active distributed Web service registry. Journal of Software, 2006,17(3): 454−462 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/17/454.htm
[16] Eberhart R, Kennedy J. A new optimizer using particles swarm theory. In: Proc. of the 6th Int'l Symp. on Micro Machine and Human Science. 1995. 39−43.

附中文参考文献:

[2] 王勇,胡春明,杜宗霞.服务质量感知的网格工作流调度.软件学报,2006,17(11):2341−2351. http://www.jos.org.cn/1000-9825/17/2341.htm
[3] 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
[12] 胡春华,吴敏,刘国平,王四春.服务工作流中基于用户需求的调度模型及算法研究.小型微型计算机系统,2007,28(6):1008−1014.
[15] 杜宗霞,怀进鹏.主动分布式 Web 服务注册机制研究与实现.软件学报,2006,17(3):454−462. http://www.jos.org.cn/1000-9825/17/454.htm

胡春华(1973—),男,湖南新化人,博士生,讲师,主要研究领域为 Web 服务工作流,实时系统调度.

刘国平(1962－),男,博士,教授,博士生导师,主要研究领域为网络控制,智能控制.

吴敏(1963—),男,博士,教授,博士生导师,主要研究领域为计算机网络,智能系统.

徐德智(1963－),男,博士,教授,CCF 会员,主要研究领域为 XML 技术,语义 Web.