基于语义 Web Service 的需求驱动服务合成

赵安平 王晓勇 邱玉辉

(西南大学计算机与信息科学学院 重庆 400715) (西南大学语义网格实验室 重庆 400715)

摘 要 语义 Web Service(SWS)是应对商务对商务环境的挑战,是迈向自动 Web 服务的重要步骤。基于 SWS 的商务服务合成是服务计算领域最为活跃和开放的研究问题之一。提出了一种基于 SWS 的商务服务合成的基本思想和方法。给出了商务服务合成的形式化模型。介绍了一个服务合成中基于超图的服务集合挖掘方法。给出了面向服务合成的需求驱动服务选择及其解决方案模型。

关键词 SWS,商务过程,服务集合,服务选择,服务合成

中图法分类号 TP301 **文献标识码** A

SWS Based Business Requirement Driven Service Composition

ZHAO An-ping WANG Xiao-yong QIU Yu-hui

(Faculty of Computer and Information Science, Southwest University, Chongqing 400715, China)

(Semantic Grid Laboratory, Southwest University, Chongqing 400715, China)

Abstract Semantic Web Service (SWS) is a step towards automating Web Services in order to face the challenges in Business to Business. SWS based business services composition is one of the most hyped and addressed issues in the field of services-oriented computing. This paper provided basic ideas and methods toward SWS based business services composition. First, a formal model of business services composition was introduced. Second, an approach to service set mining for service composition was presented, and finally, the paper presented the solution model of requirement driven service selection and composition.

Keywords SWS, Business process, Service set, Service selection, Service composition

1 引言

随着语义 Web Service (SWS)及其相关技术和标准的开发,众多构成分布式计算框架的 SWS 出现在商务服务过程中。SWS 是迈向自动 Web 服务的重要步骤,它将不断发展的 Web 服务体系结构和语义 Web 相结合,以应对商务对商务(B2B)环境和企业应用集成(EAI)的挑战。实现了对 Web 服务的语义描述,以及动态无缝的服务发现、选择、合成、协调和执行。因此,越来越多的组织正在使用 SWS 来改善与商务伙伴之间合作协调的效率。

如今,更多的企业只实现其核心业务,而在网上外包其他相关的互联网应用服务。Web 服务变得越来越复杂,涉及了很多复杂的分布式交互过程中的业务对象。单一的服务不可能实现用户所要求的服务功能,为了实现用户的请求,需要组合现有的简单服务,通常,动态地组合简单的 SWS 为用户提供功能更复杂的合成服务以满足用户的复杂需求。一个合成服务综合了多个简单服务的作用和功能,形成一个功能更复杂的单个服务,因此它也可以作为一个单个服务去进一步组合或作为完整的应用和服务为客户提供解决方案。

服务合成是服务计算领域最为活跃和开放的研究问题之一,因为服务计算的主要目标是有效并且高效地重用和组合现有的简单服务而形成新的商务服务。为了实现这一最终目标,以下几个方面应该加以考虑。首先,随着每天不断出现的新服务,服务的空间是巨大的。如何有效地确定合理的候选服务集合是一个挑战。第二,如果有多个具有相似变量特征的服务,如何找到其中一个最适当的,以更好地满足商务需求,以及如何准确地捕获不断变化的商务需求。第三是如何形式化服务合成的问题[1]。

为了提供系统化的方法,实现需求驱动的服务合成,本文提出了一种基于 SWS 的商务服务合成的基本思想和方法。

2 商务服务合成的形式化

定义 1(服务集群) 一个服务集群 SC 定义为

$$SC=\{s_1,s_2,\cdots,s_n\},1\leqslant n\leqslant M \quad (1)$$

式中,$s_1,\cdots,s_M$ 代表在 SC 中具有相同特定功能的 M 个服务。

在服务选择和商务服务合成的过程中,最终多个具体真实的服务将从多个服务集群中选出并进行组合,一个商务过程的合成只关注服务集群层而不是单个的具体服务。因为一个选择出的具体服务可能会在调用时变得不可用,它应该由在同一个服务集群中的另一个可用服务来代替,而这一切对

到稿日期:2009-10-02 返修日期:2009-12-07 本文受 973 国家重大基础研究计划(2003CB317008),西南大学研究生科技创新基金(ky2009005)资助。

赵安平(1978—),男,博士生,主要研究方向为服务计算、语义网格,E-mail:zap@swu.edu.cn;王晓勇(1972—),男,博士生,工程师,主要研究方向为语义网格;邱玉辉(1938—),男,教授,博士生导师,主要研究方向为人工智能、语义网格。

于商务过程中的用户来说是透明的。因此，商务服务选择可被形式化为一个和多个服务集群相关的函数。

定义 2(服务集) 服务集 S_{set} 指一个具体商务过程中候选服务的集合，记作

$$S_{set}=\{s_{i(1)},s_{i(2)},\cdots,s_{i(n)}\},1\leqslant i\leqslant M;1\leqslant i(n)\leqslant N \quad (2)$$

式中，$s_{i(n)}$ 表示在一个服务集群 SC_i 中的 n_{th} 商务服务。

在一个特定的商务过程中所有相关的服务集群聚集在一起形成一个服务集。换句话说，服务集的概念代表了从高级的服务选择技术中获得的候选服务的总集合。服务集包含多个服务集群，每个服务集群包含多个候选服务。

式(2)表明一个商务服务是由 $i(n)$ 个单个服务构成的。$i(n)$ 的大小取决于商务需求，使用不同的组合规则和属性，同样的服务集合可以产生不同的商务过程。

定义 3(商务服务选择) 一个商务服务选择的过程 BS 定义为

$$BS=f(SC_1,SC_2,\cdots,SC_i),1\leqslant i\leqslant M \quad (3)$$

式中，SC_i 表示一个服务集群，f 是一个将多个服务集群收集在一起进行商务服务选择的特定函数。式(3)表明高级服务选择的目标是在每一个服务集群定义的服务集合中发现可用服务。商务服务的选择过程始终是一个在特定服务集上的函数 f。

定义 4(商务服务合成) 设 (f,x) 和 (g,y) 分别为两个服务的属性，c_f 为一个定义的合成规则算子，称

$$BP=\begin{cases}\varphi((f,x),(g,x))=\{(f,c_f(x,y))\},\text{if } f=g\\ \varphi((f,x),(g,x))=\{(f,x),(g,y)\},\text{otherwise}\end{cases} \quad (4)$$

为一个商务服务合成。BP 表示在一个特定的合成函数 φ 下合成的商务过程。

3 挖掘服务集

3.1 基于 SWS 的服务语义交互链

在 SWS 的帮助下，商务过程中代理可以根据它们之间相互的通信解释和分析获得信息[2,3]。简单对象访问协议(Simple Object Access Protocol，SOAP)[4]为客户和服务定义了一个基本的通信协议，互相交换 XML 消息。它是一个在像互联网这样的分散环境中交互信息的协议，因为在 Web 服务之间所有的交互都是通过交换运行在 HTTP 协议之上的 SOAP 消息实现，SOAP 消息头(SOAP headers)典型地用于表示消息的元信息，如消息标识符等。而消息头改变技术提供一种非入侵的方式去扩展客户端的功能，提高服务提供者的日志收集能力，收集 Web 服务之间必要的语义交互信息。这种技术也具有很大的灵活性，它能实现动态的增加和删除语义管理属性，而不用改变现有服务的代码。

一个语义 SOAP 消息头(Semantic Soap Header，SSH)是一个管理属性的集合，用于填充 Web 服务之间的每一个信息流。一个 SSH 的结构取决于两个 Web 服务之间的交互管理信息，消息的发送者填写一个 SSH 的某些属性，而消息的接收者填充其余的属性。一个完整的包含了父子关系属性的典型 SSH，如图 1 中〈SSH〉…〈/SSH〉标签所示。SOAP 消息头能方便地插入 SSH 结构，附带 SSH 结构的 SOAP 消息如图 1 所示。

采用分布式消息追踪算法[5]生成服务语义交互链，每一个分布的服务在信息交互的过程中生成服务链。消息追踪算法由两部分组成，分别是每当服务发出了一个消息时的执行和每当收到一条消息时的执行。

```
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/
                  envelope/"
  SOAP-ENV:encodingStyle="http://schemas.xmlsoap.o
                  rg/soap/encoding/">
 <SOAP-ENV:Header>
  <SSH>
    <parent_SSH></parent_SSH>
    <message_id>1122-3551-5721</message_id>
    <message_name>OlineShopping
    </message_name>
    <source>shoppingagent.com</source>
    <target>shoponline.com</target>
  </SSH>
 </SOAP-ENV:Header>
 <SOAP-ENV:Body>
  <TravelArrangement>…</TravelArrangement>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

图 1 附带 SSH 的 SOAP 消息

每个 SWS 可以看到在整个商务过程中由 SWS 发起的第一条交互消息开始，到最后一条消息终止所生成的整个服务语义交互链。最后对整个服务语义交互链进行裁剪，移出一些属性并且合并请求消息结点和确认消息结点。以在线计算机购买 Online Computer Purchase 为例，在执行分布式消息追踪算法后产生服务语义交互链，并进行裁剪后，一个服务链如图 3(d)所示。

3.2 服务语义交互链的超图矩阵表示

定义 5(超图)[6] 设 $V=\{v_1,v_2,\cdots,v_n\}$ 是一个有限集。若

(1) $e_i\neq\Phi(i=1,2,\cdots,m)$

(2) $\bigcup_{i=1}^{m}e_i=V$

则称二元关系 $H=(V,E)$ 为一超图。V 的元素 $v_1,v_2,\cdots,v_n$ 称为超图的顶点，$E=\{e_1,e_2,\cdots,e_m\}$ 是超图的边集合，集合 $e_i=\{v_{i1},v_{i2},\cdots,v_{ij}\}(i=1,2,\cdots,m)$ 称为超图的边。一个超图如图 2 所示。

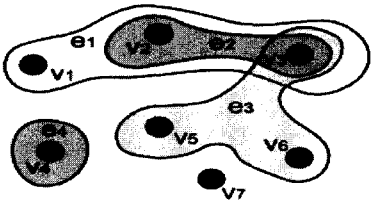


图 2 超图 $G=(V,E)$，$V=\{v_1,v_2,v_3,v_4,v_5,v_6,v_7\}$，$E=\{e_1,e_2,e_3,e_4\}=\{\{v_1,v_2,v_3\},\{v_2,v_3\},\{v_3,v_5,v_6\},\{v_4\}\}$

超边是连接两个或更多顶点之间的无向边，在数学上，超图是图的一般情况，超图中的一条超边可以连接任意多个顶点。V 可以理解为元素的集合，称作结点或顶点。E 是 V 的非空子集，叫做超边或超链。图的边是结点对，而在超图中，一条边不一定只包含两个结点，超边是任意结点的集合，因此包含任意结点数。

定义 6(超图的关联矩阵) 超图 $H=(V,E)$ 的关联矩阵为

$$S=\begin{bmatrix}s_{11}&s_{12}&\cdots&s_{1n}\\ s_{21}&s_{22}&\cdots&s_{2n}\\ \cdots&\cdots&\cdots&\cdots\\ s_{m1}&s_{m2}&\cdots&s_{mn}\end{bmatrix}$$

满足下面的条件：

- S 中每一行对应 H 中的一个顶点；
- S 中每一列对应 H 中的一条超边；
- 如果顶点 $v_i\in E_j$，元素 $s_{ij}=1$；否则，$s_{ij}=0$。

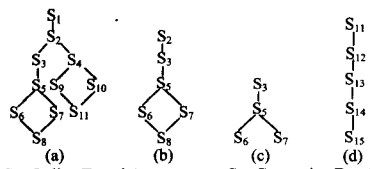
定义 7(服务集的相等) 服务集 A 和 B 是相等的，当且

仅当 $\forall x[x\in A\leftrightarrow x\in B]$，记作 $A=B$。

一个服务集 A 是服务集 B 的子集，当且仅当 A 中的所有元素都包含于 B 中，记作 $A\subseteq B$。

我们可以把在合成服务中涉及的单一服务间的关系只看作是交互或正交关系。当服务 S_A 以某种方式依赖于服务 S_B 时，称 S_A 与 S_B 交互，反之亦然。当服务 S_A 与 S_B 相互独立时，称 S_A 与 S_B 正交。它们可以被顺序地或并行地执行。服务语义交互链表明了在一个合成服务过程中涉及到的单一服务间的交互关系，将所有的单一服务看作一个顶点集，而将每一条服务交互链看作一条超图的超边，超边包含了一个商务过程中出现在服务链中的所有结点。我们并不在意服务交互的顺序，因此，用超图的关联矩阵能够非常方便地表示服务语义交互链。

用一个例子来说明这种表示方法。由文中前面提到的方法收集到的有关合成服务的服务语义交互链分别是 Online Travel Arrangement，Generating Best Travel Route，Get Weather and Road Situation and Online Computer Purchase，为便于更加清晰地理解，将 Online Travel Arrangement 链分成两个子链，因此共有 5 条服务链，如图 3 所示。



S_1. Online Travel Arrangement S_2. Generating Best Travel Route
S_3. Weather and Road Situation S_4. Hotel and Fight Reservatio
S_5. Transportation System S_6. Weather Report
S_7. Road Situation S_8. Generating Route
S_9. Hotel Reservation S_{10}. Fight Reservation
S_{11}. Online Computer Purchase S_{12}. Computer Reservation
S_{13}. Online Payment S_{14}. Credit Checking
S_{15}. Shipment

图 3　服务语义交互链

每条链对应于一个合成服务中各个单个服务间的交互路径。服务链的超图矩阵表示如图 4 所示。

	e_1	e_2	e_3	e_4	e_5
s_1	1	1			
s_2	1	1	1		
s_3	1		1	1	
s_4		1			
s_5	1		1	1	
s_6	1		1	1	
s_7	1		1	1	
s_8	1		1		
s_9		1			
s_{10}		1			1
s_{11}				1	
s_{12}		1		1	
s_{13}		1			1
s_{14}					1
s_{15}					1

图 4　服务链的超图矩阵表示

3.3　算法

算法描述：一个合成服务可以表示为一个二元组，记为 $Com\_Service=(CS, ServiceSet)$，其中 CS 是合成服务的名称；$ServiceSet$ 是合成服务 CS 所涉及的简单服务的集合。假设超图的顶点集由 n 个服务构成：$S_1, S_2, \cdots, S_n$，超图的关联矩阵为 $S=(a_{ij})_{m\times n}$，超图中的每一条超边是一个合成服务所涉及的单个服务所构成的一条服务语义交互链，包含了一个商务过程中出现在服务链中的所有结点。在服务链中出现的结点，相应地在超图的关联矩阵中 $a_{ij}=1$，否则 $a_{ij}=0$。基于此，设计服务合成中的服务集合挖掘算法 $SerMin(S=(a_{ij})_{m\times n})$。

进行算法分析，执行算法所需要的时间为 $T(n)=m+m(m-(i+1)+1)=m^2+m-m*i$。在最坏的情况下，该算法的时间复杂度是 $O(m^2)$，即矩阵 S 的所有的元素是 1。

以图 4 所示超图的矩阵为例来说明在服务合成过程中应用超图的方法进行服务集挖掘的灵活性及算法的执行过程。执行算法 $SerMin(S=(a_{ij})_{m\times n})$ 后，挖掘出的服务集合为 $CService$：$\{(S_1,\{S_1,S_2\}),(S_2,\{S_2,S_3,S_5,S_6,S_7,S_8\}),(S_{13},\{S_{13},S_{14}\})\}$，算法执行的过程及结果如表 1 所列。

表 1　算法执行步骤

Step	Com_Service	CService
1	$(S_1,\{\})$	$\{\}$
2	$(S_1,\{S_1,S_2\})$	$\{(S_1,\{S_1,S_2\})\}$
3	$(S_2,\{S_2,S_3,S_5,S_6,S_7,S_8\})$	$\{(S_1,\{S_1,S_2\}),(S_2,\{S_2,S_3,S_5,S_6,S_7,S_8\})\}$
4	$(S_{13},\{S_{13},S_{14}\})$	$\{(S_1,\{S_1,S_2\}),(S_2,\{S_2,S_3,S_5,S_6,S_7,S_8\}),(S_{13},\{S_{13},S_{14}\})\}$

请注意，在以上算法中为了简单性和更一般化，我们将所有功能相同的服务看作一个服务。这意味着，在服务集中的每个服务等同于前面部分所定义的服务集群的概念。

Algorithm　$SerMin(S=(a_{ij})_{m\times n})$

1. 合成服务集合 $CService$ 为空；
2. For $i=1$ to m Do
3. 　扫描第 i 行，统计非零元素数 k；
4. 　If $k\geqslant 2$ Then
5. 　　生成一个二元组 $Com\_Service$，将第 i 行结点设为 $Com\_Service$ 的第一个域；
6. 　　For $h=i+1$ to m Do
7. 　　扫描由第 i 行 k 个非零元素所在的列构成的矩阵 $S=(b_{ij})_{(m-i)\times k}$；
8. 　　统计第 h 行非零元素数 num；
9. 　　If $num\geqslant 2$ Then 将第 h 行的结点加入 $Com\_Service$ 中 $ServiceSet$ 域；
10. 　　End For
11. 　　If $CService$ 非空 Then
12. 　　将 $ServiceSet$ 和 $CService$ 中每一个元素的第二个域 $ServiceSet'$ 比较；
13. 　　If ($ServiceSet=ServiceSet'$ OR $ServiceSet\subseteq ServiceSet'$) Then 删除 $Com\_Service$；
14. 　　If ($ServiceSet'\subseteq ServiceSe$) Then 删除 $CService$ 中的该元素
15. 　　将 $Com\_Service$ 加入服务集合 $CService$；
16. 　　End If
17. 　End If
18. 　返回 $CService$；
19. End For

4　需求驱动的服务选择

4.1　需求驱动的服务选择描述

商务服务需求的不断动态变化，给商务服务的合成提出了严峻的挑战。首先，商务过程是由商务需求所驱动的，而商

务需求通常是非正式的、主观的，难以量化。将商务需求的描述恰当地转化为可量化、客观的、机器可读的格式，对商务服务过程的合成是至关重要的。其次，现有的基于 Web 服务的商务过程描述语言不能充分地满足详细的需求规范，因此很难向用户提供最佳的商务过程合成。一个典型的商务过程一般要求在复杂的商务需求的基础上由多个服务相互协作来完成。因此，每个服务不仅要满足单个的需求，也需要与其他服务共存，以便最好地适应整个商务服务的合成过程。

需求建模和搜索算法的目标是减少搜索空间。可以想象，一个可能的服务空间，包含了数量巨大的服务。如果在这个巨大的服务空间中去检测每个可能的候选服务，是低效的，也不实际。一个合成的商务过程应满足相应的商务需求，并且是所能产生的最佳的商务过程。因此，一个商务过程的构建，等同于从可获得的服务集中选择最合适的服务，即需求驱动的服务选择(Requriement Dirven Services Selection)RDSS。

4.2 RDSS 问题的形式化

给定一个域和 SWS 集，以及一个客户的需求，寻找 SWS 和服务请求者的需求匹配，我们称作 RDSS 问题。为了形式化以便理解 RDSS 问题，需要定义以下元素和操作算子的集合：

(1)简单 SWS 集，构成了一个服务域或社区；

(2)一个功能的集合，用于描述域的简单服务；

(3)一个简单服务的“属性:值”对的属性的集合；

(4)定义一些基于描述服务的属性之上的计算规则的操作算子；

(5)服务关系或关系链的集合。

定义 8(RDSS 问题) 一个 RDSS 问题是一个七元组，

$S_{select}=(S,F,V,P,C,R,D)$

式中，S 是有限服务集，界定了 RDSS 问题的空间。

F 是有限功能集。

V 是值的集合，$V=\{v_f \mid f\in F\}$。

P 是属性的集合，$P=\{p_s \mid s\in S\}$。每个服务 s 的属性集合 p_s 由形如(f,x)的对构成，其中，$f\in F$，$x\in v_f$，且每个功能 f 在 p_s 中最多出现一次。属性集 P 定义了一个给定服务的功能的值，包含了所有服务 S 的属性。

C 是合成算子的集合，$C=\{c_f \mid f\in F\}$。一个合成算子指定了当一个只满足客户商务需求的部分功能的新的服务加入一个给定的服务集时，两个服务功能的值怎样被合成一个新值。

R 是关系链的集合，$R=\{r_f \mid f\in F\}$。每个功能 f 具有一个关系链 r_f，表示服务之间关系的信息。

D 是一个有限的需求集(也是属性集)，每个需求 d 是一个形如(f,x)的对，其中 $f\in F$，$x\in v_f$。

4.3 RDSS 的解决方案

设 $S_{select}=(S,F,V,P,C,R,D)$是一个 RDSS 问题。一个解决 S_{select} 的方案结构是 $C_s=(I,Q)$，其中 I 是形如(k,s)对的集合，Q 是形如(f,x)对的属性集合。一个(k,s)对表示服务 s 在方案中被调用 k 次，在一般情况下，一个 SWS 在商务合成过程中仅被调用一次($k=1$)；Q 是简单或合成属性的集合，如果满足一个请求需要调用多个服务时，Q 就会包含合成属性，被调用的服务的属性会被合成为一个新的合成属性。

1. $C_s=(\emptyset,\emptyset)$。

2. 如果 $C_s=(I,Q)$是一个 SWS(简单服务或合成服务)，且 s 是一个服务，$s\in S$，满足下列条件则 $S'=(I',Q')$也是一个服务：

对于每一个$(f,x)\in p_s$ 和$(g,y)\in Q$，定义合成 $\varphi((f,x),(g,y))$或者 $Q=\emptyset$。

$I'=nI\backslash\{(k,s)\}\cup\{(k+1,s)\}$，$\exists(k,s)\in I$；$I'=I\cup\{(1,s)\}$

$$Q'=\begin{cases}\cup\varphi((f,x),(g,y))\mid(f,x)\in p_s,(g,y)\in Q, & \text{if } Q\neq\phi\\ p_s, & \text{if } Q=\phi\end{cases}$$

定义 9(有效的 RDSS 问题方案) 在一个 RDSS 问题 $S_{select}=(S,F,V,P,C,R,D)$的候选方案中，如果当且仅当对于每一个需求 $d=(f,x)\in D$，存在一个质量 $q=(g,y)\in Q$，Q 是质量的集合，使得 f 等于 g，且存在一个测试 t_f 说明了在什么条件下一个需求被满足使得 $t_f(x,y)=$真，则称该方案是有效的 RDSS 问题方案。

这意味着在一个候选方案成为有效方案之前，在 D 中所有的请求必须被满足。

由于许多 SWS 有相同或相似的功能是相当普遍的，因此，有可能会产生多个满足要求的合成服务。在这种情况下，使用非功能属性提供的信息整体效用来评价合成服务，最常用的方法是使用效用函数[1]。请求者应指定每个非功能属性的权值，最好的合成服务应该是排序最靠前的。当配置一个新的商务过程时，第一步是服务选择标准的确定，也是最关键的问题。通常将其最紧密地与商务需求相匹配为一个服务选择步骤的优化准则。

结束语 本文主要探讨了基于语义 Web 服务的商务服务合成，并提出了一种基本的思想和方法。通过对商务服务合成形式化的分析，提出一个在服务合成中必要的服务集合挖掘的方法。进一步，需求驱动的服务选择也是一个在服务合成过程中非常重要的问题，文章讨论了 RDSS 问题的方案模型。

本文仅仅提供了一种商务服务合成的基本思想和方法，在现实的应用中，为了完成可行的服务合成，许多研究问题仍然期待更多的研究者去解决。

参 考 文 献

[1] Zhang Liangjie, Zhang Jia, Cai Hong. Services Computing [M]. Beijing: Tsinghua University Press, 2007

[2] Burstein M, Bussler C, Finin T. A Semantic Web Services Architecture[J]. IEEE Internet Computing, 2005, 9(5): 72-81

[3] Geng Xiangyang, Huang Linpeng, Li Dong. Intelligent Data Transferring Based on Semantic Web Services[C]// Proceedings of the 2004 IEEE International Conference on Services Computing. Washington DC: IEEE Computer Society, 2004: 463-466

[4] Simple Object Access Protocol[OL]. http://www.w3.org/TR/soap. 2000

[5] Sahai A, Machiraju V, et al. Message Tracking in SOAP-based Web Services[C]// Proc. of IEEE/IFIP Network Operations and Management Symposium (NOMS). 2002

[6] Berge C. Graphs and Hypergraphs [M]. New York: Elsevier, 1973

[7] Papazoglou M P, Traverso P, et al. Service-Oriented Computing: State of the Art and Research Challenges[J]. Computer, 2007: 64-71

[8] Meng Hui, Wu Li-fa. Mining Frequent Composite Service Patterns[C]// Proc. of 7th International Conference on Grid and Cooperative Computing. 2008: 713-718

[9] Lian Qianhui, et al. Service Pattern Discovery of Web Service Mining in Web Service Registry-Repository[C]// Proc. of IEEE International Conference on e-Business Engineering (ICEBE'06). 2006

(下转第 235 页)

直接进入到子代群体中，这样可以防止最优个体由于遗传算子的偶然性而被破坏掉，且能以接近 1 的概率来找到全局最优解；也可将父代与子代一起排序，按群体规模截取前 N 个个体进入匹配集，作为下一步交叉操作的对象，这样可以使每一代中的优良个体得到保护，而淘汰那些不良个体，以期通过交叉等操作产生更优的个体，从而提高寻优速度。

2.5.2 自适应交叉和变异操作

普通自适应算法中，个体适应度值越接近最大适应度值，交叉概率与变异概率就越小；当它等于最大适应度值时，交叉概率和变异概率为零。这种调整方法对于种群处于进化后期比较合适，但对于进化初期不利，因为进化初期种群中的较优个体，尤其是适应度值接近种群最大适应度值的个体，几乎处于一种不发生变化的状态。这增加了进化走向局部最优的可能。因此，需要做进一步修改，使种群的最大适应度值的个体的 P_c 和 P_m 不为零。修改后的公式为：

$$P_c=\begin{cases} P_{c1}-\dfrac{(P_{c1}-P_{c2})(f'-f_{avg})}{f_{\max}-f_{avg}}, & f'>f_{avg} \\ P_{c1}, & f'\leqslant f_{avg} \end{cases}$$

$$P_m=\begin{cases} P_{m1}-\dfrac{(P_{m1}-P_{m2})(f_{\max}-f)}{f_{\max}-f_{avg}}, & f>f_{avg} \\ P_{m1}, & f\leqslant f_{avg} \end{cases}$$

当 $f'=f_{\max}$ 时，$P_c=P_{c2}>0$；当 $f=f_{\max}$ 时，$P_m=P_{m2}>0$。这样，种群中的较优个体相对于普通自适应遗传算法，拥有更高的交叉概率与变异概率。同时，为了保证每一代的最优个体不被破坏，采用最优保存策略，将其直接复制到下一代中。

上述公式中，一般取 $P_{c1}=0.9$；$P_{c2}=0.6$；$P_{m1}=0.1$；$P_{m2}=0.001$。

3 仿真实验结果

作者采用 Gridsim 网格任务调度模拟器，对改进后的遗传算法 IGA 进行了仿真，并将 IGA 与传统自适应遗传算法 AGA 就两种情况进行了试验：一种模拟轻负载情况；另一种模拟重负载情况，试验结果如图 4 和图 5 所示。

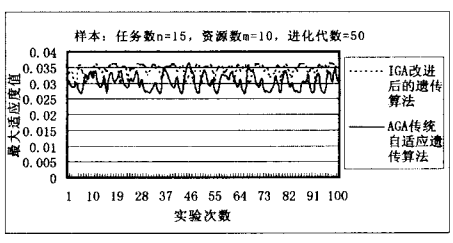


图 4 轻负载(任务数 $n=15$，资源数 $m=10$)的情况

由上述仿真实验结果分析可知，改进后的算法既具有较好的全局收敛能力，又有较快的收敛速度，能为网格任务调度节约时间，且改进后的算法对于规模大的网格任务调度效果更好，所以实现了本文以网格任务调度最优跨度为目标的实验目的。

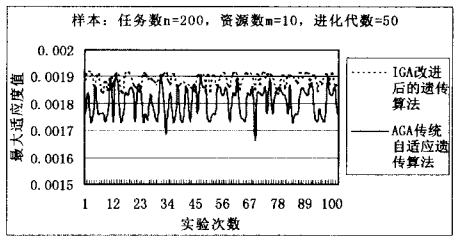


图 5 重负载(任务数 $n=200$，资源数 $m=10$)的情况

结束语 本文提出了一种基于改进遗传算法的网格任务调度算法，改进后的算法产生的初始种群既可以保证种群的平均适应度值较高，又由于有随机产生的个体而可以保证种群的多样性。算法迭代过程中采用了一种局部收敛判断的新标准以及改进的变异操作来防止局部收敛。仿真结果表明，该改进算法具有良好的效率和收敛性，能更有效地解决网格任务调度问题。

参 考 文 献

[1] Abrahan A, Buyya R, Nath B. Nature's heuristics for scheduling jobs on computational grids[C]// The 8th IEEE International Conference on Advance Computing and Communications India. 2000

[2] Karonls NT, Tonnen B, Foster I. MPICH-G2: A Grid Enabled Implementation of the Message Passing Interface[J]. Journal of Parallel(JPDC), 2003, 163(5): 551-563

[3] Foster I, Kesselman M. The Grid: Blueprint for a Future Computing Infrastructure[M]. USA: Morgan Kaufmann Publishers, 1999

[4] Beman F, Fox G, Tony H. Grid Computing-making the Global Infrastructure a Reality[M]. USA: John Wiley and Sons Ltd, 2003: 65-100

[5] Spooner D P, Jarvis S A. Local Grid Scheduling Techniques using Performance Predition[J]. IEEE Proceedings-Computers and Digital Techniques, 2003, 150(2): 87-96

[6] 杨博，陈志刚. 一种基于双层进化结构的网格任务调度算法[J]. 计算机工程与应用，2006，42(15)：8-10

[7] 都志辉，陈渝，刘鹏. 网格计算[M]. 北京：清华大学出版社，2002

[8] 王小平，曹立明. 遗传算法——理论、应用与软件实现[M]. 西安：西安交通大学出版社，2002

[9] 袁禄来，曾国荪，姜黎立，等. 网格环境下基于信任模型的动态级调度[J]. 计算机学报，2006，29(7)：1217-1224

[10] 吴高锋，蒋玉明，等. 基于 QoS 改进的 Min-Min 网格调度算法[J]. 微计算机信息，2009，3-9：110-112

[11] 李敏，吴浪，张开碧. 求解旅行商问题的几种算法的比较研究[J]. 重庆邮电大学学报：自然科学版，2008，20(5)：624-626

(上接第 155 页)

[10] Sam Y, Boucelma O. Customizable Web Services Description, Discovery and Composition: An Attribute Based Formalism[C]// Proc. of 2007 IEEE/WIC/ACM International Conference on Web Intelligence. 2007

[11] Liang Q, Miller S, Chung J. Service Mining for Web Service Composition[C]// Proc. of IEEE International Conference on Information Reuse and Integration (IEEE IRI-2005). 2005

[12] Gaaloul W, Bhiri S, Godart C. Research Challenges and Opportunities In Web Services Mining[C]// Proc. of System and Information Service Web, INFORSID2006. 2006

[13] Zhuge H. The Knowledge Grid Environment, IEEE Intelligent [J]. Systems, 2008, 23(6): 63-71

[14] 代宇，刘宴兵，程瑶. 基于异步 Web Service 调用的 Web 应用程序研究[J]. 重庆邮电大学学报：自然科学版，2008，20(6)：746-748