

## 基于社团服务链的 Web 服务组合方法

何丽\*, 赵富强, 饶俊

(天津财经大学 信息科学与技术系, 天津 300222)

(\* 通信作者电子邮箱 renke21@vip.sina.com)

**摘要:**针对 Web 服务组合的时间效率提高问题,提出了一种基于服务社团和服务链的 Web 服务组合方法。在构造的服务网络上应用基于信息中心度的服务社团发现方法,将 Web 服务网络划分为不同的服务社团,然后构造了社团服务链发现算法和基于服务链的 Web 服务组合算法,这些算法将服务社团内 Web 服务之间的所有可组合关联转变成服务链,实现了基于社团服务链和服务质量(QoS)剪枝的 Web 服务组合过程。实验结果表明,与传统的图深度遍历 Web 服务组合方法相比,基于社团服务链的 Web 服务组合方法在 5 个测试集上的响应时间平均提高了 46%,最好情况为 67%。社团服务链可以有效地减少针对当前服务请求的服务搜索空间,提高服务组合的时间效率。

**关键词:** Web 服务;服务社团;服务链;服务组合;服务质量

**中图分类号:** TP311.11 **文献标志码:** A

### Web service composition method based on community service chain

HE Li\*, ZHAO Fuqiang, RAO Jun

(Department of Information Science and Technology, Tianjin University of Finance and Economics, Tianjin 300222, China)

**Abstract:** A new Web service composition method based on service communities and service chains was proposed in this paper to improve the time efficiency of service composition. In the method, a service network was constructed for the Web service collection, the service community discovery algorithm based on information center was applied to find service clubs in the service network, and then the community service chain discovery algorithm and Web service composition algorithm based on service chain were built. With these algorithms, all of service interface associations in a service club were changed into service chains, and the Web service composition process based on community service chains and Quality of Service (QoS) pruning was implemented. The experimental results indicate that, compared with the traditional service composition method based on graph depth traversal, the response time on five test sets in the service composition method with community service chains is on average improved by 42%, and up to 67%. Community service chains can effectively reduce the service search space for the current service request and improve the time efficiency of service composition.

**Key words:** Web service; service community; service chain; service composition; Quality of Service (QoS)

## 0 引言

为满足日益复杂的服务请求,将多个 Web 服务按照一定的逻辑关系组合起来,形成功能更强大的组合服务,这一过程称为 Web 服务组合。针对一个服务请求,如何从庞大的服务网络中快速地找到满足用户要求的 Web 服务是实现服务组合的关键。目前,基于服务质量(Quality of Service, QoS)的服务组合优化是 Web 服务组合的研究热点,文献[1]提出了一种基于凸包构建的组合服务优化算法,用以解决 QoS 感知的服务组合优化问题;文献[2-5]提出了基于 QoS 模型的动态 Web 服务组合方法;文献[6]建立了针对 Web 服务组合过程的动态 QoS 计算模型。以上这些方法验证了基于 QoS 剪枝的 Web 服务组合的有效性。

Web 服务组合反映了可组合 Web 服务之间的信息传递关系以及它们在功能接口上的逻辑关联,根据这种逻辑关联的紧密程度,可以将 Web 服务网络划分成不同的服务社团。文献[7]提出了基于 Web 服务匹配关系的复杂网络构建方法;文献[8]提出了基于 Web 服务组链的服务组合方法,它利用服务组将并发服务组合转换成顺序服务组合,提高了服务

组合的时间效率;文献[9]提出了基于谱聚类的 Web 服务社团发现方法;文献[10]提出了基于社会网络的 Web 服务选择方法;文献[11]提出了基于元 Web 服务的分布式 Web 服务组合算法,该算法通过为服务组合建立有效的索引机制,减少了无效服务组合的分析次数,提高了服务组合的时间效率。以上文献探讨了基于服务组的 Web 服务发现和组合方法的有效性。为使用 Web 服务网络的社团结构来提高 Web 服务组合的时间效率,本文提出了一种基于社团服务链的 Web 服务组合方法,该方法应用社会网络原理,将 Web 服务网络划分成若干个服务社团,并通过社团服务链发现算法,将服务社团内 Web 服务之间的所有可组合关联用服务链表示,使针对当前服务请求的服务组合过程转变为服务链的匹配和组合过程,并在服务链组合过程中应用 QoS 剪枝技术来进一步减少服务组合过程的问题搜索空间,实验证明能够获得较好的服务组合时间效率。

## 1 Web 服务社团

**定义 1** Web 服务。Web 服务是由服务提供者提供的,可以被独立调度和执行的软件。Web 服务描述为  $ws = \langle I, O \rangle$ ,

收稿日期:2012-08-06;修回日期:2012-09-06。 基金项目:天津市高等学校科技发展基金资助项目(20110819)。

作者简介:何丽(1969-),女,安徽舒城人,副教授,博士,主要研究方向:Web 服务、Web 数据挖掘; 赵富强(1974-),男,河北涉县人,讲师,博士,主要研究方向:社会网络、顾客满意度; 饶俊(1979-),男,江西上饶人,讲师,博士,主要研究方向:数据挖掘。

$Q$ ),  $I$  是  $ws$  的输入,  $O$  是  $ws$  的输出,  $Q$  是  $ws$  的 QoS 属性向量, 它包括了  $ws$  的运行费用, 运行时间和可靠性等级三个质量属性<sup>[6]</sup>。

**定义 2** Web 服务接口关联。对 Web 服务集合中的任意两个服务  $ws_i$  和  $ws_j$ , 若  $ws_i.O \cap ws_j.I \neq \emptyset$  或  $ws_i.O \cap ws_j.I \neq \emptyset$ , 则称  $ws_i$  和  $ws_j$  接口关联。

Web 服务网络用有向图  $G$  表示, 服务集合中的每个 Web 服务表示为  $G$  中的一个顶点, 两个 Web 服务之间的接口关联用  $G$  中对应顶点之间的有向边表示。服务集合  $S$  中的两个 Web 服务  $ws_i$  和  $ws_j$ , 若  $ws_i.O \cap ws_j.I \neq \emptyset$ , 则建立一条从顶点  $ws_i$  指向顶点  $ws_j$  的有向边; 若  $ws_j.O \cap ws_i.I \neq \emptyset$ , 则建立一条从顶点  $ws_j$  指向顶点  $ws_i$  的有向边。根据定义 2 对  $S$  中任意两个 Web 服务进行接口关联判断, 并为接口关联的顶点之间建立有向边就形成了  $S$  的服务网络。服务网络的规模越大, 其社团结构将越明显<sup>[7]</sup>。应用复杂网络划分方法, 可以将一个服务网络划分成若干个服务社团, 使接口关联紧密的 Web 服务聚集在同一服务社团, 不同服务社团的 Web 服务之间接口关联稀疏。

**定义 3** Web 服务社团。Web 服务社团由具有紧密接口关联的 Web 服务及其接口关联组成, 描述为  $wsc = \langle SS, M \rangle$ 。SS 是  $wsc$  的 Web 服务集合;  $M$  是邻接矩阵, 它描述了 SS 中所有 Web 服务之间的接口关联。

GN (Girvan and Newman) 是一种常用的社团发现算法, 本文使用基于信息中心度的 GN 算法来发现服务网络中的社团。该方法使用边的信息中心度来代替 GN 算法中的边介数, 并使用网络有效率来衡量网络节点传输信息的有效性。对一个由  $N$  个节点和  $M$  条边构成的网络  $G$ , 假设节点之间的信息总是沿着最短路径传播, 则网络节点之间的信息传输有效率为它们之间最短路径的倒数。设  $d_{ij}$  为  $G$  中任意两个节点  $i$  和  $j$  之间的最短路径, 若节点  $i$  和  $j$  之间不存在路径, 令  $d_{ij}$  的倒数为 0, 则  $G$  的网络有效率<sup>[12]</sup> 定义为:

$$E(G) = \frac{\sum_{i \neq j \in G} \frac{1}{d_{ij}}}{N(N-1)}$$

边  $k$  的信息中心度  $C_k$  定义为移出该边后整个网络信息有效率减小的相对量, 即:

$$C_k = \frac{E(G) - E(G')}{E(G)}$$

其中  $E(G')$  是移出边  $k$  后, 由  $G$  中的  $N$  个节点和剩余  $M-1$  条边构成网络的信息有效率。

对一个服务网络  $G$ , 基于信息中心度的社团发现过程分为 4 个主要步骤:

- 1) 计算  $G$  中每条边的信息中心度;
- 2) 选取信息中心度最大的边  $k$ , 将它从  $G$  中移除;
- 3) 重新计算每条边的信息中心度;
- 4) 返回步骤 2) 循环迭代, 直到每个节点都被划分到一个服务社团中, 然后返回  $G$  的服务社团集合  $WSC = \{wsc_1, wsc_2, \dots, wsc_K\}$ ,  $K$  为服务社团个数。

为实现基于社团服务链的 Web 服务组合, 对  $WSC$  中的任意两个服务社团  $wsc_a$  和  $wsc_b$ , 建立服务社团关联矩阵  $M_{ab}$ , 即对任意  $ws_i \in wsc_a$ . SS 和任意  $ws_j \in wsc_b$ . SS 的两个 Web 服务, 若  $ws_i$  和  $ws_j$  是接口关联的, 则令  $M_{ab}[i, j] = 1$ ; 否则令  $M_{ab}[i, j] = 0$ 。

## 2 基于社团服务链的 Web 服务组合

### 2.1 社团服务链发现算法

**定义 4** 服务链。服务链描述了 Web 服务的组合顺序, 描述为  $sc = \langle ws_h, ws_t, sl \rangle$ 。其中:  $ws_h$  和  $ws_t$  分别表示  $sc$  的头服务和尾服务,  $sl$  是  $k(k \geq 1)$  个元素的顺序组合, 即  $sc.sl = e_1 \Theta e_2 \Theta \dots \Theta e_k, e_i (1 \leq i \leq k)$  表示一个组合元素。

服务链中的组合元素可以是一个串行服务链或并行服务链。接口关联的两个 Web 服务  $ws_i$  和  $ws_j$  顺序组合后形成一个串行服务链  $sc = ws_i \Theta ws_j$ , 顺序组合用符号“ $\Theta$ ”表示; 对头服务和尾服务相同的两个或多个串行服务链进行并行组合, 形成一个并行服务链, 并行组合用符号“ $\parallel$ ”表示。例如, 若有服务链  $ws_m \Theta ws_i \Theta ws_n$  和  $ws_m \Theta ws_j \Theta ws_n$ , 并行组合后形成服务链  $ws_m \Theta ws_i \parallel ws_j \Theta ws_n$ 。服务链  $sc$  可以看作是一个功能复杂的 Web 服务,  $sc$  的输入  $In(sc) = ws_h.I$ ,  $sc$  的输出  $Out(sc) = \bigcup_{i=1}^k Out(e_i)$ , 函数  $In(sc)$  和  $Out(sc)$  分别用来返回  $sc$  的输入和输出。

**定义 5** 空服务。为规范服务链的并行组合过程, 引入了空服务。空服务是一个虚拟服务, 约定其输入为空, 其输出能满足所有 Web 服务的接口关联要求, 并忽略其 QoS 属性。引入空服务后, 可以将多个尾服务相同而头服务不同的服务链合并成一个并行服务链。

图 1 中的  $ws_0$  是一个空服务, 引入  $ws_0$  后, 可以将串行服务链  $ws_1 \Theta ws_3$  和  $ws_2 \Theta ws_3$  合并成并行服务链  $ws_0 \Theta ws_1 \parallel ws_2 \Theta ws_3$ 。

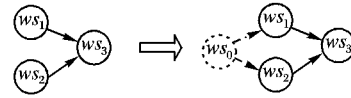


图 1 空服务

社团服务链可以看作是社团内服务组合的索引, 它描述了一个社团内所有 Web 服务之间的可组合关联。访问服务社团邻接矩阵, 对接口关联的任意两个 Web 服务进行顺序组合可以形成一个串行服务链, 然后依次对这些服务链进行顺序组合或并行组合, 就可以获得一个服务社团的服务链集合。算法描述如下。

**算法 1** 社团服务链发现算法。

输入 服务社团  $wsc = \langle SS, M \rangle$ 。

输出  $wsc$  的服务链集合  $SCS$ 。

步骤如下:

- 1) FOR ( $wsc$  中的任意两个 Web 服务  $ws_i, ws_j$ ) DO  
IF ( $wsc.M[i, j] = 1$ ) THEN  
① 串行组合  $ws_i$  和  $ws_j$  形成服务链  $sc, sc.sl = ws_i \Theta ws_j, sc.ws_h = ws_i, sc.ws_t = ws_j$ ;  
②  $SCS = SCS \cup \{sc\}$ ; END IF  
END FOR
- 2) FOR ( $SCS$  中的任意两个服务链  $sc_i$  和  $sc_j$ ) DO  
IF ( $sc_i.ws_t = sc_j.ws_h$ ) THEN  
① 串行组合  $sc_i$  和  $sc_j$  形成服务链  $sc: sc.ws_h = sc_i.ws_h, sc.ws_t = sc_j.ws_t, sc.sl = sc_i \Theta sc_j$ ;  
②  $SCS = SCS \cup \{sc\}$ , 并从  $SCS$  中删除  $sc_i$  和  $sc_j$ ; END IF  
END FOR
- 3) 查找  $SCS$  中具有相同尾服务的服务链, 对它们进行并行组合。  
若存在  $k(k \geq 2)$  个这样的服务链  $sc_1, sc_2, \dots, sc_k$ , 则:  
①  $ws_t = Deletetail(sc_1, sc_2, \dots, sc_k)$ ;  
// 删除  $sc_1, sc_2, \dots, sc_k$  的尾服务

②IF (  $sc_1, sc_2, \dots, sc_k$  的头服务相同 ) THEN  
 $ws_h = \text{Deletehead}(sc_1, sc_2, \dots, sc_k);$   
 // 删除  $sc_1, sc_2, \dots, sc_k$  的头服务  
 ELSE  $ws_h = ws_0;$  //  $ws_0$  为空服务  
 ③并行组合  $sc_1, sc_2, \dots, sc_k$  形成  $sc, sc.sl =$   
 $ws_h \Theta sc_1 \parallel sc_2 \parallel \dots \parallel sc_k \Theta ws_i, sc.ws_h = ws_h, sc.ws_i = ws_i;$   
 ④ $SCS = SCS \cup \{sc\}$ , 并从  $SCS$  删除  $sc_1, sc_2, \dots, sc_k;$   
 4) 循环执行步骤 2) 和 3), 直到找不到任意两个满足条件的服务链。

## 2.2 基于社团服务链的 Web 服务组合算法

**定义 6** 服务请求。带有 QoS 需求的服务请求描述为  $wsr = \langle I, O, Q \rangle$ , 其中:  $I$  和  $O$  分别表示  $wsr$  的输入和输出要求,  $Q$  描述用户的 QoS 需求。

对一个服务链  $sc$ , 若  $wsr.I \supseteq In(sc)$ , 则称  $sc$  和  $wsr$  输入匹配; 若  $wsr.O \subseteq Out(sc)$ , 则称  $sc$  和  $wsr$  输出匹配。为返回满足用户 QoS 需求的服务组合, 需要在组合过程中计算各个候选服务链的 QoS 属性值。与 Web 服务一样, 服务链的 QoS 属性也包括运行费用  $C$ 、运行时间  $T$  和可靠性等级  $R$  三个质量属性, 各个属性值的计算方法如下。

对一个串行组合  $sc = e_1 \Theta e_2 \Theta \dots \Theta e_k$ , 它的各个 QoS 值计算如式(1):

$$\begin{cases} C_{sc} = \sum_{i=1}^k C_{e_i} \\ T_{sc} = \sum_{i=1}^k T_{e_i} \\ R_{sc} = \min(R_{e_1}, R_{e_2}, \dots, R_{e_k}) \end{cases} \quad (1)$$

对一个并行组合  $sc = e_1 \parallel e_2 \parallel \dots \parallel e_k$ , 它的各个 QoS 值计算如式(2):

$$\begin{cases} C_{sc} = \max(C_{e_1}, \dots, C_{e_k}) \\ T_{sc} = \max(T_{e_1}, \dots, T_{e_k}) \\ R_{sc} = \min(R_{e_1}, R_{e_2}, \dots, R_{e_k}) \end{cases} \quad (2)$$

**定义 7** 服务请求与服务链的 QoS 匹配。对服务请求  $wsr$  和服务链  $sc$ , 若  $C_{sc} \leq C_{wsr}, T_{sc} \leq T_{wsr}, R_{sc} \geq R_{wsr}$  同时满足, 则称  $wsr$  与  $sc$  QoS 匹配, 记为  $wsr < sc$ ; 否则称  $wsr$  与  $sc$  QoS 不匹配, 记为  $wsr > sc$ 。

对一个服务请求  $wsr$ , 算法 2 描述了基于社团服务链和 QoS 剪枝的 Web 服务组合过程。

**算法 2** 基于社团服务链集合的 Web 服务组合算法。

输入  $K$  个社团服务链集合  $SCS_K = \{SCS_1, SCS_2, \dots, SCS_K\}$ , 社团关联矩阵  $M$  和服务请求  $wsr$ 。

输出 满足  $wsr$  要求的 Web 服务组合集合  $Result$ 。

步骤如下:

```

1) Result = NULL;
2) FOR (1 ≤ i ≤ K) DO
  FOR (SCSi 中每个与 wsr 输入匹配的服务链 sc) DO
    ①根据式(1) ~ (2) 计算 sc 的 QoS 值 Csc, Tsc 和 Rsc;
    ②IF (wsr < sc) THEN
      FOR each wsk in sc DO // k ≥ 1
        a) sc' = subchain(sc, k);
          // 截取 sc 前 k 个服务形成子链 sc'
        b) IF (sc' 满足 wsr 的输出要求) THEN
          Result = Result ∪ {sc'}; RETURN;
        ELSE k++;
      END IF
    END FOR
  ③ELSE

```

```

a) i = wsc_number(WSC, wsk); // 求 wsk 的服务社团序号
b) FOR (j = 1 to K, j ≠ i) DO
  m = number(wscj, wsk); // 求 wsk 在 wscj 中的编号
  FOR each wsn in wscj DO
    n = number(wscj, wsn);
    IF (Mj[m, n] = 1) THEN
      sc' = subchain(sc, k);
      对 SCSj 中头服务为 wsn 的服务链 sc'', 令
      sc = sc' Θ sc'', GOTO ②;
    END IF
  END FOR
END FOR
END IF
END FOR
END IF
END FOR
END FOR

```

## 3 实验

### 3.1 实验环境与实验数据

为仿真服务网络的社团结构, 本文在三个概念集  $C1, C2, C3 (C1 \cap C2 \cap C3 = \emptyset)$  上分别生成 50 个 Web 服务的输入和输出, 在  $C1 \cup C2, C2 \cup C3$  和  $C1 \cup C3$  上分别生成 10 个 Web 服务的输出和输出, Web 服务的输入和输出参数个数最多不超过 3 个, 然后分别在区间  $[0, 10], [1.0, 5.0]$  和集合  $\{1, 2, 3\}$  上随机生成每个 Web 服务的运行费用、运行时间和可靠性等级。实验以生成的 180 个 Web 服务建立样本集 Dset, 然后分别从 Dset 中随机抽取 30, 60, 90, 120 和 150 个 Web 服务建立 5 个测试集, 分别记为 DS1, DS2, DS3, DS4 和 DS5。实验运行环境为 CPU 3.1 GHz, 主存 4 GB, 操作系统为 Windows XP。

### 3.2 实验设计与结果分析

**实验 1** 为验证本文构造的 Web 服务网络的社团结构特征, 在 R 环境下运行 IGraph 软件包, 然后在上述 5 个测试集上分别运行基于信息中心度的 GN 网络划分算法, 并使用模块度作为网络划分质量的评价标准, 得到 5 个测试集 GN 划分的模块度值如表 1 所示。

表 1 服务网络 GN 划分的模块度

测试集	边数	模块度 $Q$	测试集	边数	模块度 $Q$
DS1	63	0.5122	DS4	328	0.6378
DS2	117	0.6054	DS5	409	0.4901
DS3	149	0.5158			

模块度  $Q$  是一种常用的网络划分质量评价标准,  $Q$  值的上限为 1,  $Q$  值越大, 表明网络的社团结构越明显, 实际网络的  $Q$  值通常在  $0.3 \sim 0.7^{[12]}$ 。表 1 表明, 本文构造的 5 个测试集网络划分的模块度值均在实际网络的  $Q$  值范围内, 具有较好的社团结构特征。

**实验 2** 为验证本文方法的服务组合时间效率, 本实验使用服务组合响应时间作为评价标准。对一个服务请求  $wsr$ , 其服务组合响应时间定义为从提交  $wsr$  到返回服务组合结果这一段时间间隔, 以秒为单位。 $wsr$  的输入和输出在  $C1 \cup C2 \cup C3$  上随机生成,  $wsr$  的 QoS 生成方法与上述 Web 服务的 QoS 生成方法相同。实验首先以 3 个服务请求的平均服务组合响应时间为比较对象, 服务请求的输入/输出参数个数设置为 3, 然后分别在上述 5 个测试集上运行图深度遍历服务组合算法和本文算法, 对照结果如图 2 所示。

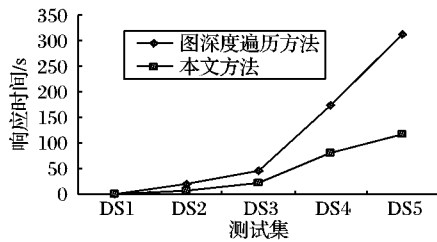


图2 服务组合响应时间比较

从图2可以看出,本文方法能够获得更好的服务组合响应时间,并且,随着测试集规模的增大,本文方法在服务组合响应时间上的优势更加明显。在DS1上,针对3个服务请求服务组合的平均响应时间提高了7%;而在DS5上,平均响应时间提高了67%。这是因为社团服务链的引入,能够将当前服务请求准确地定位到少数关联的服务社团中,从而减少了针对当前服务请求的Web服务搜索和接口匹配的时间开销,缩短了服务组合的响应时间。

为说明本文方法在服务组合时间效率上的稳定性,本实验还设计了五类服务请求,它们的输入/输出参数个数分别被设置为2,3,5,7和9,并使用上述服务请求的生成方法为每类服务请求生成3个实例,然后选择在模块度最好的测试集DS4上分别运行图深度遍历服务组合算法和本文算法,图3描述了针对各类服务请求的服务组合响应时间变化情况。

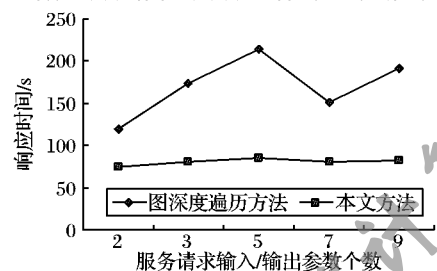


图3 服务请求参数个数对组合响应时间的影响

在图3中,图深度遍历算法的服务组合响应时间对服务请求输入/输出参数个数的变化比较敏感,而本文算法的响应时间变化幅度相对较小。在本文算法中,当测试集的社团结构较好时,因为跨社团的服务链组合相对较少,使针对服务请求的组合过程能够局部在少数社团服务链集合,并保持了服务组合响应时间的相对稳定;而在图深度遍历算法中,与服务请求接口匹配的Web服务数量会随着其输入/输出参数个数的变化而变化,从而导致了服务组合响应时间的大幅度变化。

## 4 结语

本文将复杂网络的原理应用到Web服务组合算法中,根据Web服务的接口关联,建立Web服务网络,并对Web服务网络进行社团划分,使得接口关联的Web服务聚集到同一个服务社团中,并为服务社团建立服务链,实现了基于社团服务链的动态Web服务组合,并在服务组合过程中应用QoS剪枝,有效地减少了针对当前服务请求的问题搜索空间,获得了较好的服务组合响应时间。

### 参考文献:

- [1] 李俊,郑小林,陈松涛,等.一种高效的服务组合优化算法[J].中国科学:信息科学,2012,42(3):280-289.
- [2] 杨怀洲,李增智.QoS敏感的服务组合动态配置研究[J].西安交通大学学报,2010,44(2):25-30.
- [3] 孔维梁,刘清堂,杨宗凯,等.基于动态QoS的Web服务组合[J].计算机科学,2012,39(2):268-272.
- [4] PASTRANA J L, PIMENTEL E, KATRIB M. QoS-enabled and self-adaptive connectors for Web services composition and coordination[J]. Computer Languages, Systems and Structures, 2011, 37(1): 2-23.
- [5] HWANG S Y, LIM E P, LEE C H, et al. Dynamic Web service selection for reliable Web service composition[J]. IEEE Transactions on Services Computing, 2008, 1(2): 104-116.
- [6] 张琦,侯红.Web服务动态组合中QoS计算方法研究[J].计算机工程,2011,37(12):41-43.
- [7] 朱志良,邱媛源,李丹程,等.一种Web服务复杂网络的构建方法[J].小型微型计算机系统,2012,33(2):199-205.
- [8] 王佳强,刘大有,李嘉菲,等.基于服务组链的Web服务组合方法[J].吉林大学学报:理学版,2009,47(1):148-154.
- [9] ZHANG X Z, YIN Y, ZHANG M W, et al. Web service community discovery based on spectrum clustering[C]// CIS09: Proceedings of the 2009 International Conference on Computational Intelligence and Security, Washington, DC: IEEE Computer Society, 2009: 187-191.
- [10] 苏佰川,张国义,许胤龙.基于社会网络的Web服务选择算法的研究[J].微型机与应用,2012,31(6):46-49.
- [11] 雷万保,朱怡安,钟冬.基于元Web服务的分布式Web服务组合算法[J].华中科技大学学报:自然科学版,2010,38(10):18-21.
- [12] FORTUNATO S, LATORA V, MARCHIORI M. A method to find community structures based on information centrality[J]. Physical Review Letters, 2004, 70(5): 1-13.

(上接第249页)

- [25] ZARANDI H R, MAGHSOUDLOO M, KHOSHAVI N. Two efficient software techniques to detect and correct control-flow errors[C]// Proceedings of the 16th Pacific Rim International Symposium on Dependable Computing. Washington, DC: IEEE Computer Society, 2010: 141-148.
- [26] ARLAT J, FABRE J C, RODRÍGUEZ M, et al. Dependability of COTS microkernel-based systems[J]. IEEE Transactions on Computers, 2002, 51(2): 138-163.
- [27] GU W N, KALBARCZYK Z, IYER R K. Error sensitivity of the Linux kernel executing on PowerPC G4 and Pentium4 processors[C]// Proceedings of the 2004 International Conference on Dependable Systems and Networks. Washington, DC: IEEE Computer Society, 2004: 887-896.
- [28] ALBINET A, ARLAT J, FABRE J C. Characterization of the im-

part of faulty drivers on the robustness of the Linux kernel[C]// Proceedings of the 2004 International Conference on Dependable Systems and Networks. Washington, DC: IEEE Computer Society, 2004: 867-876.

- [29] DARAN M, THEVENOD-FOSSE P. Software error analysis a real case study involving real faults and mutations[C]// Proceedings of the 1996 ACM SIGSOFT International Symposium on Software Testing and Analysis. New York: ACM Press, 1996: 158-171.
- [30] DURAES J A, MADEIRA H S. Emulation of software fault: a field data study and a practical approach[J]. IEEE Transactions on Software Engineering, 2006, 32(11): 849-867.
- [31] AUGUSTON M. Software architecture built from behavior models[J]. ACM SIGSOFT Software Engineering Notes, 2009: 34(5): 1-15.