CBSD:一种基于Chord的模糊服务发现方法

赵文栋 田 畅 彭来献
(解放军理工大学通信工程学院 南京 210007)

摘 要 针对基于DHT的结构化服务发现方法不支持模糊查找的问题,采用服务聚类技术与结构化服务发现技术相结合的方式,提出了一种基于Bloom filter聚类优化的结构化Web服务发现方法。该方法利用Bloom filter实现服务语义映射并通过服务训练队列实现服务描述聚类特征向量的提取,利用相关性计算实现服务描述的预分类,利用Chord算法实现服务的发布/发现,无需冗余发布,既可保证服务语义相近的服务发布到相同的节点上,又可有效地支持服务的模糊查找,并在此基础之上提出了一种基于Bloom filter的分布式服务组合算法。最后,通过仿真验证了所提方法的可行性。

关键词 服务发现,分布式,聚类,服务组合,模糊
中图法分类号 TP393 **文献标识码** A

CBSD:A Fuzzy Service Discovery Algorithm Based on Chord

ZHAO Wen-dong TIAN Chang PENG Lai-xian
(Institute of Communication Engineering, PLA University of Science and Technology, Nanjing 210007, China)

Abstract The main drawback of the structured service discovery method based on DHT doesn't support fuzzy search in the distributed computing environment. A Bloom filter based structured service discovery method was raised that combines the service clustering and structured service discovery technology. This method uses Bloom filter to represent the service semantics. The clustering feature vectors are got by service training queue. Before the services are published into the chord ring, they are clustered by the relevance among the feature vectors. Without redundancy advertisement, this method can guarantee that the services with similar semantics can be published to the same node and can support fuzzy service discovery. Based on this method, a distributed service composed algorithm was raised. At last, the feasibility of the proposed method was demonstrated by simulation.

Keywords Service discovery, Distributed, Cluster, Service composition, Fuzzy

1 引言

随着Web服务的普及、服务数量的急剧增加,集中式的服务发现架构(如UDDI)已成为系统进一步发展的主要瓶颈。这种集中式的查询机制容易引起单点故障,不适合大规模的服务发现系统。分布式的查询方法具有良好的可扩展性并且可以很好地适应服务的动态性,已经成为目前大规模系统设计的一个必然趋势,并取得了很多研究成果,如结构化的、无结构化的P2P系统或分层分布式体系架构设计思想在目前的网络中获得了广泛的应用。

随着网络规模的不断扩大,基于无结构化的服务发现架构,随着查询深度的不断加深,如随机游走的方式或TTL方式,其查询通信量将会迅速增加。而现有常用的结构化服务查询方式,由于在Overlay层每次查找的耗费基本上是常度数或$O(\log n)$,受网络规模的影响不大,在分布式系统中显示出了巨大的优势。

结构化的服务发现方式,其主流算法采用的是基于DHT的设计思想,算法设计的初衷是在因特网上以P2P的方式实现文件的共享,并得到了广泛的应用。虽然网络上发布文件的数量巨大,但是与服务描述相比,文件的命名有很大的确定性,如一个软件或一部电影,虽然在网上存在重多备份,但是其文件名的变化可能只有几个。而服务/请求的描述则不同,即针对同一个服务/请求,不同的用户可能会使用含义相同而表述不同的词汇描述,由于散列函数的自然特性,即使输入有细微差别,其散列结果也可能相差很大,如果直接使用传统的基于DHT的发布/发现方法,会存在以下问题:

(1)耗费网络存贮资源。同一种服务的不同备份由于描述不同,经散列后会多次发布到网络中不同的节点,这不但会浪费网络的存贮资源,而且,在动态环境下,如果某个服务源失效,很难找到后备服务源,从而影响了服务的共享与效能的发挥。

(2)不支持服务的模糊匹配。设服务描述为ServiceA,服务请求为RequestB,根据描述与请求间的匹配关系,可以把服务匹配情况分为以下4种:

到稿日期:2013-03-12 返修日期:2013-07-12 本文受国家973项目(2009CB3020402),国家自然科学基金项目(61103224),江苏省自然科学基金项目(BK2011118)资助。
赵文栋(1972—),男,博士生,副教授,主要研究方向为计算机网络,E-mail:nj_mouse@163.com。

a)(ServiceA=RequestB)∧(Description(ServiceA)=Description(RequestB))。表示服务所能提供的功能恰能满足用户请求,且具有同样的描述。

b)(ServiceA=RequestB)∧(Description(ServiceA)!=Description(RequestB))。表示服务所能提供的功能恰能满足用户请求,但描述方式不同。

c)RequestB⊆ServiceA。表示请求所需功能是服务的子集。

d)ServiceA∩RequestB=δ。表示服务只能满足请求的部分功能。

如果直接采用描述或请求信息作为散列输入,在基于DHT的查找方式下,只有情况a)可以找到所需的服务。虽然在b)、c)两种情况下,网络中存在可满足用户需求的服务,但是由于描述不同,其生成的散列键值一定不同,且无法保证语义相似的描述生成相似的键值,因此均无法找到对应的服务。

(3)不支持服务组合。为了提高服务的共享程度,服务源在发布服务描述时,一般采用较小的发布粒度,即每个服务的功能相对简单。在服务查询时,很可能出现如下情况:网内存贮的任一单个服务无法满足用户需求,但是几个服务的组合可以。针对这种情况,采用传统的结构化查找方式是很难解决的。

出现以上问题的根本原因在于散列函数的自然特性,即生成的散列键值不包含原始服务描述或请求的语义信息。除了表述完全相同的情况,无法从两个不同的键值推断出原始输入信息源之间的逻辑关系。

在分布式的环境下,基于DHT的服务发现机制,在不牺牲路由效率的情况下使其能很好地支持服务模糊查找,是个极具挑战性的问题。

本文第2节介绍了相关研究工作;第3节给出了相关定义;第4节重点介绍了本文所提出的基于预分类的结构化服务发布(发现)方法;第5节对本文的方法进行仿真实验和性能分析;最后总结了全文并给出下一步工作。

2 相关工作

基于结构化DHT网络,扩展其功能并实现服务的模糊查找,目前已经取得了很多研究成果,主要可分为服务聚合、冗余发布、混合查询3类。下面分别进行论述。

(1)服务聚合方式

其核心思想是将语义上相近的服务映射到相近的Overlay层节点上,以减少每次查找的次数[1-6]。

文献[1]提出了一种d维超立方体的服务索引及聚类机制。然而作者只是将DHT网络作为底层的通信网络,对服务描述向量如何与DHT有效地结合并没有作进一步讨论。文献[7]中提出了一种基于神经网络的服务聚类方法。首先对服务注册中心的服务进行语义分析并获取服务的语义特征向量,然后通过神经网络生成一组标识服务——元服务。当用户查找服务时,首先计算服务请求与各元服务的余弦相似度,然后按相似度进行排序,并返回给用户前N个服务。pSearch[5]则基于CAN网络提出了一种可支持基于内容的全文检索方法。这种方式的缺点是,当关键词数目非常多时,CAN的性能会急剧下降。文献[4]基于Chord提出了一种支持部分前缀匹配的路由算法——Squid。它利用Hilbert空间填充曲线(HSFC)将关键词映射成键值。Squid算法可以有效地支持部分前缀匹配及多关键词的查询。但对前缀匹配的支持程度受到维数D的影响。文献[8]提出的方法的设计思想与文献[4]类似,不同的是文献[8]采用的是Z曲线,其底层的支撑网络是CAN。文献[2,9]采用树的方式来表示各关键词的相关性,并以此生成服务标识,达到服务聚类的目的。但是此类方法要求服务标识需具有良好的前缀匹配特性。

(2)冗余发布/查询

其核心思想是提取描述中的关键词,并构成关键词的集合,在服务发布时每个关键词散列一次。文献[10]基于Chord提出了一种反向索引机制,可实现多关键词匹配。假设服务及查询请求的描述是由多个关键词组成的,其表示形式为$\{k_1,k_2,\cdots,k_t\}$,服务在发布请求时,将查询信息发布到存有键值$\{h(k_1),h(k_2),\cdots,h(k_t)\}$的节点。如果某个服务请求具有$t$个关键词,则其在Overlay网络中会发布$t$个查询请求。而服务只需按其描述任一关键词发布一次。文献[11]提出了一种基于分词的反向索引的办法,即当一个关键词映射到键值时,首先,将关键词分成N片,每片分别进行散列并生成散列值。服务发布后,每个节点都存贮了一组关键词及与此关键词相关联的文件的链接。当查找服务时,同样把查询的关键词分成M片,并依次进行散列、查找。这种算法的中心思想是通过冗余发布的方式实现基于DHT的模糊查找。如果服务描述(请求)的关键词比较多,则会引入很多的服务发布或查找冗余。

(3)混合查询

本类算法采用结构化与非结构的服务查询方法相结合的方式,实现服务的模糊查找[12-14]。

MKey[13]在组织网络结构时,将网络分成两层,首先依据地理位置将网络划分成不同的簇,簇内的节点采用无结构化的组织方式,每个簇有一个簇首,簇首组成Chord环;簇内采用洪泛的方式查找。这种查询方式产生的通信量是与查询中关键词的数目及过滤器的长度线性相关的。如果参数设置不合理,则其性能会变差。

本文采用与文献[10]相同的服务描述模型,并综合考虑动态服务聚类与系统负载两方面的问题,结合前期的研究成果[15],提出了一种基于Chord的分布式服务发现方法。该架构的基本特点是,网络中各注册服务器构成一个逻辑上及物理上完全分布的结构化网络。网络中的服务(请求)只需发布一次,即可实现支持服务基于能力的模糊查找。当用户查找服务时,无需洪泛,依据本文给出的服务发现算法可快速定位可能包含有满足需求的服务器。

3 相关知识

为了方便描述CBSD(Cluster Based Service Discovery)算法,现给出如下定义[15]。

定义1(服务(请求)的Bloom filter描述) 对于服务(需求)描述中的任一属性A_i,通过k个散列函数,将其映射到长度为m的Bloom过滤器向量中,记为$BF^{k,m}(S)$。

定义2(服务与请求的相似度) 服务S与请求Q之间的相似度定义为$L(S,Q)=\frac{|S\cap Q|}{|S\cup Q|}$。其中$|S\cap Q|$表示服务$S$

与请求 Q 之间原子属性相同的个数，$|S\cup Q|$ 则表示二者所包括的所有属性的个数。

定义 3(服务与请求的覆盖度) 服务 S 对请求 Q 的覆盖度定义为 $C(S,Q)=\frac{|S\cap Q|}{|Q|}$。其中 $|S\cap Q|$ 表示服务 S 与请求 Q 之间原子属性相同的个数，$|Q|$ 则表示请求所包括的所有原子属性的个数。

服务 S 与请求 Q，经过相同散列函数组映射后，得到 $BF^{k,m}(S)$ 及 $BF^{k,m}(Q)$，则：

定义 4(B-相似度) 服务与请求基于 Bloom 过滤器的相似度定义为：

$$L_{BF}^{k,m}(S,Q)=\frac{|BF^{k,m}(S)\cap BF^{k,m}(Q)|}{|BF^{k,m}(S)\cup BF^{k,m}(Q)|}$$

定义 5(B-覆盖度) 服务对请求基于 Bloom 过滤器的覆盖度定义为：

$$C_{BF}^{k,m}(S,Q)=\frac{|BF^{k,m}(S)\cap BF^{k,m}(Q)|}{|BF^{k,m}(Q)|}$$

式中，$|BF^{k,m}(Q)|$ 表示过滤器中比特位为“1”的个数。$|BF^{k,m}(S)\cap BF^{k,m}(Q)|$ 表示 $BF^{k,m}(S)$ 及 $BF^{k,m}(Q)$ 中对应比特位均为“1”的个数。$|BF^{k,m}(S)\cup BF^{k,m}(Q)|$ 表示 $BF^{k,m}(S)$ 及 $BF^{k,m}(Q)$ 中对应比特位任一位为“1”的个数。

则有如下定理成立：

定理 1 在相同的 Bloom 过滤器生成条件下，基于相同属性全集生成的服务 S 及请求 Q，如果 $\overline{L}_{BF}^{k,m}(S_1,Q)>\overline{L}_{BF}^{k,m}(S_2,Q)$，则有 $\overline{L}(S_1,Q)>\overline{L}(S_2,Q)$。如果 $\overline{C}_{BF}^{k,m}(S_1,Q)>\overline{C}_{BF}^{k,m}(S_2,Q)$，则有 $\overline{C}(S_1,Q)>\overline{C}(S_2,Q)$。其中 $\overline{X}()$ 代表对应变量的统计平均值。

详细内容，请参见文献[15]。

4 基于预分类的服务发布/发现方法及性能分析

CBSD 方法主要包括两部分的内容，一是服务聚类特征及特征向量的获取，二是基于分类的服务发布/发现，下面分别介绍。

4.1 服务聚类特征向量生成算法

本小节首先给出服务特征向量的定义，然后详细介绍服务聚类及特征向量的获取、动态扩充算法。

定义 6(服务特征向量) 假设按照某种分类标准，将领域内所有服务的原子属性划分为不同的集合，且集合内的原子属性具有很强的聚类特性，则此集合的服务特征向量为集合内所有原子属性的 Bloom filter 表示。

CBSD 中聚类特征及特征向量生成算法的伪代码描述如算法 1 所示。

算法 1 服务聚类特征及特征向量生成算法

Input：一组由 Bloom filter 描述的服务描述训练队列

Output：由分组号及特征服务向量组成的二元组集合

Begin：

1. 调用 K-means 算法，生成服务分类组 n，并给出 Bloom filter 描述的一个剖分 $\{B_1,B_2,\cdots,B_n\}$。其中 B_n 代表服务训练集合划分后的划分结果子集。
2. For i=1 to n
   For j=1 to m
   计算 $S_1\cup S_2\cup\cdots\cup S_m$，并将计算结果以〈n，vector〉形式存入集合 Set，其中 S_i 为集合 B_i 中的服务描述
3. Return set{〈n，vector〉}

End

本算法的基本设计思想是采用一组能代表当前网内主要服务特征的服务描述文件，按照定义 1 生成服务基于 Bloom filter 的描述。然后，采用 K-means 算法，完成服务的聚类操作及服务组特征向量的提取，其聚类的标准采用定义 4。

算法中的第 2 步，通过集合“并”操作获取聚类集合元素基于 Bloom filter 表示的特征向量的正确性，可由定理 2 保证。

定理 2 设服务 $A=\{S_1,S_2,\cdots,S_n\}$，基于 Bloom filter 的表示为 B_A，服务 $B=\{K_1,K_2,\cdots,K_m\}$，基于 Bloom filter 的表示为 B_B，则以下关系成立：

If $C=A\cup B\Rightarrow B_C=B_A\cup B_B$

证明：对于任意的服务 $S\in C$，有 $S\in A$ 或 $S\in B$，则其 Bloom filter 的映射 $B(S)\in B_A$ 或 $B(S)\in B_B$，即 $B(S)\in B_A\cup B_B$。

假设 $B_C\neq B_A\cup B_B$，不失一般性，设某一比特位“1”属于 B_C 但不属于 $B_A\cup B_B$，则此“1”位是由 C 中某一服务映射 S 过来的，由前提条件 $C=A\cup B$ 知，如果 $S\in A$，则 B_A 中对应位应置“1”，如果 $S\in B$，则 B_B 中对应位应置“1”，此“1”一定属于 $B_A\cup B_B$，假设不成立。

综上所述，此定理得证。

由定理 2 可看出，不必获取聚类集合中的各原子服务，只通过集合中 Bloom filter 的“并”操作，即可获取聚类集合中各原子服务的 Bloom filter 表示。

同时为了弥补由于训练队列选取不当，可能会出现的某些服务组所生成的特征向量无法很好地涵盖本组内服务属性的问题，本文采用算法 2 实现对服务特征向量的动态扩展。

算法 2 服务特征向量动态扩展算法

Input：服务基于 Bloom filter 的描述 $BF^{k,m}(S)$

α，B-覆盖度阈值

Begin：

1. for i=1 to N
2. 按定义 5 计算与各服务特征向量的覆盖度，并获取与本服务度 B-覆盖度 β 最大的向量 vector 及组号 m。
3. If β>α 且 β≠1
   则扩充向量 vector=vector∪$BF^{k,m}(S)$
   Else
   构建新的服务特征向量

End

其中，α 的选取为经验值。扩充后的服务特征可通过全局洪泛的方式，如按 Chord 网中各节点标识由小到大的顺序发布，以保证全局统一。

通过对服务特征向量的动态扩充，本算法可保证服务特征向量能有效地涵盖本组内服务的所有原子特性。

采用基于 Bloom filter 的服务特征向量描述本域内服务属性的好处有以下几点：

(1)服务特征向量的生成与组内各服务属性的排列顺序无关。无论服务描述中各关键词的排列顺序如何变化，按定义 1 均生成相同的 Bloom filter 向量。

(2)简化了服务请求基于 Bloom filter 向量的匹配操作。如果域内存在能满足服务请求各项属性的服务，由于服务向量是动态扩充的，则下式一定成立：

$$\overline{C}_{BF}^{k,m}(q,vector)=1$$

另一种生成服务组特征向量的简易办法是通过对领域内服务属性预先划分的方式，直接生成本服务组的特征向量。但是，设计时如果划分不合理，则会对服务的分类带来很大的影响，而本文给出的方法则消除了这个问题。

4.2 服务发布/发现算法

CBSD 中服务发布/发现算法的伪代码如算法 3 所示。

算法 3 服务发布/发现算法

Input：服务请求基于 Bloom filter 的描述

Begin：

1. For i=1 to N

   按定义 5 计算与各服务特征向量的覆盖度，并获取与本服务（请求）覆盖度最大的向量 vector 及组号 m。

2. 计算 m 的散列值 X=hash(m)。

3. 调用 chord 算法，完成服务（请求）的发布。

End

某服务器收到用户提交的服务描述时，首先计算服务所归属的组号，然后按组号生成散列值，散列函数可采用 Chord 算法常采用的 SHA-1，并按 Chord 算法完成服务信息的发布。

按上述发布过程，如果原始服务间具有良好的聚类特性，通过预先分类的方式，服务在网络中无需冗余发布，即可实现分布式环境下服务基于语义的聚类，可保证各服务属性相似的服务一定能发布到 Overlay 层相同的节点上。采用这种方式不但可以滤除大量无关的服务，有效减少精确匹配的服务计算量，提高基于结构化网络的服务查找效率，而且在精确查找不成功的情况下，用户无需重新发起查找请求，在同一 Overlay网络节点可完成服务的模糊查找。

需说明的是，本算法在键值生成的过程中采用的是预分类的组号，而没有用特征向量，其主要考虑是，保证特征向量的动态变化不会影响到以前的发布结果，即如果服务 A 对某服务向量进行了动态扩充，A 仍会发布到本服务组所属的 Overlay节点上，不会因为服务特征向量的扩充而破坏服务的聚合特性。

4.3 面向组合服务的服务发现算法

当服务请求比较复杂，其请求的内容涵盖多个聚类组时，采用算法 3 不会找到合适的服务，即用户所需求的服务是多个服务的组合，且这多个服务分布在不同的服务组中。针对这种情况，本小节对算法 3 进行扩展，提出了一种面向组合服务的服务发现算法，其伪代码描述如算法 4 所示。

算法 4 面向组合服务的服务发现算法

Input：用户服务请求基于 Bloom filter 的描述

Output：满足用户服务请求的服务

Begin

1. For i=1 to N

   按定义 4 计算与各服务特征向量的覆盖度，并获取与本服务（请求）覆盖度最大的向量 vector。

2. If C($B_{request}$, Vector)< 1

   调用算法 3，将服务请求描述文件发送至指定 Overlay 节点，并调用相关精确匹配算法选择最可能满足用户功能请求的 Top M 个服务及服务描述文件。

4. For i=1 to M

   4.1 依据服务请求描述文件及选中的服务描述文件，依据领域功能本体计算所缺失的服务。

   4.2 重复 2—4，直到找到满足用户请求的服务组合。

5. Return 服务组合序列

End

本算法的基本思想是采用贪婪算法的设计思想，当 $C(B_{request}, Vector)<1$ 时，由算法 2 的分析可看出，当前服务组内不存在满足用户请求的服务，即用户请求可能需要分布在 Overlay 层网络中的多个点上的服务组合。因此，本算法首先从最有可能满足用户服务请求的点开始查找，如果在指定的点上不能查到满足用户请求的服务（组合），则将在此点上选中的 N 个最有可能满足用户服务请求的服务（组合）返回给发起服务请求的服务器节点。

此处，返回服务（组合）及描述文件主要是以下原因：

（1）根 Bloom filter 的映射原理[16]，由于其假阳性概率的存在，依据定理 2，可以保证 If $C=A\cup B\Rightarrow B_C=B_A\cup B_B$，但无法保证 If $C=A-B\Rightarrow B_C=B_A-B_B$。即属于 A 但不属于 B 的原子服务的 Bloom filter 映射 B_C，正好等于 B_A-B_B。为了保证递归查找结果的准确性，所缺失原子服务的 Bloom filter 映射需按照原始服务描述文件重新计算。

（2）由于本算法采用的 Bloom filter 映射方式只考虑了服务的某种匹配，而未考虑其 IOPE（Input、Output、Precondition、Effect）的匹配情况，为了最终保证所查找到的服务可以顺利执行，且提高查找效率，在第一次查找时，选择 M 个与用户服务请求最相近的服务（组合），且每次查找不但要完成服务基于功能的匹配，而且要完成服务基于 IOPE 的精确匹配，以保证选择的服务序列可顺利执行。关于精确匹配算法，目前已经有很多研究成果，由于不是本文的研究重点，此处不展开论述。经过递归查找后，由初始发起服务请求的服务器节点将查找结果返回给用户。

下面，以仿真的方式验证本文所提方法的有效性。

5 性能分析

本节采用 matLab 对 CBSD 中相关算法的性能做仿真，需说明的是，本节重点仿真采用 CBSD 方法服务发布后，在结构化 Overlay 网络中服务的聚合特性。由于服务查找算法与服务发布算法采用相同的机制，因此本节未对查找算法做仿真；同理，服务组合算法从其工作过程来看，相当于单次服务查找的一个递归过程，本节也未对其做仿真。

本节仿真参数如表 1 所列。

表 1 仿真参数

全集属性数	分类数	类内服务属性	训练服务数	过滤器长度	散列函数数目	测试服务数
250	10	25	400	400	4	1000

由于本节仿真的目的主要是验证具有良好聚类特性的服务，即依据其生成的 Bloom filter 描述后，是否还存在较好的聚类特性，因此，仿真参数选择了比较理想的情况。如表 1 所列，共描述了 250 个原子服务属性，所有这些服务属性分为 10 个组，每组 25 个。各组的属性关键词在集合中所处的顺序是随机的，其统计特性为均匀分布。在生成训练队列时，每组随机生成 40 个服务，服务间的相似度计算采用定义 2，基于 Bloom filter 的相似度计算采用定义 5，分类算法采用 K-means 法。同时为了便于仿真服务特征向量的动态扩展效果，其特征向量只涵盖组内约 80% 的原子服务特性。

原始服务间聚类情况如图 1 所示。

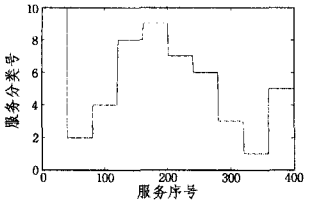

图 1　原始服务聚类情况统计图

经 Bloom filter 映射后，服务间聚类情况如图 2 所示。

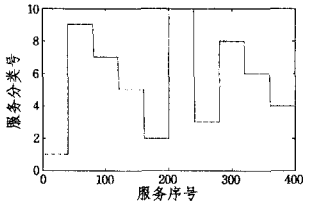

图 2　原始服务映射后聚类情况统计图

在统计的过程中，服务的序号保持不变，由图 1 及图 2 可以看出，无论是原始服务还是经 Bloom filter 映射后，依据服务间的相似性划分均保持了良好的聚类特性，即采用 Bloom filter 的表示方式可以很好地反映服务间的聚类特性。既然生成的 Bloom filter 描述具有良好的聚类特性，是否可以采用文献[2,9]所提方法的设计思想，将其直接应用于结构化的 P2P 网络。下面我们以常用的 Chord 及 Pastry 为例，对算法的聚类特性进行仿真。

图 3 显示的是服务发布完毕后，按 Chord 环中各节点标识由小到大的顺序统计的各节点所存贮服务的情况。其中 Overlay 层的节点数为 20，其键值在$[0,2^{BLen}-1]$空间中均匀分布，其中 *BLen* 表示 Bloom filter 的长度。

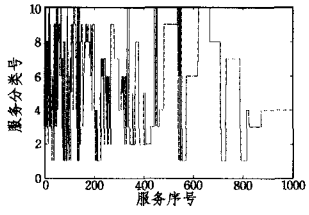

图 3　基于 Chord 算法的服务分布情况统计图

基于 Pastry 算法的服务发布完成后，任取一节点，统计情况如图 4 所示，其 Overlay 层的节点数为 10，标识长度为 *BLen*。

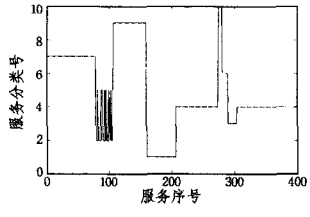

图 4　基于 Pastry 算法的单节点服务分布情况统计图

由统计结果可以看出，虽然服务在发布前具有较好的聚合特性，但是，采用直接映射的方式将其发布到 Chord 或 Pastry 网络中后，在各节点并没有保持良好的聚合特性，即语义相近的服务并没有很好地映射到相同或相邻的 Overlay 网络节点上。经分析，主要原因如下：

(1)Chord 算法其服务的发布是按生成键值的大小顺序排列的。即节点中存贮的服务的键值必须处于本节点的键值与其后继节点的键值之间，而与所表示键值的二进制字符串中"1"的分布无关，如以下情况：

设某节点的散列键值为：0x10000000，某服务 A、B 生成的标识分别是 0x01111111 与 0x10111111，虽然从比特分布情况来看，二者有很大的相似性，但按 Chord 算法，它们会发布到 Overlay 层不同的点上。

(2)Pastry 算法是一种基于前缀的算法，在服务匹配的过程中，考虑到了比特分布的情况，但是对于 * match * 一类的情况，即前缀不同、中间比特分布相同的情况，此类算法无法处理，从而导致了其分布特性差的问题。

服务组内的 Bloom filter 描述由于并不具备很好的偏序特性或前缀特性，因此不能直接应用于此类结构化服务发布算法。

图 5 则显示了采用本文给出的基于预先分类的服务发布方法，在 Chord 环上发布完 1000 个测试服务后，按 Chord 节点标识统计的服务分布情况。每个服务组随机生成 100 个测试服务，且其服务属性与服务特征向量所涵盖的服务存在 0%～10%的差异，α 取值 0.8。

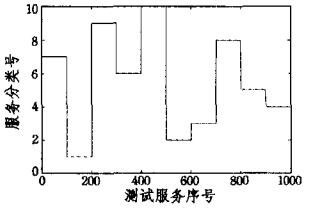

图 5　基于 CBSD 方法服务分布情况统计图

由图 5 可以看出，采用本文给出的算法，在经过预分类处理后，同服务组内服务语义相近的节点均发布到了相同的 Overlay层的服务节点上。

结束语　基于 DHT 的结构化服务发现方法由于受网络规模影响小，在分布式服务发现环境中显示出了巨大的优越性。本文针对基于 DHT 的结构化服务发现方法不支持模糊服务查找的问题，采用服务聚类技术与结构化服务发现技术相结合的方式，在前期研究工作的基础之上，提出了一种基于 Bloom filter 聚类优化的结构化 Web 服务发现方法。该方法首先通过服务训练队列的方式获取网络服务的聚类特征及服务组内获取特征向量；然后将此分类特征与 Chord 算法有机结合，无需冗余发布，即可保证服务语义相同或相近的服务会发布到相同的 Overlay 网络节点，从而有效地支持服务的模糊查找；在此基础之上，对服务特征向量的动态扩充及服务组合算法做了进一步探讨，并采用计算机仿真的方式对本方法的有效性做了验证。下一步将重点考虑本方法在服务动态加入、退出网络情况下的性能及改进方法。

参 考 文 献

[1] Joung Y, Yang L, Fang C. Keyword search in DHT-based peer-to-peer networks[J]. IEEE JSAC, 2007, 25(1): 46-61

[2] Wang T, Di R H. A Semantic Web Service Discovery Model Based on Pastry System[C]// ChinaGrid. 2010: 205-208

[3] Liu L, Ryu K D, Lee K. Supporting efficient keyword-based file search in peer-to-peer file sharing systems[C]// GLOBECOM. 2004: 1259-1265

[4] Schmidt C, Parashar M. Enabling flexible queries with guarantees in P2P systems[C]// IEEE Internet Compu. 2004: 19-26

[5] Tang C, Xu Z, Mahalingam M. PSearch: Information retrieval in structured overlays[C]// ACM SIGCOMM. 2003: 89-94

[6] Rajmohan R, Padmapriya N. A Domain Ontology Based Service Matching for Chord Based Super Peer network[C]// ICDSE. 2012: 214-219

[7] 陈蕾，杨庚，张迎周，等. 基于核 Batch SOM 聚类优化的语义 Web 服务发现机制研究[J]. 电子与信息学报，2011，33(6)：1307-1312

[8] Rosch P, Sattler K, Weth C, et al. Best effort query processing in DHT-based p2p systems[C]// ICDE. 2005: 1186-1189

[9] Szekeres A, Baranga S H, Dobre C, et al. A Keyword Search Algorithm for Structured Peer-to-Peer Networks[C]// SYNASC. 2010: 253-260

[10] Zhu Y W, Hu Y M. Ferry: A P2P-Based Architecture for Content-Based Publish/Subscribe Services[J]. IEEE Transactions on Parallel and Distributed Systems, 2007, 18(5): 672-685

[11] Harren M, Hellerstein J M, Huebsch R, et al. Complex queries in DHT-based peer-to-peer networks[C]// IPTPS. 2002: 242-259

[12] Ganesan P, Sun Q, Garcia-Molina H. Adlib: A self-tuning index for dynamic peer-to-peer systems[C]// ICDE. 2005: 256-257

[13] Jin X, Yiu W P K, Chan S H G. Supporting multiple keyword search in a hybrid structured peer-to-peer network[C]// ICC. 2006: 42-47

[14] Tang C, Dwarkadas S. Hybrid global-local indexing for efficient Peer-to-Peer information retrieval[C]// NSDI. 2004: 25-39

[15] 赵文栋，张进，彭来献，等. 一种基于 Bloom 过滤器的服务模糊匹配算法[J]. 计算机科学，2013，40(3)：175-179

[16] Bloom B. Space/time trade-offs in hash coding with allowable errors[J]. Communications of the ACM, 1970, 13: 422-426

(上接第 155 页)

执行的死活动。

• 有界性：在分析过程组合或者会话协议时，如果库所表示一个状态或者条件，则它包含的 token 的数量要么是 0，要么是 1，否则就意味着出错。如果一个库所表示用来交换消息的缓冲区，则有界性能用来检查缓冲区是否溢出。

组合过程如果违反任何组件服务会话协议将不能有效工作。对会话协议的任何违反将导致 token 陷于对应的颜色 Petri 网中，从而成为整个 Petri 网的死标识。因此，协议的一致性验证可以化简为在颜色 Petri 网中进行死标识的检测。

结束语 我们提出了基于颜色 Petri 网的网格服务复杂会话协议和过程组合的统一模型，颜色 Petri 网比有限状态机和传统 Petri 网更适合这种任务，能帮助网格服务设计者以一种与过程组合中的组件服务的会话协议一致的方式来构造过程组合。由于模型有坚实的颜色 Petri 网的理论基础，模型中的网格服务会话协议和过程组合能用许多技术和模拟工具来进行颜色 Petri 网的设计与验证，这样有利于在设计阶段尽可能早地发现错误并进行纠正。

我们下一步的工作是开发一个能自动导出组合服务的会话协议的算法，并基于 Petri 网标识语言 PNML[21] 对 WSDL 进行扩展以使其包含复杂会话协议规范。

参 考 文 献

[1] Globus Alliance, IBM, HP. Web Service Resource Framework [OL]. http://www.globus.org/wsrf, 2004-06

[2] Orriens B, Yang J, Papazoglou MP. A Framework for Business Rule Driven Service Composition[A]// Proceedings of 4th International Workshop on Conceptual Modeling Approaches for e-Business Dealing with Business Volatility, LNCS 2819[C]. Berlin: Springer-Verlag, 2003: 14-27

[3] Jensen K, Kristensen L M, Wells L. Coloured Petri Nets and CPN Tools for Modelling and Validation of Concurrent Systems [J]. International Journal on Software Tools for Technology Transfer, 2007(9): 213-254

[4] CPN tools[EB/OL]. http://www.cpntools.org/, 2013-09-15

[5] Design/CPN[EB/OL]. http://www.daimi.au.dk/designCPN/, 2013-09-15

[6] 岳昆，王晓玲，周傲英. Web 服务核心支撑技术：研究综述[J]. 软件学报，2004，15(3)：428-442

[7] Krishnan S, Wagstrom P, Laszewski G. GSFL: A workflow framework for grid services[R]. Technical Report, ANL/MCS-P980-0802. Argonne National Laboratory, 2002

[8] 张磊，苑伟政，王伟. 基于领域本体的制造网格服务自动组合技术研究[J]. 计算机应用，2006，26(1)：57-60

[9] 吕庆中，刘梅彦，麦中凡. GSCoL：OGSA 框架下的网格服务组合语言[J]. 计算机工程与应用，2004，40(3)：7-11，44

[10] Narayanan S, McIlraith S. Analysis and simulation of Web services[J]. Computer Networks, 2003, 42(5): 675-693

[11] Peterson J L. Petri Net Theory and the Modeling of Systems [M]. Englewood Cliffs: Prentice-Hall, 1981

[12] Murata T. Petri nets: Properties, analysis and applications[J]. Proceedings of IEEE, 1989, 77(4): 541-580

[13] 蒋昌俊. Petri 网的行为理论及其应用[M]. 北京：高等教育出版社，2003

[14] 袁崇义. Petri 网原理与应用[M]. 北京：电子工业出版社，2005

[15] 林闯. 随机 Petri 网和系统性能评价(第 2 版)[M]. 北京：清华大学出版社，2005

[16] 吴哲辉. Petri 网导论[M]. 北京：机械工业出版社，2006

[17] Aalst WMP. Verification of Workflow Nets[A]// Application and Theory of Petri Nets 1997, LNCS 1248[C]. Berlin: Springer-Verlag, 1997: 407-426

[18] Zhai Zheng-li, Zhou Lei, Yang Yang, et al. A Multi-agent Framework for Grid Service Workflow Embedded with Coloured Petri Nets[A]// Proceedings of 4th International Conference on Grid and Cooperative Computing, LNCS 3795[C]. Berlin: Springer-Verlag, 2005: 117-122

[19] Zhai Zheng-li, Yang Yang, Tian Zhi-min. A Multi-agent based Grid Service Discovery Framework Using Fuzzy Petri Net and Ontology[A]// Proceedings of 8th Asia Pacific Web Conference, LNCS3841[C]. Berlin: Springer-Verlag, 2006: 911-916

[20] Adam N, Alturi V, Huang W K. Modeling and Analysing of Workflows Using Petri Nets[J]. Journal of Intelligent Information Systems, 1998, 10(2): 131-158

[21] Hu X, Hu M, Liu S. Petri Net Markup Language[J]. Computer Technology and Development, 2011, 21(12): 66-69