一种基于服务簇网元模型的Web服务发现方法

栾文静 杜玉越
(山东科技大学信息科学与工程学院 青岛 266590)

摘 要 定义了服务簇的网元模型,提出了一种基于服务簇网元模型的Web服务发现方法。首先,通过计算Web服务的功能描述及参数的语义相似度,对服务库中的服务进行聚类;其次,对服务参数进行统一标注,建立服务簇的网元模型,并对服务簇参数矩阵进行规范化处理;最后,基于服务簇参数矩阵,实现服务快速发现。基于Petri网,首次提出了服务簇的形式化模型,并在此基础上进行了服务快速发现。结果表明,利用网元模型建模服务簇是有效的、合理的,并且与传统的基于参数匹配的服务发现相比,所提方法有效地减少了参数匹配次数,提高了服务发现效率。

关键词 参数不确定性,语义相似度,服务聚类,网元模型,参数矩阵,服务发现
中图法分类号 TP311 **文献标识码** A

Web Service Discovery Method Based on Net Unit Model of Service Cluster

LUAN Wen-jing DU Yu-yue
(College of Information Science and Engineering, Shandong University of Science and Technology, Qingdao 266590, China)

Abstract A net unit model of service cluster was defined and a Web service discovery method based on the model was proposed. First, service clustering was completed based on the semantic similarity of functional description and the parameters. Second, the net unit models of service clusters were built with the unified labels of service parameters, and then service cluster parameter matrix was constructed with standardize treatment. And finally, service discovery was realized by using the service cluster parameter matrix. On the base of Petri net, the formal model of service cluster was proposed for the first time, and a quick service discovery was realized. The results indicate that the model to construct service clusters is effective and reasonable. Compared with the traditional service discovery methods based on the parameter matching, this proposed method can effectively reduce the number of parameters matching times and improve the service discovery efficiency.

Keywords Parameter uncertainty, Semantic similarity, Service clustering, Net unit model, Parameter matrix, Service discovery

1 引言

语义Web服务[1]是一种具有良好前景的技术,它采用机器可理解的方式来描述Web服务自身的功能[2,3],从而提供Web服务之间的互操作。随着面向服务计算体系架构(Service Oriented Architecture, SOA)的进一步推广,语义Web服务数量不断增加。如何合理地对服务建模以快速返回满足用户需求的服务成为Web服务研究的关键问题。

为了提高服务发现效率,文献[4-6]将聚类技术应用到服务发现的预处理过程中,将服务库中的海量服务进行聚类,形成一个个服务集合(服务簇,Web Service Cluster),使得同一簇中的服务相似度极高,不同服务簇中的服务高度相异。但是,它们大都关注聚类方法的研究,很少涉及服务簇的形式化描述,也没有给出具体的基于服务簇的服务发现方法。文献[7]在服务匹配之前,先对服务库中的服务进行聚类预处理,在进行服务发现时,可以过滤掉与服务请求完全不同类别的服务,避免在相似度较低甚至不相似服务的匹配计算上浪费大量时间,从而提高匹配效率,节省资源开销。但是,在选定的服务簇中发现满足需求的服务时,仍然采用服务请求与服务簇中的服务逐一比较的方式,因此其效率较低。本文对服务簇中服务的参数进行有效处理,形成了参数矩阵,进而对其进行统一组织和管理,使得在选定特定服务簇之后,能够根据参数矩阵快速地过滤掉不满足服务请求的服务,缩减从特定服务簇中发现服务的时间,从而进一步提高服务发现效率。

本文第2节定义了Web服务、服务请求,并基于Petri网的基础知识定义了服务的网元模型;第3节按照服务聚类、服务簇网元模型的构建、服务簇参数矩阵规范化、具体的服务发

到稿日期:2011-10-17 返修日期:2012-03-09 本文受国家973计划项目(2010CB328101),国家自然科学基金(61170078),教育部高等学校博士学科点专项科研基金博导类课题(20113718110004),山东省科技发展计划项目(2011GGX10114),山东科技大学研究生科技创新基金(YCA110335)资助。

栾文静(1987-),女,硕士生,主要研究方向为Web服务动态组合、Petri网理论与应用、CSCW,E-mail:wenjingmengjing@163.com;杜玉越(1960-),男,博士,教授,博士生导师,主要研究方向为CSCW、软件工程、形式化技术、Petri网理论与应用。

现算法的顺序系统地介绍了基于服务簇网元模型的服务发现方法；第4节对算法进行了分析，并以机票查询服务簇为例，从匹配次数和返回结果两方面对传统的服务匹配和基于服务簇参数矩阵的服务发现进行了比较；最后是本文的结论和下一步的研究工作。

2 基本概念

定义 1(Web 服务) 一个服务可以描述为一个六元组 $WS=(BI,SF,BC,I,O,QoS)$，其中，

(1) BI 表示服务的基本信息，包括服务的名称、服务 ID、服务提供者的名称等；

(2) SF(Service Functionality)为服务的功能语义描述，表示为概念对的形式，即 $SF=(action,object)$，其中 $action$ 表示服务的动作，$object$ 表示该动作的对象；

(3) $BC=C_1\wedge C_2\wedge\cdots\wedge C_n$，表示行为的约束和规范集合，其中 $C_i(i=1,\cdots,n)$表示一个具体的约束；

(4) I/O 表示服务的输入/输出参数集合，且 $\forall P\in I/O$，$P=(P.name,P.concept,P.constraint)$；

(a) $P.name$ 为输入/输出参数的名称；

(b) $P.concept$ 为该输入/输出参数所关联的领域本体概念，用来体现服务参数的语义；

(c) $P.constraint\in\{0,1\}$，当 $P\in I$ 时，表示该参数是否为实现服务所必需的输入，1 表示该参数是必选的，0 表示该参数是可选的；当 $P\in O$ 时，$P.constraint=1$；

(5) QoS 提供对服务质量的描述，主要包含可用性、可访问性、响应时间、费用、可靠性和安全性等属性。

例 1 设 WS 是一个预订机票酒店的服务。其中 $SF=(Book,Hotel)$，$BC=C_1\wedge C_2$，C_1 为 100<money<200，表示预定的酒店价格在 100～200 元之间；C_2 为 Star=3，表示要预定三星级的酒店。

定义 2(服务请求) 一个服务请求可以描述为五元组 $WS_{req}=(SF_{req},BC_{req},I_{req},O_{req},QoS_{req})$，其中，

(1)SF_{req}表示服务请求者提出的服务要完成的功能；

(2)BC_{req}表示服务请求者提出的行为约束；

(3)I_{req}表示服务请求者可以提供的输入集合，并且每个输入参数都用概念进行标注；

(4)O_{req}表示服务请求者所需要的输出集合，并且每个输出参数也都用概念进行标注；

(5)QoS_{req}表示服务请求者对服务质量各属性的约束集合。

Petri 网[8,9]是一种分布式系统的建模和分析工具，具有严格的数学定义和图形表达能力，能够有效地描述和分析具有并发、异步、分布、并行和不确定等性质的信息处理系统。下面给出有关 Petri 网的形式化定义。

定义 3(Petri 网) 一个 Petri 网可以表示为四元组 $PN=(P,T,F,M_0)$，其中，

(1)P 是一个有限库所集；

(2)T 是一个有限变迁集，且 $P\cup T\neq\emptyset$，$P\cap T=\emptyset$；

(3)$F\subseteq(P\times T)\cup(T\times P)$是一个弧集；

(4)$dom(F)\cup cod(F)=P\cup T$，其中，

$dom(F)=\{x\in P\cup T\mid\exists y\in P\cup T:(x,y)\in F\}$

$cod(F)=\{y\in P\cup T\mid\exists x\in P\cup T:(x,y)\in F\}$

(5) $M:P\to\mathbf{N}$ 为标识函数，其中 $\forall p\in P$，$M(p)$表示标识 M 下库所 p 中的托肯数，M_0 是初始标识。

定义 4(输入/输出集) 设 $PN=(P,T,F,M_0)$为一个 Petri 网，$x\in P\cup T$ 为网 N 的一个元素，记

$\cdot x=\{y\mid y\in P\cup T\wedge(y,x)\in F\}$为 x 的输入集；

$x^{\cdot}=\{y\mid y\in P\cup T\wedge(x,y)\in F\}$为 x 的输出集；

$\cdot x\cup x^{\cdot}$ 为元素 x 的外延。

为了对服务进行更加直观的描述，基于上述 Petri 网的相关概念以及 Web 服务的定义，将 Web 服务的输入/输出参数提取出来，抽象出服务的网元模型，关于服务网元的定义如下。

定义 5(M 在集合 U 上的投影) 设 $PN=(S,T,F,M_0)$是一个 Petri 网，映射 $M:S\to\mathbf{N}$ 为网 N 的一个标识(marking)，记标识的集合为 $\mathbf{N}^S$，则 $M\in\mathbf{N}^S$，那么给定集合 U，M 在集合 U 上的投影记为 $M_{|U}\in\mathbf{N}^S$，且

$$M_{|U}(s)=\begin{cases}M(s), & s\in S\cap U\\ 0, & s\in U\setminus S\\ \phi, & s\in S\setminus U\end{cases}$$

对于 $\forall n\in\mathbf{N}$，若 $S=\bigcup_{i=1}^{n}U_i$，其中 $\forall j,k\in\mathbf{N}_n$，$U_j\cap U_k=\emptyset$，则 $M=\sum_{i=1}^{n}M_{|U_i}$。

定义 6(服务网元 Service net unit) $Nu=(t,I_t,O_t,F_t,MI,MO)$为一个服务网元(简称网元)，其中，

(1)t 为一个变迁，表示完成特定服务功能的操作；

(2)$I_t=\cdot t=\{s_{i1},s_{i2},\cdots,s_{iu}\}$是 t 的输入库所集，表示服务的输入参数集合，$O_t=t^{\cdot}=\{s_{o1},s_{o2},\cdots,s_{ov}\}$是 t 的输出库所集，表示服务的输出参数集合；

(3)$F_t=(I_t\times t)\cup(t\times O_t)$为连接变迁 t 和库所 s 的弧，$s\in I_t\cup O_t$；

(4)$MI=[m(s_{i1}),m(s_{i2}),\cdots,m(s_{iu})]$，称为网元的使能标识或输入标识，对 $\forall j\in\mathbf{N}_u^+$，$MI(s_{ij})=m(s_{ij})=s_{ij}.constraint$；

(5)$MO=[m(s_{o1}),m(s_{o2}),\cdots,m(s_{ov})]$，称为网元的输出标识，对 $\forall k\in\mathbf{N}_v^+$，$MO(s_{ok})=m(s_{ok})=s_{ok}.constraint$；

(6)变迁 t 在标识 M 下的发生规则：

a)若 $\forall s\in I_t:M(s)\geqslant MI(s)$，则变迁 t 在 M 下有发生权，记为 $M[t>$；

b)若 $M[t>$，则 t 在标识 M 下可以发生，得到一个新的标识，记作 $M[t>M'$，对于 $\forall s\in I_t$，

$$M'(s)=\begin{cases}M(s)+1, & s\in O_t-I_t\\ M(s)-1, & s\in I_t-O_t\ \text{and}\ MI(s)=1\\ M(s), & \text{else}\end{cases}$$

在定义 6 中，服务网元中的变迁对应服务功能，输入/输出库所对应服务的输入/输出参数，输入/输出库所集对应服务的输入/输出参数集合。为了便于表示，我们用服务名称命名网元变迁，用输入/输出参数名称命名库所，这样就建立起了服务网元与服务描述之间的对应关系。此外，用“$MI\to MO$”表示输入/输出参数满足的推导关系。

例如：给定一个机票查询服务 $WS=(BI,SF,BC,I,O,QoS)$，其中 $SF=$(Query, FlightTicket)，$I=\{(i_1$, Departure, 1), (i_2, Arrival, 1), (i_3, Date, 1), (i_4, Type, 0)$\}$，$O=\{(o_1$,

FlightNo,1),(o_2,Time,1),(o_3,Price,1),(o_4,FlightType,1)},则根据定义5可以得出WS的输入标识$MI=[m(i_1),m(i_2),m(i_3),m(i_4)]=[1,1,1,0]$,输出标识$MO=[m(o_1),m(o_2),m(o_3),m(o_4)]=[1,1,1,1]$。图1给出了服务WS的网元表示。

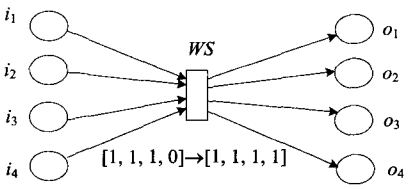

图1 服务WS的网元模型

3 基于服务簇网元模型的服务发现

3.1 服务的分层聚类

目前,已有许多文献提出利用语义相似度计算来实现服务聚类的方法,而关于服务聚类的标准也有所不同,主要有:基于功能相似的服务聚类方法、基于过程相似的服务聚类方法,还有基于服务结构相似性及服务属性相似性等对服务进行聚类的方法[10,11]。而所有的这些聚类方法的目的均是相同的,即使得同一个服务簇中的服务之间的相似度很高,不同服务簇中的服务高度相异。

本文采用分层聚类思想,按照不同的聚类标准对服务注册库中的服务层层聚类,以达到更好的聚类效果。在本文中,服务聚类主要依据服务功能相似性和服务参数相似性进行。当然,为提高聚类的精确性,还可以依据服务的过程相似性、服务属性相似性、QoS相似性等对服务进行深层聚类,具体方法可以参照相关文献。本文提出的基于服务簇网元模型的服务发现方法是建立在服务聚类基础之上的,不完全依赖于聚类方法的选择。

服务功能采用概念对的形式进行描述,且服务的输入/输出都是由一组输入/输出参数进行刻画,而每一个输入参数都对应于给定本体定义中的一个概念。即使在某一领域使用相同的领域本体,不同的服务提供商在描述服务时,所采用的描述方法也不完全相同,因此在对服务进行相似度计算之前,需要先对服务功能描述和参数进行一系列的处理。下面给出相似度计算的具体方法。

(1)计算服务功能相似度

首先,采用文献[12]的功能语义标注机制将服务功能统一描述为本体库中概念构成的概念对形式,然后利用如下公式计算服务WS_1和WS_2的服务功能之间的相似度;

$$\mathrm{Sim}(SF_1,SF_2)=\alpha\mathrm{Sim}(action_1,action_2)+\beta\mathrm{Sim}(object_1,object_2)$$

式中,α和β是权值,$\alpha,\beta\in[0,1]$且$\alpha+\beta=1$。

(2)分别对输入/输出参数进行两两最优配对

由于各服务提供商对服务参数的描述方法不尽相同,因此在计算相似度之前,需要先对服务的输入/输出参数进行配对,来消除服务参数描述中存在的同词异义或异词同义等语义异构问题。具体的配对过程可以参照文献[6]的方法,在此不做赘述。下面仅以两个服务为例,简要说明参数配对的结果。

设两个服务WS_1、WS_2的输入参数分别为$I_1=\{i_{11},i_{12},i_{13},\cdots\}$、$I_2=\{i_{21},i_{22},i_{23},\cdots\}$,输出参数分别为$O_1=\{o_{11},o_{12},o_{13},\cdots\}$、$O_2=\{o_{21},o_{22},o_{23},\cdots\}$,在进行输入/输出参数的相似度计算之前,对两组输入/输出参数进行配对,最终选用参数之间的最优配对,建立I_1和I_2、O_1和O_2之间的关联关系:$\langle I_1,I_2\rangle=\{\langle i'_{11},i'_{21}\rangle,\langle i'_{12},i'_{22}\rangle,\langle i'_{13},i'_{23}\rangle,\cdots\}$,其中$i'_{1u}\in I_1$,$i'_{2u}\in I_2$,$0\leqslant u\leqslant\min\{|I_1|,|I_2|\}$;$\langle O_1,O_2\rangle=\{\langle o'_{11},o'_{21}\rangle,\langle o'_{12},o'_{22}\rangle,\langle o'_{13},o'_{23}\rangle,\cdots\}$,其中$o'_{1v}\in O_1$,$o'_{2v}\in O_2$,$0\leqslant v\leqslant\min\{|O_1|,|O_2|\}$。服务$WS_1$和$WS_2$输入/输出参数的两两最优配对结果如图2所示。

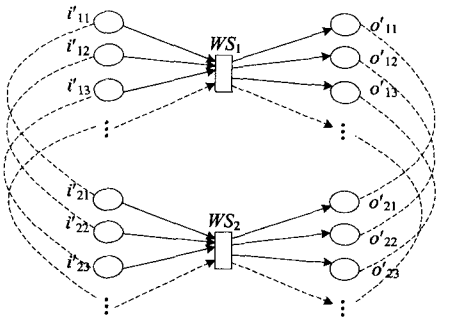

图2 服务WS_1和WS_2参数的两两配对

(3)分别计算输入/输出参数的相似度

在最优配对的基础上,计算两个服务的输入相似度$\mathrm{Sim}(I_1,I_2)$及输出相似度$\mathrm{Sim}(O_1,O_2)$。

(4)计算服务的相似度

综合考虑服务的功能相似性和输入/输出参数相似性,则服务的相似度可以表示为:

$$\mathrm{Sim}(WS_1,WS_2)=\omega_1\mathrm{Sim}(SF_1,SF_2)+\omega_2\mathrm{Sim}(I_1,I_2)+\omega_1\mathrm{Sim}(O_1,O_2)$$

式中,$\omega_1,\omega_2,\omega_3$是权值,$\omega_1,\omega_2,\omega_3\in[0,1]$且$\omega_1+\omega_2+\omega_3=1$。

按照上述相似度计算方法,对服务注册库中的服务进行两两之间的相似性评估,采用现有的聚类方法,如K中心点与聚合聚类相结合的算法,实现对服务注册库中服务的聚类。

3.2 服务簇网元模型的构建

通过服务聚类处理,服务注册库中的服务被划分成多个服务簇,同一个服务簇中的服务具有高度的相似性,而不同的服务簇之间高度相异。为了更好地体现同一个服务簇中服务的共性和个性,从而更有效地进行服务发现,我们给出一种合理的服务簇描述方法。

设$WSC=\{WS_1,WS_2,\cdots,WS_n\}$为一个服务簇,其中$WS_u=(BI_u,SF_u,BC_u,I_u,O_u,QoS_u)$,$SF_u=(action_u,object_u)$。经过以下3个步骤,由服务簇中服务的描述可得到服务簇的描述。

(1)服务簇功能

由于同一簇中服务的服务功能相似度极高,因此采用簇中任意服务的服务功能来描述服务簇功能。

(2)服务簇输入参数集合

基于参数的最优配对关系,通过对输入/输出参数进行统一标注,使得语义相似度极高的参数能够采用相同的本体概念来描述,这可以在很大程度上消除语义异构。因此,给出参数标准化的定义。

定义7(参数标准化) 给定参数集合$S=\{s_1,s_2,s_3,\cdots,$

$s_n\}$，将 S 中具有最优配对关系的参数用相同的本体概念进行标注，称为参数标准化。得到的本体概念集合 S'' 称为参数集合 S 的标注集合，记作 $S_{label}=S''$。

参数标准化过程如下：

Step1　$S=\{s_1,s_2,s_3,\cdots,s_n\}$，$S_{label}=\emptyset$；

Step2　若 $S\neq\emptyset$，则随机选择集合 S 中的任意一个参数 s_u，且 $S_{label}=S_{label}(\{s_u\})$，$S=S-\{s_u\}$；

Step3　扫描集合 S，

(a) 基于参数两两配对，若 $s'\in S$ 是 s_u 的最优配对，则用 s_u. concept 对 s'. concept 进行标注，即 s'. concept $=s_u$. concept。然后，$S=S-\{s'\}$；

(b) 否则，返回 Step2；

Step4　重复执行 Step2、Step3，直到集合 $S=\emptyset$；

Step5　输出 S_{label}。

经过上述过程，不仅完成了参数的标准化过程，使得高度相似的参数用相同的本体概念来标注，同时也得出了集合 S 的标注集合 S_{label}。因此，服务簇的输入集合，即集合 $I=I_1\cup I_2\cup\cdots\cup I_n$ 的标注集合 I_{label}，便可按照上述参数标准化的过程求解；

(3) 服务簇输出参数集合

按照(2)中的方法，求解集合 $O=O_1\cup O_2\cup\cdots\cup O_n$ 的标注集合 O_{label}，即服务簇的输出集合。

为了更清晰、直观地表示服务簇，将给出服务簇网元结构的定义，它与服务网元类似，只是由于多数情况下服务簇中的服务不唯一，导致单个向量不足以表示服务簇的输入/输出参数约束，因此，需采用矩阵形式表示服务簇的输入/输出参数约束，同时表示输入/输出的对应关系。

定义 8(服务簇网元)　$CNU=(NU,t_{I|O},It,Ot,Ft,AI,AO)$ 为一个服务簇网元(简称簇网元)，其中，

(1) $NU=\{Nu_1,Nu_2,Nu_3,\cdots,Nu_n\}$，$n\in\mathbf{N}^+$，是一组网元的集合；

(2) $t_{I|O}$ 是网元簇变迁；

(3) $It=({}^\cdot t_1\cup {}^\cdot t_2\cup\cdots\cup {}^\cdot t_n)_{label}$，$Ot=(t_1^\cdot\cup t_2^\cdot\cup\cdots\cup t_n^\cdot)_{label}$，$It$ 和 Ot 分别表示服务簇的输入/输出集合；

(4) $Ft=(It\times t_{I|O})\cup(t_{I|O}\times Ot)$；

(5) $AI=[(MI_1)_{|(It)},(MI_2)_{|(It)},\cdots,(MI_n)_{|(It)}]^{\mathrm{T}}$ 表示输入矩阵；

(6) $AO=[(MO_1)_{|(Ot)},(MO_2)_{|(Ot)},\cdots,(MO_n)_{|(Ot)}]^{\mathrm{T}}$ 表示输出矩阵；

(7) 变迁 $t_{I|O}$ 在 M 下的发生规则：

若 $\exists M\in\mathbf{N}^{|t|}:M\geqslant(MI_i)_{|(It)}$，$i\in\mathbf{N}_n^+$，则 $t_{I|O}$ 在标识 M 下使能；若 $t_{I|O}$ 使能，则 $t_{I|O}$ 可以发生，且 $M[t_{I|O}>M'$，及 $t_{I|O}$ 发生后，$M'=(MO_i)_{((Ot)}$。

由定义 8 得出，输入/输出矩阵的行(Row)对应于服务簇中的服务，列(Column)对应于服务簇输入/输出参数集合中的参数，且变迁 $t_{I|O}$ 的输入和输出分别受输入矩阵 AI 和输出矩阵 AO 的限制。由于给定标识 M，输入矩阵的多个行向量 $(MI_i)_{|(It)}$，使得 $(MI_i)_{|(It)}\leqslant M$，则变迁 $t_{I|O}$ 能够发生，若 $M[t_{I|O}>M'$，则此时 M' 不唯一。

例如：一个机票查询的服务簇 WSC 中包含 4 个原子服务：WS_1，WS_2，WS_3 和 WS_4，具体描述如表 1 所列。

表 1　WS_1-WS_4 的相关描述

服务	服务功能	输入	输出	参数向量
WS_1	(Query,FlightTicket)	$(i_{11},$Departure$,1),(i_{12},$Arrival$,1),(i_{13},$FlightDate$,1),(i_{14},$FlightClass$,0)$	$(o_{11},$FlightType$,1),(o_{12},$Time$,1),(o_{13},$Price$,1),(o_{14},$FlightCompany$,1)$	[1 1 1 0]→[1 1 1 1]
WS_2	(Check,AirplaneTicket)	$(i_{21},$DepartureCity$,1),(i_{22},$ArrivalCity$,1),(i_{23},$Date$,1),(i_{24},$Price$,0)$	$(o_{21},$FlightNo$,1),(o_{22},$FlightTime$,1),(o_{23},$Type$,1)$	[1 1 1 0]→[1 1 1]
WS_3	(Query,AirTicket)	$(i_{31},$Departure$,1),(i_{32},$Arrival$,1),(i_{33},$Date$,1)$	$(o_{31},$Price$,1),(o_{32},$Time$,1),(o_{33},$FlightType$,1),(i_{34},$FlightSeatings$,1)$	[1 1 1]→[1 1 1 1]
WS_4	(Query,FlightTicket)	$(i_{41},$Date$,1),(i_{42},$FlightType$,1),(i_{43},$Departure$,1),(i_{44},$Arrival$,1)$	$(o_{41},$FlightNo$,1),(o_{42},$FlightAffiliation$,1),(o_{43},$FlightTime$,1),(o_{44},$TicketPrice$,1),(o_{45},$Seatings$,1)$	[1 1 1 1]→[1 1 1 1 1]

依据定义 6，可以给出 4 个服务：WS_1，WS_2，WS_3 和 WS_4 的网元模型，如图 3 所示。

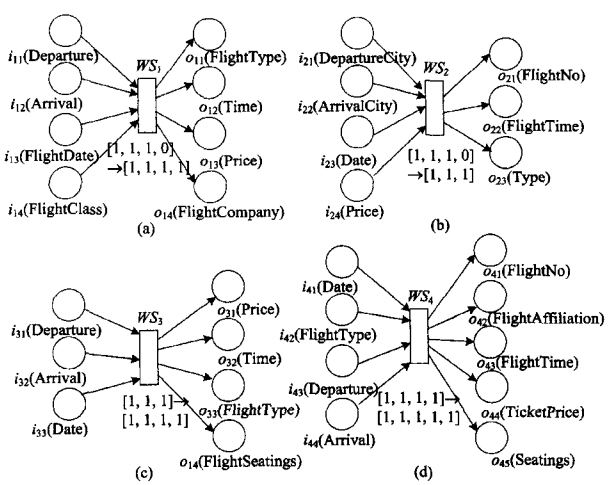

图 3　WS_1-WS_4 的网元模型

假设采用本节所述的参数统一标注方法，对服务簇的输入/输出集合中的参数进行统一标注后，得到的标注结果如表 2 所列，则服务簇 WSC 的网元模型如图 4 所示。

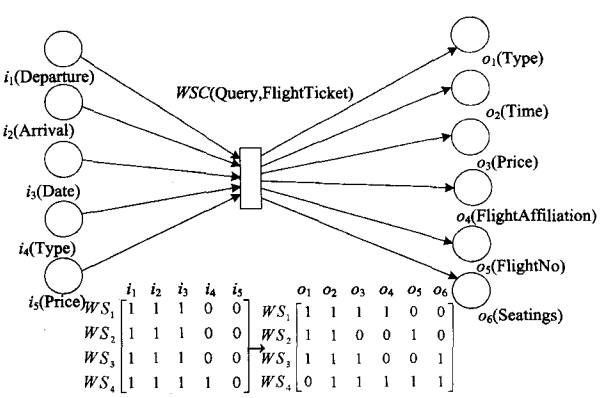

图 4　机票查询服务簇的网元模型

表2 服务簇参数统一标注的结果

	参数配对关系	统一标注
输入参数	{Departure,DepartureCity}	Departure
	{Arrival,ArrivalCity}	Arrival
	{FlightDate,Date}	Date
	{Type,FlightClass,FlightType}	Type
	{Price,TicketPrice}	Price
输出参数	{Type,FlightClass,FlightType}	Type
	{Time,FlightTime }	Time
	{Price,TicketPrice}	Price
	{FlightAffiliation,FlightCompany }	FlightAffiliation
	{FlightNo}	FlightNo
	{FlightSeatings,Seatings}	FlightSeatings

3.3 服务簇参数矩阵规范化

基于语义相似度计算对服务库中的服务进行聚类，形成服务簇，能够更加有效地对服务进行组织和管理，在服务发现阶段，也能够尽早地筛选掉与服务请求差别很大的多个服务，进而缩小服务发现的规模，提高服务发现的效率。在传统的基于服务聚类的服务发现算法中，当服务发现范围锁定为特定的服务簇时，需要将服务请求与服务簇的多个服务逐一进行比较，以进一步选出满足服务请求的服务。

为了便于更快地从特定服务簇中选择出满足服务请求的服务，提出了一种服务簇参数矩阵规范化方法，其将服务簇的输入/输出参数按照一定的顺序进行排列，从而提高服务发现的效率。

定义9(参数频数) 在包含 n 个服务的服务簇参数矩阵中，设 il 为第 l 列对应的参数，则称 $sum_l=\sum_{j=1}^{n}m_j(s_{il})$ 为参数 il 的频数。

对参数矩阵 A 的列顺序进行调整，使得调整之后，矩阵的列是按照列对应参数的频数自左向右非递增顺序排列的，则称这样的矩阵为规范化参数矩阵，记为 $|A|$。求解某一参数矩阵的规范化矩阵的过程，称为参数矩阵规范化。

下面以输入矩阵 AI 为例，给出参数矩阵规范化的具体过程。

设输入矩阵 $AI=[(MI_1)_{|(It)},(MI_2)_{|(It)},\cdots,(MI_n)_{|(It)}]^{\mathrm{T}}$

$$=\begin{bmatrix}(MI_1)_{|(It)}\\(MI_2)_{|(It)}\\\cdots\cdots\\(MI_n)_{|(It)}\end{bmatrix}=\begin{bmatrix}m_1(s_{i1}),m_1(s_{i2}),\cdots,m_1(s_{iu})\\m_2(s_{i1}),m_2(s_{i2}),\cdots,m_2(s_{iu})\\\cdots\cdots\\m_n(s_{i1}),m_n(s_{i2}),\cdots,m_n(s_{iu})\end{bmatrix}$$ 表示服务簇包含 n 个服务，对应于输入矩阵的 n 个行，服务簇输入集合中包含 u 个参数，对应于输入矩阵的 u 个列。

Step1 对矩阵的每一列进行求和，得到每个参数的频数，如第 $l(1\leqslant l\leqslant u)$ 个参数 il 的频数等于矩阵第 l 列的和，即 $sum_l=\sum_{j=1}^{n}m_j(s_{il})$；

Step2 按照 $Sum_l(1\leqslant l\leqslant n)$ 的值，非递增顺序进行排序；

Step3 按照 Step2 的排序结果对矩阵的列顺序进行调整，得到 AI'，且 $Sum_1\geqslant'Sum_2'\geqslant\cdots\geqslant Sum_n'$。

输入矩阵的规范化过程结束，$|A|=AI'$。

对于图4所示的服务簇来说，其输入矩阵本身是规范化矩阵，不需处理。而对其输出矩阵进行规范化，得到输出参数矩阵的规范化矩阵为：

$$\begin{array}{c}\\WS_1\\WS_2\\WS_3\\WS_4\end{array}\begin{array}{c}\begin{array}{cccccc}o_2&o_1&o_3&o_4&o_5&o_6\end{array}\\\begin{bmatrix}1&1&1&1&0&0\\1&1&0&0&1&0\\1&1&1&0&0&1\\1&0&1&1&1&1\end{bmatrix}\end{array}$$

3.4 服务发现算法

在进行服务发现之前，通过对服务注册库中的服务进行聚类预处理，可以大大缩小服务发现的规模，提高服务发现效率。但是现有的基于服务聚类的服务发现算法在对服务簇中的服务进行筛选时，仍然采用的是服务请求与簇中服务逐个匹配的方法，没有充分利用服务簇中服务高度相似的特点。因此，为了更好地体现和利用这一优势，利用上文提出的参数矩阵对特定服务簇中的服务进行筛选，以达到进一步提高发现效率的目的。

基于服务簇网元模型的服务发现算法：

输入：服务簇网元集合 WSC 和服务请求(SF_{req}，I_{req}，O_{req})

输出：满足请求的服务集合 S

1) 初始化：S=∅，δ=设定值；

2) 从 WSC 中任意选择一个服务簇 wsc，WSC=WSC−{wsc}，计算 $Sim(SF_{req},SF_{wsc})$；

3) 若 $Sim(SF_{req},SF_{WSCi})<\delta$，则转 2)；否则，对服务簇 wsc 中的服务进行筛选，假设服务簇 $wsc=\{ws_1,ws_2,\cdots,ws_n\}$ 的输入/输出集合分别为 It 和 Ot；

4) 求解 $MO_{req|(Ot)}$，令 k=v；

(4.1) 如果 $m_{req}(o_k)=0$，则转(4.3)；

(4.2) 如果 $m_{req}(o_k)=1$，且矩阵行数不为零，则将服务簇输出矩阵第 k 列中 0 对应的行删掉，否则转 6)；

(4.3) k=k−1；

(4.4) 若 k≥1，则转(4.1)，否则，转 5)；

5) 求解 $MI_{req|(It)}$，令 j=1；

(5.1) 如果 $m_{req}(i_j)=1$，则转(5.3)；

(5.2) 如果 $m_{req}(i_j)=0$，且矩阵行数不为零，则将服务簇输入矩阵第 j 列中 1 对应的行删掉，否则转 6)；

(5.3) j=j+1；

(5.4) 若 j≤u，则转(5.1)，否则，转 6)；

6) S={输出矩阵剩余行对应的服务}∩{输入矩阵剩余行对应的服务}，输出 S。

4 比较分析

假设一个服务簇中有 n 个服务，服务簇的输入/输出参数集为 It/Ot，服务请求的输入/输出参数个数分别为 $|I_{req}|/|O_{req}|$，则 $u=|(It\cup I_{req})_{label}|$，$v=|(Ot\cup O_{req})_{label}|$，将服务请求的输入、输出请求分别转化为向量表示的形式，并与服务簇中的服务进行匹配。

若采用传统的参数匹配方法，分两步进行服务匹配：首先，输出请求向量与每个服务的输出向量逐一进行匹配，匹配次数为 $n\times v$ 次，删掉不能提供请求者需要的输出的服务(假设为 m 个)；然后对请求输入向量与剩余的每个服务的输入向量逐一进行匹配，匹配次数为 $(n-m)\times u$ 次；最终得到满足

请求的服务。

若采用所提基于参数矩阵的服务发现方法，也通过输出参数匹配和输入参数匹配两步完成服务匹配。首先，对于输出请求向量中0对应的参数不需做匹配处理，对于1对应的参数，按照频数从小到大的顺序取参数，与输出矩阵中对应列中的参数值进行匹配，假设经过第 i 次匹配后，一次删掉的服务个数为 m_i，则共匹配 $n+\sum_{i=1}^{|O_{req}|}(n-m_i)$ 次，即完成了输出参数匹配，得到满足输出请求的服务有 $n-\sum_{i=1}^{|O_{req}|}m_i$ 个。对于输入参数，则输入请求向量中1对应的参数不需做匹配处理；对于0对应的参数，按照矩阵中参数频数从大到小的顺序取参数，与输入矩阵中对应列中的参数值进行匹配，假设经过第 j 次匹配后，一次删掉的服务个数为 m_j，则共匹配 $(n-\sum_{i=1}^{|O_{req}|}m_i)+\sum_{j=1}^{|I_{req}|}(n-\sum_{i=1}^{|O_{req}|}m_i-m_j)$ 次。因此，本文方法的匹配次数要远远小于传统匹配方法。

下面以3.2节给出的机票查询服务簇为例(见图4)，从匹配次数和返回结果两方面对传统的服务匹配和基于服务簇参数矩阵的服务发现进行比较分析。图4所示的服务簇的规范化的参数矩阵为：

$$\begin{array}{c} \\ WS_1 \\ WS_2 \\ WS_3 \\ WS_4 \end{array}\begin{array}{c} \begin{array}{ccccc} i_1 & i_2 & i_3 & i_4 & i_5 \end{array} \\ \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \end{bmatrix} \end{array} \rightarrow \begin{array}{c} \\ WS_1 \\ WS_2 \\ WS_3 \\ WS_4 \end{array}\begin{array}{c} \begin{array}{cccccc} o_2 & o_1 & o_3 & o_4 & o_5 & o_6 \end{array} \\ \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 \end{bmatrix} \end{array}$$

假设给定4个机票查询的服务请求 $WS_{req1}-WS_{req4}$，分别为：

WS_{req1}：(Type, Price)→(Time)

WS_{req2}：(Departure, Arrival, Date, Type)→(FlightNo, Seating)

WS_{req3}：(Departure, Arrival, Date)→(Type, Time, Seating)

WS_{req4}：(Departure, Arrival, Date, Type, Price)→(Type, Time)

则对请求做简单处理，得到其向量表示如下：

WS_{req1}：[0 0 0 1 1]→[1 0 0 0 0 0]

WS_{req2}：[1 1 1 1 0]→[0 0 0 0 1 1]

WS_{req3}：[1 1 1 0 0]→[1 1 0 0 0 1]

WS_{req4}：[1 1 1 1 1]→[1 1 0 0 0 0]

分别采用传统的基于参数匹配的服务发现和本文提出的基于参数矩阵的服务发现方法，从服务簇中发现满足 $WS_{req1}-WS_{req4}$ 的服务、其匹配的次数及返回的服务集合，如表3所列。

表3 匹配次数比较

	传统的参数匹配方法		本文方法	
	匹配次数	返回服务集合	匹配次数	返回服务集合
WS_{req1}	44	$\emptyset$	8	$\emptyset$
WS_{req2}	29	$\{WS_4\}$	7	$\{WS_4\}$
WS_{req3}	29	$\{WS_3\}$	9	$\{WS_3\}$
WS_{req4}	39	$\{WS_1, WS_2, WS_3\}$	7	$\{WS_1, WS_2, WS_3\}$

如表3所列，可以清楚地看到，针对相同的服务请求，两种方法返回的服务集合是相同的，但是提出的基于服务簇参数矩阵的服务发现与传统的基于参数匹配的服务发现相比，其匹配次数大大减少。

结束语 本文定义了服务簇的网元模型，并给出了服务簇参数矩阵的构建及规范化方法，以此为基础的服务发现与传统的基于参数匹配的服务发现相比，当从特定的服务簇中筛选满足服务请求的服务时，其能够逐步缩小服务发现规模，减少参数匹配的次数，进而提高服务发现的效率。

本文的工作只是基于服务簇研究服务问题的开始，还有许多问题有待进一步研究。例如，对于如何更好地体现服务簇中多个服务的个性和共性问题以及服务簇组合问题等，本文只考虑了服务发现问题，没有涉及服务组合。下一步将集中研究服务簇的组合以及某服务是否存在于特定服务簇的判定方法，以便进一步提高服务发现和组合的效率。

参考文献

[1] M cIlraith S A, Son T C, Zeng H. Semantic Web services [J]. IEEE Internet Systems, 2001, 16(2): 46-53

[2] 叶蕾，张斌. 基于功能语义的Web服务发现方法 [J]. 计算机研究与发展，2007，44(8)：1357-1364

[3] Liu Wei, Du Yu-yue, Sun Hai-chun, et al. A Fast Algorithm for Web Service Composition Based on Dynamic Description Logic [J]. Information Technology Journal, 2010, 9(6): 1150-1157

[4] Rajagopal S, Thamarai Selvi S. Semantic grid service discovery approach using clustering of service ontologies [C]// Proceedings of IEEE TENCON 2006. Hong Kong, China, 2006: 1-4

[5] Skoutas D, Sacharidis D, Simitsis A, et al. Ranking and Clustering Web Services Using Multicriteria Dominance Relationships [J]. IEEE Trans. Serv. Comp., 2010, 3(3): 163-177

[6] Nayak R, Lee B. Web service discovery with additional semantics and clustering [C]// Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence. Washington, DC, 2007: 555-558

[7] 孙萍，蒋昌俊. 利用服务聚类优化面向过程模型的语义Web服务发现[J]. 计算机学报，2008，31(8)：1340-1352

[8] Murata T. Petri Nets: Properties, Analysis and Applications[J]. Proceedings of the IEEE, 1989, 77(4): 541-580

[9] Du Y Y, Qi L, Zhou M C. A vector matching method for analysing logic Petri nets [J]. Enterprise Information Systems, 2011, 5(4): 449-468

[10] Simper E. Reusing ontologies on the Semantic Web: A feasibility study [J]. Data & Knowledge Engineering, 2009, 68(10): 905-925

[11] 吴健，吴朝晖，李莹，等. 基于本体论和词汇语义相似度的Web服务发现 [J]. 计算机学报，2005，28(4)：595-602

[12] Shin D-H, Lee K-H, Suda T. Automated generation of composite web services based on functional semantics [C]// Web Semantics: Science, Services and Agents on the World Wide Web 7. 2009: 332-343