第 34 卷增刊 系统工程理论与实践 Vol.34, Suppl.
2014 年 6 月 Systems Engineering — Theory & Practice June, 2014

文章编号: 1000-6788(2014)S0-0131-12 中图分类号: TP393 文献标志码: A

基于服务网络的服务组合推荐方法

潘伟丰[1,2], 李 兵[2,3], 姜 波[1], 琚春华[1]

(1. 浙江工商大学 计算机与信息工程学院, 杭州 310018; 2. 武汉大学 软件工程国家重点实验室, 武汉 430072;
3. 武汉大学 计算机学院, 武汉 430072)

摘 要 服务组合是服务计算领域的研究热点. 针对现有服务组合方法主要是针对 web 服务提出来的, 过分依赖 web 服务 WSDL (Web Service Description Language) 描述文档的不足, 提出了一种基于服务网络的服务组合推荐方法, 为实现不具有 WSDL 文档的服务的组合问题提供了一种解决方案. 该方法: 基于服务组合历史, 构建服务网络模型, 抽象服务间的协作和竞争关系; 利用复杂网络方法挖掘服务使用模式; 提出了基于服务使用场景的服务组合推荐算法. 使用 ProgrammableWeb 上 API 服务和 mashup 应用的真实数据来说明本文方法的可行性和有效性. 从实验结果可以看出, 本文提出的方法可以弥补同类方法的不足, 为解决服务组合相关问题提供了一种新思路.

关键词 服务组合; 服务网络; 复杂网络; 服务计算

Service composition recommendation based on service networks

PAN Wei-feng[1,2], LI Bing[2,3], JIANG Bo[1], JU Chun-hua[1]

(1. School of Computer Science and Information Engineering, Zhejiang Gongshang University, Hangzhou 310018, China;
2. State Key Laboratory of Software Engineering, Wuhan University, Wuhan 430072, China; 3. School of Computer, Wuhan University, Wuhan 430072, China)

Abstract Service composition is a hot research field of service computing. Considering the deficiencies of existing service composition methods, such as they are mainly presented for web services and overly rely on the information in WSDL (Web Service Description Language) document, this paper presents a new service composition recommendation method, which is based on service composition histories, providing a new line for solving the service composition problem of services without a WSDL document. It utilizes the service composition histories and proposes service networks to abstract the collaboration and competition relationships between services; it uses complex network theory to mine the service usage patterns; it analyzes the service usage scenarios and finally proposes an algorithm for service recommendation. API services and mashup applications in ProgrammableWeb are used as subjects to demonstrate the feasibility of the proposed approach. Experimental results show that the proposed method can compensate for the lack of similar methods and provide a new way to solve the related problems of the service composition.

Keywords service composition; service network; complex network; service computing

1 引言

面向服务的计算 (Service-Oriented Computing, 简称 SOC) 是针对分布式系统的新型计算模式, 其核心思想是将服务作为基本构建模块, 创造可以跨越组织边界和计算平台的动态业务流程和敏捷应用, 已成为软件领域最热门的话题之一 [1]. 大多数功能相对单一的服务是无法满足用户需求的, 必须将共享的服务组合起

收稿日期: 2013-12-25
资助项目: 国家自然科学基金 (61202048); 浙江省自然科学基金 (LQ12F02011); 软件工程国家重点实验室开放基金 (SKLSE-2012-09-21); 浙江省电子商务技术重点科技创新团队 (2010R50041)
作者简介: 潘伟丰 (1982–), 男, 汉, 浙江杭州人, 博士, 副教授, 硕士生导师, 研究方向: 服务计算、软件工程、复杂网络和智能计算, E-mail: panweifeng1982@gmail.com; 李兵 (1969–), 男, 汉, 湖北武汉人, 教授, 博士生导师, 研究方向: 服务计算、云计算和社会计算, E-mail: bingli@whu.edu.cn; 姜波 (1970–), 女, 汉, 浙江黄岩人, 教授, 硕士生导师, 研究方向: 服务计算与社会计算, E-mail: nancybjiang@mail.zjgsu.edu.cn; 琚春华 (1962–), 男, 汉, 浙江常山人, 教授, 博士生导师, 研究方向: 服务计算与社会计算, E-mail: jch@mail.zjgsu.edu.cn.

来才能达到目的 [2]. 因此, 如何将多个服务组合起来形成功能更为强大的服务 (复合服务), 就成为目前服务计算领域重要的研究方向和研究热点 [3].

人们针对服务组合问题开展了大量的研究工作, 根据服务组合的理论或技术基础大致可以分为 3 类 [4]: 1) 基于工作流技术的服务组合方法: 利用服务组合和工作流模型的相似性, 通过工作流建模工具对组合流程建模, 再将流程任务用服务绑定实现服务组合, 如文献 [5]; 2) 基于 AI 规划技术的服务组合方法: 将服务的输入/输出参数、前提和结果等描述映射为动作的形式化描述, 通过推理得出服务的组合序列, 生成服务组合方案, 如文献 [6]; 3) 基于语义的服务组合方法: 从语义角度研究服务与需求的匹配以及服务和服务之间的可组合性, 如文献 [7]. 上述方法虽然取得了一定的研究成果, 但是仍有如下不足: 现有的方法大多是针对 web 服务的, 服务组合都依赖于服务 WSDL (Web Service Description Language, WSDL[1]) 描述文档中的信息. 但是, 随着服务的概念和相关技术与 Web 2.0、云计算等新兴计算模式的融合, 在传统 Web 服务基础上又出现了多种不同类型的服务, 如 REST 风格的 Web 服务、开放 API (Open API) 和 mashup 等, 它们并不存在 WSDL 描述文档. 因此, 如何实现像开放 API 这种没有 WSDL 文档的服务的自动组合就成为服务组合领域的一个新问题.

我们曾在文献 [8] 提出了一种基于服务使用历史的服务组合推荐方法 LFH (Learn from History): 根据服务使用历史, 构建服务及其组合方案的二分网络, 通过对二分网络的投影获得服务之间的协作网络 (节点代表服务, 边代表服务间的协作关系, 边上的权值量度服务间协作的可能性), 最终基于服务协作网络为用户推荐服务的组合方案. 该方法克服了先前服务组合方法对 WSDL 文档的依赖, 适用于像开放 API 之类服务的组合问题, 为服务组合提供了一种新思路. 但是 LFH 方法存在两点不足: 1) LFH 方法不能为没有历史使用记录的服务推荐组合方案: LFH 方法构建的服务协作网络仅包括被使用过的服务, 未使用的服务不在网络中, 可推荐的服务数量少; 2) LFH 方法构建的网络主要考虑的是服务间的协作关系, 但是服务间除了协作还存在竞争关系 [9], 即两个服务在功能上相似, 当一个服务失效时可用另一个替换. LFH 方法构建的服务协作网络并未考虑服务间的这种竞争关系. 因此, 当一个服务失效时, LFH 方法不能推荐可替代的服务组合方案.

针对上述问题, 本文提出了一种基于服务网络的服务组合推荐方法 SCRSN (Service Composition Recommendation based on Service Networks). 该方法首先根据服务的使用历史和服务间的相似关系构建服务网络, 用于抽象服务及服务间的竞争/协作关系; 然后引入复杂网络原理, 分析服务的使用模式; 最后基于服务的使用模式及服务网络, 针对服务的不同使用情景, 提出了服务组合的推荐算法, 为用户推荐服务组合方案.

本文后续内容的组织结构如下: 第 2 节将详细地介绍 SCRSN 方法, 包括服务网络的构建、服务使用模式的分析及服务推荐算法等; 第 3 节将以 ProgrammableWeb 上的 mashup、API 服务的真实数据验证 SCRSN 方法的可行性和有效性; 第 4 节介绍本文方法的应用平台; 最后是结论与展望.

2 SCRSN 方法

随着体系结构研究的深入, 人们逐渐意识到系统的结构是决定系统质量的重要因素 [10]. 同时, 从面向结构的程序设计到面向对象/构件程序设计, 再到面向服务的软件开发方式, 软件的设计和实现的重点已从实现局部的编程难题, 转向了如何将代码有效地进行组织. 特别是对于大规模复杂软件系统而言, 结构的组织就尤为重要. SCRSN 方法基于服务使用历史及对服务相似性的量度, 构造服务网络从结构角度抽象服务及服务间的竞争/协作关系, 并进而指导服务组合方案的推荐. 图 1 给出了 SCRSN 方法的基本框架. 以下各小节将对框架中的部分内容进行说明, 并给出相关概念的定义.

2.1 数据收集

随着 SOC 研究与应用的逐步展开, 服务数量和种类 (如 web 服务、REST 风格的 Web 服务、开放 API 等) 的增长速度也将不断加快 [11], 以服务为中心的互联网正在形成. 在这样一个海量的集合上, 进行服务发现、选择, 并进而通过服务组合满足用户的需求, 其难度在不断增加. 服务注册库是服务信息组织的核心, 为服务开发者和消费者提供了一个交互的场所, 如 seekda[2]、WebserviceX.NET[3]、Programmable Web[4] 等. 注

1. WSDL 官方网站 http://www.w3.org/TR/wsdl.
2. seekda 官方网站 http://webservices.seekda.com.
3. WebserviceX.NET 官方网站 http://www.webservicex.net/ws/default.aspx.
4. ProgrammableWeb 官方网站 http://www.programmableweb.com/.

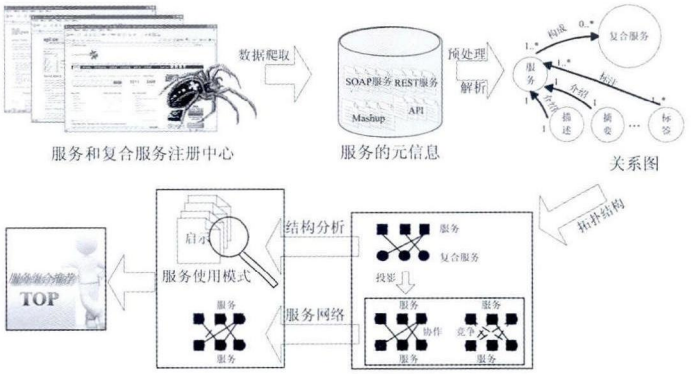


图 1　SCRSN 方法框架

册库通常是按照特定的注册模型进行组织的, 包含了服务的元信息, 如服务的名称、描述、地址、标签、提供者等. 对于一些由服务通过组合构成的复合服务, 甚至包含构成这个复合服务使用的服务列表.

SCRSN 方法将收集注册库中各类服务的元信息 (如名称、描述、标签等), 并存于本地数据库. 这些爬下来的数据可能存在一些噪声, 如作为标签的单词拼写错误、服务重复注册等. SCRSN 方法对收集到的数据进行预处理, 通过一些算法修正单词的拼写错误、删除重复注册的服务等, 尽量排除数据中的噪声. 数据的收集可以借助一些整站下载器, 如 Teleport Ultra[5]、webzip[6]、WinHTTrack[7]等来完成: 首先, 用整站下载器将网站中的相关页面以 html 静态页面的形式爬到本地; 然后, 编写一些脚本解析 html 页面中的相关字段并存于本地数据库中; 最后, 再对收集到的数据进行预处理以减少噪声.

2.2　服务网络的定义

多个服务可以通过组合形成功能更为强大的服务 (复合服务). 服务组合主要表现了服务之间的协作关系. 同时, 功能相近的服务间又存在竞争关系. 如果我们将服务 (复合服务) 看成节点, 服务间的交互关系 (协作/竞争) 看成边, 那么服务及服务关系就形成了网络结构. 为了清晰地表达服务间的这种关系, 我们利用服务组合数据 (复合服务), 通过一定的规则建立服务的网络.

定义 1 复合服务 - 服务网络　复合服务 - 服务网络 (Composited Service-Service Network, 简称 CS2SN) 可以用一个二分网络表示, 即 CS2SN $= (N_{cs}, N_s, D)$. 其中: N_{cs} 为复合服务节点的集合, 表示所有的复合服务; N_s 是服务节点的集合, 表示复合服务使用的所有服务; D 是一个无向边的集合, 表示复合服务对服务的使用关系. CS2SN 的边只存在于不同类型节点间, 即 $D = \{\{cs_i, s_j\}, cs_i \in N_{cs}, s_j \in N_s\}$. CS2SN 的连接矩阵 ψ 描述了复合服务节点和服务节点间的连接关系:

$$\psi_{ij} = \begin{cases} 1, & \{cs_i, s_j\} \in D \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

其中: ψ 是一个 $A \times B$ 的二值矩阵, $A = |N_{cs}|$ 表示复合服务的数量, $B = |N_s|$ 表示服务的数量. 若 $\psi_{ij} = 1$, 则第 i 个复合服务节点和第 j 个服务节点间存在边, 否则不存在边.

CS2SN 表示的是两类不同节点 (复合服务节点和服务节点) 间的关系. 同类型节点间的关系 (如服务和服务的关系) 可以通过对 CS2SN 作投影得到 (如图 2 所示).

定义 2 服务协作网络　服务协作网络 (Service Cooperation Network, 简称 SCN) 可以用一个单模无向图表示, 即 SCN $= (N_s, D_s)$. 其中: N_s 为服务节点的集合, 表示所有复合服务所使用服务的集合; D_s 是一个无向边的集合, 表示服务间的协作 (组合) 关系, 即若两个服务共同作为某个复合服务的构成部分, 则代表这两个服务的节点间存在一条无向边. SCN 的连接矩阵 ψ^s 可由 ψ 得到, 即:

$$\psi^s_{ij} = \sum_k \psi_{ki}\psi_{kj} \tag{2}$$

其中: ψ^s_{ij} 表示服务 i 和 j 共同参与的复合服务的数量, ψ^s_{ii} 表示服务 i 参与的复合服务的数量. 因此, 若 $\psi^s_{ij} > 0$ 则服务 i 和 j 之间存在协作关系, 否则不存在协作关系.

5. Teleport Ultra 官方网站 http://www.tenmax.com/teleport/ultra/home.htm.
6. webzip 官方网站 http://www.spidersoft.com/webzip.
7. WinHTTrack 官方网站 http://www.httrack.com.

同时, 我们也给 SCN 中每条边赋予一个权值, 用于刻画这条边两端节点代表的服务间可以协作的可能性. 服务节点 i 和 j 间边的权值 w_{ij}^s 定义如下:

$$w_{ij}^s = \frac{1}{N_s^k - 1}\psi_{ij}^s \tag{3}$$

其中: N_s^k 是第 k 个复合服务使用的服务的数量. 显然, w_{ij}^s 描述了服务 i 和 j 间可以协作的可能性大小. w_{ij}^s 值越大, 则服务 i 和 j 间可以组合的可能性越大. 因此, SCN 实质上包含了从已有复合服务中获得的服务之间是否可以协作的知识. 可以想象, 复合服务越多, 服务之间可以组合的知识就越丰富, 当复合服务的数量趋于无穷时, SCN 中将包含服务间的所有可组合关系. 但是, 复合服务的数量往往是有限的, 所以已有复合服务中服务之间真实的连接网络 $\omega_s = (N_s, E_s)$ 和 SCN 满足:

$$E_s \cap D_s \neq \emptyset \tag{4}$$

定义 3 服务相似网络 服务相似网络 (Service Similarity Network, 简称 SSN) 可以用一个单模无向图表示, 即 SSN $= (N_s, D_{sim})$. 其中: N_s 为服务节点集合, 表示所有复合服务使用的服务集合; D_{sim} 是一个无向边的集合, 表示服务间的相似关系, 边上的权值表示两个服务间功能的相似程度. 服务的相似程度代表了服务的竞争关系, 也表示服务之间的可替换关系. 权值越大竞争越激烈, 也越有可能相互替换. 一般而言, 多个服务在功能上等价或相似, 它们之间必然存在竞争: 即在某些应用场景下, 用户或者选择服务 A 或者选择服务 B, 而没有必要同时 "雇佣" 多个竞争性的服务 [12].

服务的元信息 (服务的名称、描述、摘要、分类信息、标签等) 通常包含了服务功能的描述, 通常表现为长短不同的文本. 本文任意服务间功能的相似度主要是通过相应服务元信息的相似度得到. 计算两个文档之间相似度最著名的方法是向量空间模型 [13]. 对于一个文档, 进行分词以及停用词等语言处理以后, 可以得到一组词语 (或短语), 同时每个词语 (或短语) 在文档中有相关的权重信息, 若每个词语 (或短语) 对应一维, 一个 N 维空间向量就可以建立起来:

$$d_j = (w_{1,j}, w_{2,j}, \cdots, w_{N,j}) \tag{5}$$

其中: d_j 表示第 j 个文档对应的空间向量; N 表示文档中不同的词语 (或短语) 的数量; $w_{i,j}$ 是第 i 个词语 (或短语) 的权重.

根据 $w_{i,j}$ 值的不同表达方式, 向量空间模型可以分成两种类型: 1) 布尔向量空间模型: 文档中所有出现的词语 (或短语) 相对应的维值为 1, 不出现为 0. 显然, 布尔向量空间模型比较简单, 不能体现词语 (或短语) 在文档中的权重. 2) 词频 - 逆向文件频率模型 (TF-IDF 模型): 在 TF-IDF 模型中, 每个词的权重信息可以由两个参数量度: 词频 (term frequency) tf_{ij}, 表示词语 (或短语) w_i 在文档 d_j 中出现的频率; 逆文本词频 (inverse document frequency) $\log(M/n_j)$, 其中 M 表示文本集中所有的文档数量, n_i 表示文本集中所有含有词语 w_i 的文档数量. 文档 d_j 中词 w_i 的权重通常使用 $w_{i,j} = tf_{i,j} \cdot \log(M/n_i)$ 表示. 这个权重被称为 TFIDF 权重.

建立文档向量空间模型以后, 常用两个文档向量之间的余弦系数表示文档间的相似系数. 余弦系数通常使用下面的公式进行计算:

$$Sim(d_i, d_j) = Cos(d_i, d_j) = \frac{\sum_{k=1}^{|T|} w_{k,i} \times w_{k,j}}{\sqrt{\sum_{k=1}^{|T|} w_{k,i}^2 \times \sum_{k=1}^{|T|} w_{k,j}^2}} \tag{6}$$

其中: $Sim(d_i, d_j)$ 表示文档 d_i 和 d_j 的相似性; $Cos(d_i, d_j)$ 表示 d_i 和 d_j 间的余弦系数; T 表示词 (或短语) 的集合; $|T|$ 是词 (或短语) 的个数.

服务的元信息, 如服务的名称、描述、地址、标签、提供者等, 本质上是一些文本, 但是表现形式上存在差异. 针对元信息文本的不同形式, 可以使用布尔向量空间模型或 TF-IDF 模型求相应元信息的相似度: 1) 长文本中包含的词语比较多, 而且词语重复概率高, 所以针对长文本 (如描述), 我们可以使用 TF-IDF 模型. 2) 短文本包含的词语比较少, 且重复率低, 所以针对短文本 (如摘要), 我们可以使用布尔向量空间模型. 3) 对于像标签这种一个个词语 (或短语) 组成的特殊短文本, 词语基本不重复, 可以使用布尔向量空间模型.

在求得了元信息各个部分的相似度之后, 服务的相似度可以通过各个部分的加权和来量度:

$$Sim(s_i, s_j) = \sum_{k=1}^{K} \alpha_k Sim(s_{i,k}, s_{j,k}) \tag{7}$$

其中: $Sim(s_i, s_j)$ 表示服务 s_i 和 s_j 的相似性; K 表示参与服务相似度计算的元信息的种类; $Sim(s_{i,k}, s_{j,k})$ 表示服务 s_i 和 s_j 的相似性在元信息 k 上的分量; a_k 是相应分量的权重.

为了避免权值设置的主观性, 本文采用变异系数法 (coefficient of variation method, 简称 CV)[14] 为 a_k 赋值. CV 可以直接利用各项指标所包含的信息, 通过计算得到指标的权重, 是一种客观的赋权方法, 在统计实践中很常用. 它主要分两个步骤完成权值计算:

1) 为了消除各度量量纲不同的影响, 需要用变异系数来衡量各度量取值的差异程度. 各度量的变异系数如下:

$$V_i = \frac{\sigma_i}{\bar{x}}, \quad i = 1, 2, \cdots, n \tag{8}$$

其中: V_i 是第 i 项度量的变异系数, 也称为标准差系数; σ_i 是第 i 项度量的标准差; $\bar{x}$ 是第 i 项度量的算数平均值; n 表示度量的个数.

2) 第 i 项度量的权重 α_i 为:

$$\alpha_i = \frac{V_i}{\sum_{i=1}^{n} V_i} \tag{9}$$

图 2 显示的是从我们收集的真实 API 服务和 mashup 应用数据中构建的 CS2SN、SCN 和 SSN. 其中: mashup 应用对应复合服务, API 服务对应服务.

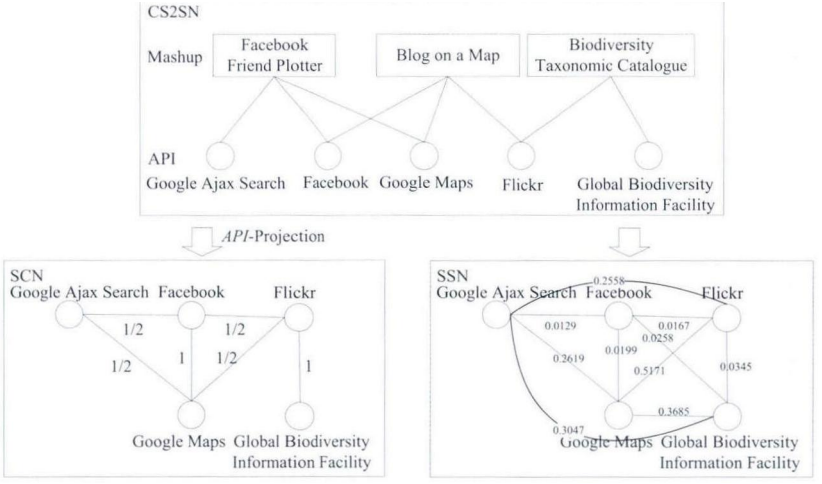


图 2　CS2SN、SCN 和 SSN 构建举例

2.3 服务使用模式分析

由服务组合构成的复合服务, 包含了服务使用的成功经验 (其中的服务必定是可组合的). 分析复合服务的组成结构, 挖掘服务的使用模式, 可以为构建新的复合服务提供指导. 服务和复合服务间的构成关系可以用 CS2SN 抽象. 因此服务的使用模式可以通过分析 CS2SN 的结构得到.

复杂网络成为人们关注的热点之一. 它的基本观点是结构决定功能. 强调用整体的观点研究系统 [15]. 特别是随着 "小世界" 和 "无标度" 等特征的发现, 科学家们掀起了一股研究复杂网络的热潮, 研究遍及物理、数学、生物等多个领域. 二分网络是复杂网络的重要形式, 得到了人们的普遍关注, 人们提出了多种方法挖掘其蕴含的结构特征和演化规律. CS2SN 本质上也是一个二分网络, 所以可以使用二分网络研究中的方法进行研究. 本小节我们使用二分网络研究中比较常用的度数中心度 (degree centrality, 简称 DC) 指标来分析 CS2SN 的结构特征, 挖掘服务的使用模式.

定义 4 度数中心度 [16]　在二分网络中, 一个点的度数中心度是该节点所隶属的事件数, 一个事件的度数中心度是该事件所拥有的行动者数. 在网络图上, 表现为与该节点有边相连的节点数. 如图 2(上部分) 中, Blog on a Map 的度数中心度是 3, 即 DC(Blog on a Map)=3.

相应的在 CS2SN 中, 服务的度数中心度描述了该服务参与构成的复合服务的数量. 复合服务的度数中心度描述了该复合服务使用的服务的数量. 这两个度数中心度反映了复合服务使用服务的模式 (或服务参与构成复合服务的模式), 即复合服务使用几个服务, 服务被几个复合服务使用. 这里我们主要使用复合服务的度数中心度为服务组合推荐提供指导.

在得到复合服务的度数中心度后, 我们将分析这些数值的分布, 从而确定大部分复合服务使用服务数的一个较小范围 $[A, B]$. 譬如如果有 100 个复合服务, 这 100 个复合服务的度数中心度从 1 到 30 不等, 但是若 90%(这个数字可以由用户设定, 建议大于 80%) 以上的复合服务的度数中心度都小于 10, 那么我们就设置 $[A, B]$ 为 $[1, 10]$. $[A, B]$ 实际上是复合服务使用服务数的一个范围, 我们新开发的复合服务, 它使用的服务数也在很大概率上属于这个范围.

2.4 服务推荐算法

如前所述, CS2SN 包含了服务的使用模式, 从 CS2SN 得到的 SCN 自然包含了服务之间交互和协作的知识. 实际上, SCN 中任意长度的路径实际上就是一个潜在的服务组合. 但是, 服务更新频繁, 服务的可用性不能持续保证, 在服务组合的过程中, 往往会碰到正在使用的某个服务失效, 这时就需要寻找功能相似的服务进行替代 (即竞争关系的服务的发现), 这是基于 SCN 的服务推荐方法无法实现的. SSN 包含了服务的相似 (竞争) 关系, 可以弥补基于 SCN 的服务推荐方法的不足. 因此, SCN 结合 SSN 可以作为服务推荐的基础设施, 要构建新的复合服务, 只需要依据一定的规则对 SCN 和 SSN 遍历即可.

如果用户在使用某个服务时发现该服务失效, 就需要寻找功能相似的服务进行替代, 即竞争关系的服务的发现. 基于 SSN 可以找到与失效服务功能最接近的可替换服务. 可替换服务的推荐算法 RSRA (replaceable service recommendation algorithm) 如算法 1 所示.

算法 1: 可替换服务推荐算法 RSRA

输入: SSN、失效服务集合 FS、用户已选择的服务集合 S_{sel}、待推荐的服务个数 k

输出: 可替换服务集合

注: $NGH(S)$ 为 SSN 中与服务 S ($S \in S_{sel}$) 相连的节点集合; CN 是 SCN 中度数中心度非 0 的节点集合

```
if |S_sel| equals 1 then
    if |(NGH(S) − FS) ∩ CN| ≥ k then
        在 (NGH(S) − FS) ∩ CN 中选择与服务 S 相连且边权 top-k 的服务进行推荐 (边权降序排列)
    else
        在 (NGH(S) − FS) ∩ CN 中选择与服务 S 相连且边权 top-(|(NGH(S) − FS) ∩ CN|) 的服务进行推荐 (边权降序排列)
    end if
else
    提示输入的服务必须为单个服务
end if
return
```

我们提出了一种基于服务网络的服务组合推荐算法 SN-SCRA (service composition recommendation algorithm based on service networks), 为用户推荐服务组合方案. SN-SCRA 首先将复合服务开发中服务的使用场景分为 3 种类型, 并根据使用场景为用户推荐服务组合方案:

1) 场景一: 用户还未选择任何服务

SN-SCRA 将 SCN 中的节点按照度数中心性降序排列, 并将前 k 个服务推荐给用户 (k 是用户指定的一个数据). 度数中心性大的节点, 表明其在系统中频繁被使用, 起到了平台的作用, 所以也可以看成是一种 "平台服务"[8].

2) 场景二: 用户已选了一个服务 S

SN-SCRA 将分两种情况: ① S 不属于无效服务集, 并且在 SCN 中不是孤立点; ② 情况 ① 之外的情况, 具体如算法 2 所示.

3) 场景三: 用户已选了 n ($n > 2$) 个服务

对于选择了 n ($n > 2$) 个服务的服务推荐问题, 实际上是在 SCN 中寻找包含这几个节点在内的一条路径. 同时, 从 2.3 小节的分析我们可以知道, 复合服务一般由 B 个以内的服务构成, 所以在推荐算法中, 针对应用场景三, SN-SCRA 限定推荐的服务路径长度 l 在 B 个以内, 即 $l \leq B$. 详见算法 3.

因此, 服务的推荐算法 SN-SCRA 详见算法 4. 如果 SCN 网络的规模比较大 (节点和边的数量大), 节点间的连接可能会比较稠密, 为了提高推荐算法的效率, 我们可以在算法中提供一个边权过滤值 w_{th} ($w_{th} \in$

$[0,1]$), 用于将那些边权值很小的边过滤掉. 因为权值很小的边意味着边两端服务组合的可能性很小.

算法 2: 单个服务推荐算法

输入: SCN、失效服务集合 FS、用户已选择的服务集合 S_{sel}、待推荐的服务个数 k

输出: 推荐的服务集合

注: $NGH(S)$ 为 SCN 中与服务 S ($S \in S_{sel}$) 相连的节点集合; CN 是 SCN 中度数中心度非 0 的节点集合

```
if S ∉ FS ∧ S ∈ CN then
  if |(NGH(S) − FS) ∩ CN| ≥ k then
    在 (NGH(S) − FS) ∩ CN 中选择与服务 S 相连且边权 top-k 的服务进行推荐 (边权降序排列)
  else
    在 (NGH(S) − FS) ∩ CN 中选择与服务 S 相连且边权 top-(|(NGH(S) − FS) ∩ CN|) 的服务进行推荐 (边权降序排列)
  end if
else
  按照算法 1 获得可替代服务集合 RS
  for each S in RS do
    if |(NGH(S) − FS) ∩ CN| ≥ k then
      将 (NGH(S) − FS) ∩ CN 中与服务 S 相连且边权 top-k 的服务加入集合 R
    else
      将 (NGH(S) − FS) ∩ CN 中与服务 S 相连且边权 top-(|(NGH(S) − FS) ∩ CN|) 的服务加入集合 R
    end if
    在 R 中选择与服务 S 相连且边权 top-k 的服务进行推荐
  end for
end if
return
```

算法 3: 多个服务推荐算法

输入: SCN、失效服务集合 FS、用户已选择的服务集合 S_{sel}、待推荐的服务个数 k、路径长度 l

输出: 推荐的服务集合

```
for S_sel 中任一节点组合 N_s 和 N_e do
  求 SCN 中以 N_s 为起点, N_e 为终点, 包含 S_sel 中其它节点, 且长度为 l, l − 1, l − 2, ···, |S_sel| 的所有路径, 并将其放入集合 S_route
  for each S_r in S_route do
    if GS(S_r) ∩ FS is not NULL then
      //GS(S_r) 获得路径 S_r 中服务的集合
      for each S_i in GS(S_r) ∩ FS do
        用算法 1 求 S_i 的可替换服务集 RS_i
      end for
      得到替换后的路径 S'_r ∈ {(S'_{r,1}, ···, S'_{r,|GS(S_r)∩FS|}) | S'_{r,i} ∈ RS_i}, 并将其放入集合 S'_route, 即 S'_r ∪ S'_route
    else
      S_r ∪ S'_route
    end if
    按照路径长度对 S'_route 中的路径归类, 输出每一类中边权和 top-k 的路径进行推荐
  end for
end for
return
```

算法 4: 服务推荐算法 SN-SCRA

输入: SN、失效服务集合 FS、用户已选择的服务集合 S_{sel}、待推荐的服务个数 k、边权过滤值 w_{th}、路径长度 l

输出: 推荐的服务集合

```
将 SCN 中, 边权小于 w_th 的边删除
if S_sel is NULL then
    // 场景一
    按照场景一的方法为用户推荐服务
else if |S_sel| equals 1 then
    // 场景二
    按照场景二的方法为用户推荐服务组合
else
    // 场景三
    按照场景三的方法为用户推荐服务组合
end if
return
```

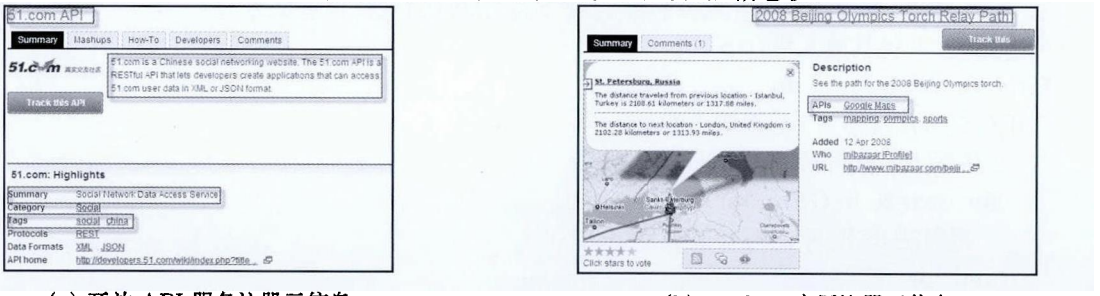
3 实例分析

本节我们将结合 ProgrammableWeb 上开放 API 服务和 mashup 应用的真实数据, 进行实证分析, 说明 SCRSN 方法的具体实施过程, 同时验证其可行性和有效性.

3.1 数据来源

ProgrammableWeb 是著名的开放 API 服务和 mashup 应用注册库, 罗列了 6,000 多个 mashup 应用和 4,000 多个 API 服务, 并提供了详细的服务注册元信息, 包括名称、描述、摘要、提供者、标签等 (如图 3 所示). 开放 API 是服务, mashup 应用是由开放 API 组合而成的复合服务, 包含服务组合的历史信息, 符合本文工作的要求. 因此, 本节我们以 ProgrammableWeb 上的 mashup 应用和开放 API 服务的真实数据为例展开研究.

我们使用自己开发的网爬工具将 ProgrammableWeb 上截止到 2011 年 12 月 14 日 (本文工作开展时) 所有开放 API 的名称、描述、摘要、标签以及 mashup 应用的名称、使用的开放 API 等 (即图 3 方框内部分) 信息爬了下来, 存储在本地数据库中. 这些数据是后续工作的基础. 开放 API 中的分类信息尽管包含很多功能信息, 但是它与标签存在很大的重复. 我们发现在我们收集的服务中, 95% 以上的服务其分类字段中的值也在标签字段中出现. 其它一些信息并未处理, 因为它们包含的功能信息较少.



(a) 开放 API 服务注册元信息　　(b) mashup 应用注册元信息

图 3 开放 API 和 mashup 注册元信息

ProgrammableWeb 上的数据虽然都是开放 API 服务提供者提交的, 但是也存在一定的随意性. 在数据收集的过程中, 我们发现这些数据存在一些错误: 1) 有些 API/mashup 存在重复注册现象. 2) 有些 API/mashup 仅提供了名称, 但是其它我们所需的注册信息缺失了. 3) 用于标识 API/mashup 的标签也存在不一致, 有些标签甚至拼写错误. SCRSN 方法按照文献 [8] 中的方法对所获得的数据进行处理. 最终, 我们的数据集包含 6,362 个 mashup 应用、4,506 个开放 API 服务和 1,175 个标签, 数据集可以从 [17] 下载.

3.2 服务网络构建及 API 服务使用模式分析

Mashup 注册元信息 (如图 3 右所示) 包含了其使用的开放 API 服务列表, 通过解析该网页, 我们构建

了所有 mashup 应用和开放 API 服务之间的 CS2SN (如图 4 所示). 其中 mashup 是复合服务 (圆形节点共 6,362 个)、开放 API 是服务 (方形节点对应 API 服务共 982 个), 两类节点间的边代表 mashup 应用使用 API 服务的关系.

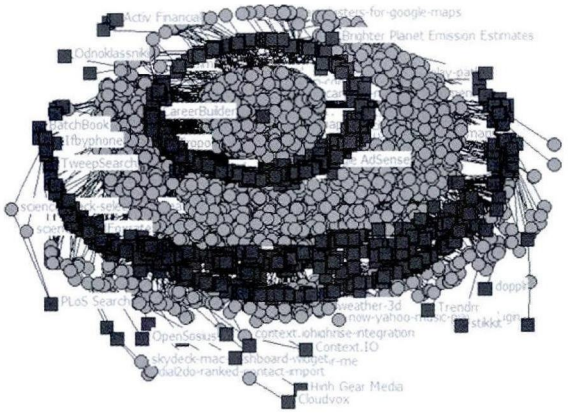


图 4 CS2SN 图示

表 1 度数中心度 top-10 的开放 API 服务

类型	名称	度数中心度
开放 API 服务	Google Maps	2,303
开放 API 服务	Twitter	661
开放 API 服务	Flickr	590
开放 API 服务	YouTube	589
开放 API 服务	Amazon eCommerce	403
开放 API 服务	Twilio	333
开放 API 服务	Facebook	320
开放 API 服务	eBay	214
开放 API 服务	Last.fm	206
开放 API 服务	Google Search	178

为了分析 mashup 应用使用开放 API 服务的模式, 我们使用度数中心度指标 DC 分析了该 CS2SN 的结构特征. 图 5 显示了具有某一度数中心度的 mashup 应用的比率和开放 API 服务的比率. 可以发现, 大部分 mashup 应用具有较小的度数中心度 (78.8% 的 mashup 应用的度数中心度仅为 1 或 2), 只有 0.99% 的 mashup 应用度数中心度大于 10, 平均每个 mashup 应用度数中心度仅为 10.9. 这说明大部分 mashup 应用仅由少量的 API 服务组合而成. 同样的现象也在开放 API 服务中发现了: 大部分的开放 API 服务具有较小的度数中心度 (54.1% 的开放 API 服务度数中心度仅为 1 或 2), 只有不到 15.0% 的 API 服务度数中心度大于 10, 平均每个 API 服务的度数中心度不到 11.0. 这说明大部分 API 服务仅参与了少量的 mashup 应用. 由此, 在本文中我们将 2.3 节中提到的范围 $[A, B]$ 确定为 [1, 10]. 同时, 度数中心度最大的 10 个开放 API 服务 (平台 API 服务) 如表 1 所示.

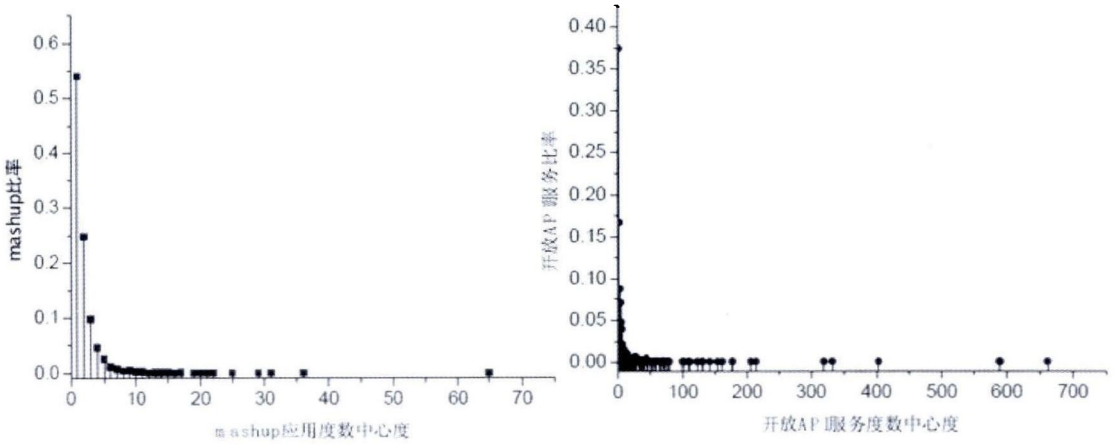


图 5 具有某度数中心度的 mashup 应用比率 (左) 和 API 服务比率 (右)

通过对 CS2SN 作投影操作, 可以得到 CS2SN 相应的 SCN (如图 6 所示). SCN 包含 982 个节点, 10,966 条边, 由 125 个弱连通子图 (不同的颜色代表不同的弱连通子图) 构成, 其中最大的弱连通子图有 846 个节点 (黄色节点部分), 最小的子图节点数仅为 1 (孤立点). 节点边上的标签是该节点所属子图的一个标识符, 边上的数值是该边两端服务协作可能性大小. 为了使图显示得清晰, 我们忽略了每个节点对应 API 服务的名称.

同时, 我们计算了 API 服务间的相似度构造了开放 API 服务的 SSN. 该网络有 4,506 个节点, 8,629,141 条边, 我们的工具无法绘制如此密集的网络 (我们也未找到可以显示如此规模网络的开源工具), 所以未提供 SSN 的网络图示. 该网络的数据可从 [17] 下载.

3.3 实验结果及分析

SCRSN 方法主要是针对文献 [8] LFH 方法中存在的两大问题提出来的, 即: 1) LFH 构建的网络主要考虑的是服务间的协作关系, 未考虑竞争关系, 在服务失效时推荐能力低下; 2) LFH 方法不能为没有历史使

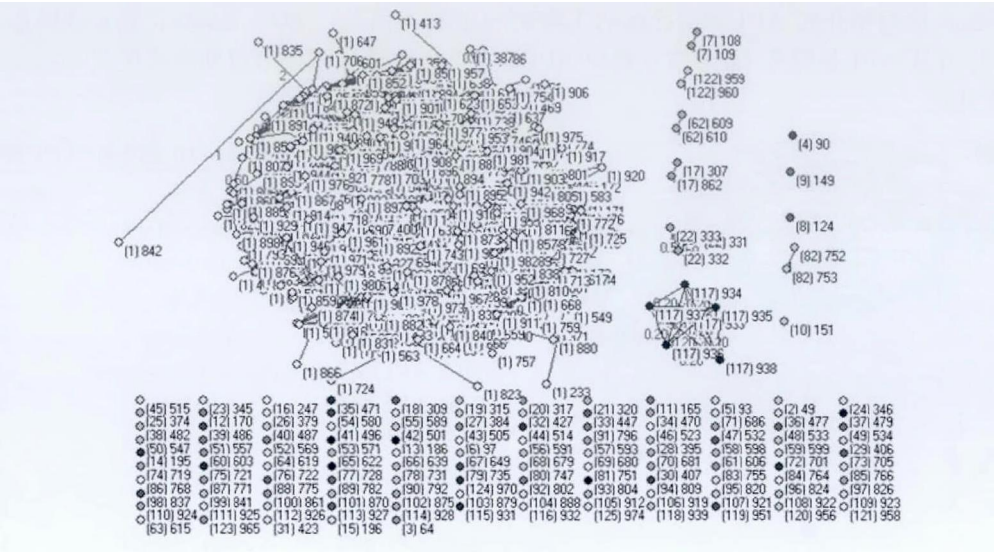


图 6 SCN 图示

用记录的服务推荐组合方案, 可推荐的服务数量少. 因此, 本节设计的实验也主要评价 SCRSN 方法在解决这两方面问题的能力, 并主要从服务推荐算法的可行性和有效性两方面进行验证. 实验的环境为 Dell 笔记本, CPU 为双核英特尔 T7250, 内存为 2G, 操作系统为 Windows XP.

3.3.1 可行性验证

LFH 方法存在的问题之一就是只能对有过使用历史的 API 服务进行推荐. 本文要解决的一个主要问题就是为没有使用历史的 API 服务进行可组合服务的推荐. 如前所述, 我们获得的数据中有 4,506 个开放 API 服务, 但是构建的 CS2SN 中只有 982 个开放 API 服务. 因此, LFH 对于其余的 3,524 个 API 服务的推荐是无能为力的. 本文提出的 SCRSN 方法由于考虑服务间的相似 (竞争) 关系, 可以实现没有使用记录服务的推荐. 如 API 服务 "3defy", 由于该服务没有使用的历史, LFH 不能进行组合推荐, 而通过本文的方法, 我们首先根据算法 1 (将 "3defy" 看成失效服务) 可以得到该 API 服务的 top-k (如 $k=5$ 时) 可替代 API 服务集合 {google-earth, adl-3d-repository-3dr, pixagogo, earthmine-direct-data, ericsson-3d-landscape}, 然后根据该集合得到 API 服务的可组合服务集合 {google-maps-data, tr.im, google-maps-flash, google-custom-search, $\cdots$, google-custom-search, google-fusion-tables, geocoder}. 所以通过本文提出的方法能够实现为没有使用历史的 API 服务推荐可组合服务.

我们设计实验测试 SCRSN 在所有 API 服务上的服务推荐能力. 在实验中, 根据 k 值的不同, SCRSN 可以提供服务推荐的服务数量存在差异, 如表 2 所示.

在表 2 中, k 为 0 时, SCRSN 可以为 865 个 API 服务提供服务推荐, 等于 LFH 可以提供推荐的 API 服务数, 占了总服务数的 19.2%; k 为 5 时, 可以为 3,233 个 API 服务提供推荐, 占总服务数的 71.7%; 而 k 为 15 时, 可以为 94.1% 的 API 服务提供推荐, 基本可以实现为大部分 API 服务推荐服务. 可见, SCRSN 方法可以实现为没有使用历史的 API 服务推荐服务. 对于失效服务, 可以以类似的方法进行验证. 我们得到了类似的结果, 这里不再赘述.

表 2 可以支持服务推荐功能的服务数量

k 值	可以提供服务推荐的服务数量
0	865
5	3,233
10	3,954
15	4,242

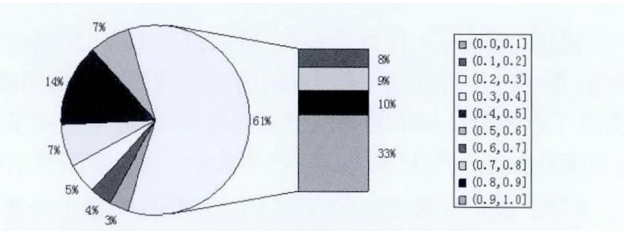


图 7 API 服务推荐有效性数据分布图

3.3.2 有效性验证

3.3.1 节验证了 SCRSN 方法的可行性, 但是这些推荐的服务是否具有组合的可能性, 即 SCRSN 方法是否有效? 我们认为 SCRSN 推荐得到的 API 服务集合包含 LFH 方法的推荐的 API 服务越多, 那么 SCRSN 方法就越有效. 因此, 我们用 $(RS_{LFH} \cap RS_{SCRSN})/RS_{LFH}$ 量度 SCRSN 方法的有效性. 其中 RS_{LFH} 和

RS_{SCRSN} 分别是 LFH 和 SCRSN 方法推荐的服务集合.

我们设计实验分别用 SCRSN 和 LFH 方法为具有使用历史的 API 服务推荐服务, 然后计算 $k = 5$ 时 $(RS_{LFH} \cap RS_{SCRSN})/RS_{LFH}$ 的值. 图 7 显示的是 $(RS_{LFH} \cap RS_{SCRSN})/RS_{LFH}$ 有效性数据的分布图.

通过图 7 可以发现: 在 68% 的 API 服务上, 通过 SCRSN 方法推荐得到的 API 服务集合包含了 50% 以上通过 LFH 方法推荐的 API 服务集合, 并且其中 33% 的 API 服务通过 SCRSN 方法可以得到的 90% 以上通过 LFH 方法得到的可组合 API 服务. 可见, 用 SCRSN 方法得到的可组合服务具有较好的准确性.

4 应用及工具开发

与武汉大学计算机学院合作, 在其构建的软件/服务注册库平台 S2R2 (Software Service Registry and Repository)[8] 中实现了 SCRSN 方法, 构建了一个开放 API 服务推荐子系统. 该子系统主要包括两大功能: API 服务查询推荐, 查询推荐的可视化. 该子系统可以进一步分为四个子系统 (如图 8 所示): 1) 爬虫子系统: 爬取网络上的 API 服务和 mashup 数据, 并存入数据库中; 2) 计算子系统: 将爬取的数据进行可组合以及相似度的计算, 并将计算的结果存入数据库中; 3) 查询子系统: 为 API 提供服务组合推荐, 对用户输入的单个或者多个 API, 从系统 API 关系库中得到推荐的 API 服务; 4) 可视化子系统: 主要提供查询结果的可视化, 将查询结果以列表及图的方式展现给用户, 并且提供 API 之间的关系.

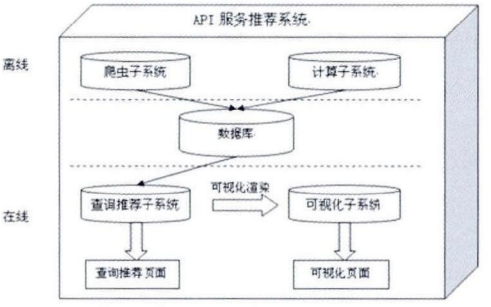
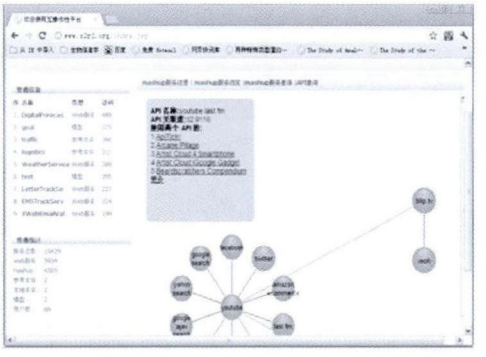


图 8　开放 API 服务推荐子系统结构图

(a) 平台 API 服务推荐 (场景一)

(b) 单个 API 推荐网络图显示 (场景二)

(c) 多个 API 推荐列表显示 (场景三)



(d) API 可组合方案的图展示

图 9　功能示例图

图 9(a) 对应场景一, 即在用户还未选择任何 API 服务时, 为其推荐平台 API 服务的情形. 图 9(b) 对应场景二, 即在用户已输入一个 API 服务 blip.tv[9] 时, 为其推荐可以组合的其它 API 服务的情形. 图 9(c) 对

8. s2r2 官方网站 http://www.s2r2.org.

9. blip.tv 官方网址 http://www.programmableweb.com/api/blip.tv.

应场景三, 即在用户输入 API 服务 blip.tv 和 mtv[10] 时, 为其推荐可组合 API 服务路径的情形. 图 9(d) 是推荐的服务组合方案的图展示效果.

5 结论与展望

本文利用服务组合历史信息及服务注册元信息, 构建服务网络, 描述服务间的协作及竞争关系, 并通过服务网络挖掘服务使用模式, 最终提出了一种新的服务推荐方法. 该方法不依赖于服务的 WSDL 文档, 为解决不具有 WSDL 文档服务的服务组合问题提供了一种解决方案, 也弥补了现有服务组合方法的不足. 本文以 ProgrammableWeb 上 mashup 应用和 API 服务的真实数据为载体进行实例研究, 说明了本文方法的具体实施过程, 同时验证了其可行性和有效性. 本文方法对于解决服务组合相关问题提供了一种新思路.

下一步我们将考虑如下研究工作: 1) 将本文方法与现有的 web 服务组合的研究工作结合起来, 充分利用服务网络和服务 WSDL 文档信息, 提高现有服务组合方法的有效性; 2) 本文最后以 Programmable Web 上 mashup 应用和 API 服务的真实数据验证本文方法的有效性, 我们拟用其它多种类型的服务 (如 web 服务、REST 风格服务) 验证本文方法的可行性和有效性.

参考文献

[1] Papazoglou M P. Service-oriented computing: Concepts, characteristics and directions[J]. Information System Journal, 2003, 10(2): 3–12.

[2] 刘迎春, 郑小林, 陈德人. 基于信任和推荐关系的可信服务发现 [J]. 系统工程理论与实践, 2012, 32(12): 2789–2795.
Liu Yingchun, Zheng Xiaolin, Chen Deren. Trustworthy services discovery based on trust and recommendation relationships[J]. Systems Engineering — Theory & Practice, 2012, 32(12): 2789–2795.

[3] Papazoglou M P, Traverso P, Dustdar S, et al. Service-oriented computing: State of the art and research challenges[J]. Computer, 2007, 40(11): 38–45.

[4] Zhang M W, Zhang B, Liu Y. Web service composition based on QoS rules[J]. Journal of Computer Science and Technology, 2010, 25(6): 1143–1156.

[5] 朱勇, 罗军舟, 李伟. 一种工作流环境下能耗感知的多路径服务组合方法 [J]. 计算机学报, 2012, 34(3): 627–638.
Zhu Yong, Luo Junzhou, Li Wei. An approach for energy aware multipath service composition based on workflow[J]. Chinese Journal of Computer, 2012, 34(3): 627–638.

[6] Oh S C, Lee D W, Kumara S R T. Effective web service composition in diverse and large-scale service networks[J]. IEEE Transactions on Services Composition, 2008, 1(1): 15–32.

[7] Medjahed B, Bouguettaya A, Elmagarmid A. Composing web service on the semantic web[J]. The International Journal on Very Large Data Bases (The VLDB Journal), 2003, 12(4): 333–351.

[8] 潘伟丰, 李兵, 邵波, 等. 基于软件网络的服务自动分类和推荐方法研究 [J]. 计算机学报, 2011, 34(12): 2355–2369.
Pan Weifeng, Li Bing, Shao Bo, et al. Service classification and recommendation based on software networks[J]. Chinese Journal of Computer, 2011, 34(12): 2355–2369.

[9] 何平, 郑益中, 孙燕红. 基于服务质量和价格的服务竞争行为 [J]. 系统工程理论与实践, 2014, 34(2): 357–364.
He Ping, Zheng Yizhong, Sun Yanhong. Service competition based on service quality and price[J]. Systems Engineering — Theory & Practice, 2014, 34(2): 357–364.

[10] Brook Jr F P. Three great challenges for half-century-old computer science[J]. Journal of the ACM, 2003, 50(1): 25–26.

[11] Jiang B, Zhang X X, Pan W F, et al. BIGSIR: A bipartite graph based service recommendation method[C]// Proceedings of the 9th World Congress on Services, Santa Clara, CA, 2013: 363–369.

[12] 陈世展, 冯志勇, 王辉. 服务关系及其在面向服务计算中的应用 [J]. 计算机学报, 2010, 33(11): 2068–2083.
Chen Shizhan, Feng Zhiyong, Wang Hui. Service relations and its application in services-oriented computing[J]. Chinese Journal of Computer, 2010, 33(11): 2068–2083.

[13] Salton G, Wong A, Yang C S. A vector space model for automatic indexing[J]. Communications of ACM, 1975, 18(11): 613–620.

[14] McClave J T, Benson P G, Sincich T. Statistics for business and economics[M]. 10th ed. New Jersey: Prentice Hall, 2008.

[15] 张铙. 网络与复杂系统 [J]. 科学中国人, 2004, 10: 37.
Zhang Ba. Network and complex system[J]. Scientific Chinese, 2004, 10: 37.

[16] Wikipedia[M/OL]. [2012-06-29] http://en.wikipedia.org/wiki/Centrality.

[17] Dataset[M/OL]. [2012-06-29] http://www.whucn.com/wfpan.htm.

10. Mtv 官方网址 http://www.programmableweb.com/api/mtv.