ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.17, No.10, October 2006, pp.2106−2117
DOI: 10.1360/jos172106
© 2006 by *Journal of Software*. All rights reserved.
E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

基于匹配路径和概率平衡树的 P2P 语义路由模型*

许立波+, 于 坤, 吴国新

(计算机网络与信息集成教育部重点实验室(东南大学),江苏 南京 210096)

A P2P Semantic Routing Model with Match Path and Probability Balance Tree

XU Li-Bo+, YU Kun, WU Guo-Xin

(Key Laboratory of Computer Network and Information Integration, Ministry of Education (Southeast University), Nanjing 210096, China)

+ Corresponding author: Phn: +86-25-83792360, E-mail: xu_libo@163.com, http://www.seu.edu.cn

Xu LB, Yu K, Wu GX. A P2P semantic routing model with match path and probability balance tree. *Journal of Software*, 2006,17(10):2106−2117. http://www.jos.org.cn/1000-9825/17/2106.htm

Abstract: Semantic routing is one of key parts in P2P routing researches. The intelligent search mechanisms support flexible semantic expression but hold low scalability and recall rate. Contrastively, semantic overlay network is scalable but either is difficult to organize or take too large maintenance spending. This paper proposes a new P2P semantic routing model in which node array is organized by match path and probability balance tree, and then an approximately balance distributed structure is obtained. All nodes will take routing decision according to query content while limiting maintenance spending to relative low level. The model supports flexible semantic search and high scalability, and ensures that each node can reach any corner of the network. The model runs with no center service in which all nodes simultaneously take index storage and data storage, and forward task to share system's running load by only maintaining a little local information.
Key words: semantic routing; P2P; match; probability; balance tree

摘 要: 语义路由是 P2P 路由技术的关键研究内容之一.智能化路由策略语义表达灵活,但可扩展性和查全率较低;语义覆盖网络可扩展性好,但要么难以组织,要么维护开销很大.提出一种新的基于匹配路径和概率平衡树的 P2P 语义路由模型(match path and probability balance tree,简称 MPPBTree),通过层次化和匹配路径组织资源存储结构和节点排布方式,达到一种近似平衡的分布特征,使节点能够根据查询内容本身进行路由决策,并同时保持较低的维护开销.模型支持灵活的语义搜索,拥有良好的可扩展性,保证任意节点的路由都能覆盖全网络.模型不要求任何中心服务的存在,所有的节点只需维护少量局部信息,且都会同时承担索引、存储、中继的功能,以均摊系统运行的负荷.
关键词: 语义路由;P2P;匹配;概率;平衡树
中图法分类号: TP311 文献标识码: A

作为 P2P 网络的核心问题,路由一直是制约系统可扩展性和搜索效率的关键因素.研究者们提出了很多可行的路由方案,如:泛洪(Gnutella)[1]、中心索引(Napster)[2]、分布式哈希表 DHT(Chord[3],CAN[4])、前/后缀地址

* Supported by the National Development and Reform Commission High-Tech Research and Development Plan of China under Grant No.CNGI-04-16-18 (国家发改委高技术研究发展计划)

Received 2005-08-24; Accepted 2005-12-14

匹配(Pastry[5],Tapestry[6])等等.这些技术在搜索效率、查询模糊性、维护开销以及可扩展性上都各有优缺点.近年来的一些研究力图依据查询内容本身及节点的知识更新来提高路由性能[7],以同时改善搜索行为在广度和时间上的效率,这被称为语义路由,即指查询请求根据其内容本身来进行路由决策.然而,路由方法往往与系统的组织结构密切相关,良好的路由模型在提高路由效率的同时,还要满足可维护性和可扩展性要求.

1 相关研究与讨论

语义路由技术已经形成了两种思路:一种是以语义覆盖网络(SON)为基础的路由策略[8−10].语义覆盖网络表达了特征值(如逻辑距离、哈希值)相近的节点在语义上也是相近的这一概念.PeerSearch[9]通过 VSM 和 LSI 的方法将文件描述矢量转换成 CAN 上的语义覆盖网络,把语义查询直接对应到语义空间,但它对资源的矢量化过程的代价不菲;还有以层次化概念的方法来实现 SON[10],但在初始分层时需要较多的人为参与.另一种是以智能动态路由为基础的路由策略[7,11−14],大多建立在类 Gnutella 的网络结构上,每个节点维护一个本地路由表(知识库),维护行为是一个动态的不断学习的过程,表达自己对网络中资源分布情况的理解.随机游走[11]通过概率分析来构建节点的本地知识库,使查询泛洪带有一定的目的性,文献[13]也表达了相近的思路;Neurogrid[12]通过检查经过包的内容,使节点学习资源分布情况以本地知识库能够提供的查询关键字数量评价该节点的能力.这类方法能够支持模糊关键字查询,但可扩展性不强,其智能会随着网络规模的扩大而逐渐降低.

语义路由的首要问题就是语义的表达.关键字查询是目前最为普遍和重要的查询方式.而在 DHTs 中,只有精确匹配查询(exact-match queries)得到很好支持.虽然通过诸如建立流行词库的方法能够部分地解决这类问题,但节点因此而付出的代价仍难以让人接受.相比之下,类 Gnutella 系统能够提供关键字查询以及其他更复杂的查询模式[15].我们的系统没有使用 Hash 方法,在查询模式上支持关键字查询甚至通配符语义查询,以保证查询的灵活性.

层次化的方法是实现语义路由的一种显见方案.通常,层次的划分都与语义相关.分层算法一般有两种:事先分层[8,10]和动态分层[9,14].前者是已事先规范好层次的各种性质并不再变更,新到的资源和节点对号入座,算法适用于实现特定功能或只含特定资源的网络中;后者是随着网络的变化而实现动态分层,然而,实现动态语义聚类的功能较为复杂和困难,分类也并不一定精确.我们采用事先分层的策略,但这种层次结构具有普遍适用性,新节点的加入过程也较简单;而在动态分层中,其加入要经历一些诸如分类识别等复杂步骤.

在无结构化 P2P 系统中,节点间连接往往可以动态改变而不是一种静态配置,如节点邻居可以随着节点自己兴趣的改变而加以更新,这能改善类 Gnutella 系统的路由效率[15],也是智能动态路由思想的基点.但用户兴趣的变化及查询语义的多义化会使得智能路由经常失效,网络规模的扩大更使智能路由往往限于局部最优或局部搜索的境地,造成搜索盲区的存在而降低召回率.我们的系统规范化组织节点间连接,使得连接处于一种较为稳定的状态,从而使路由也能稳定存在,并且每个节点的路由都能覆盖全网络.

本文提出了一种基于匹配路径和概率平衡树的语义路由方案 MPPBTree(match path and probability balance tree),节点按照一定的语义规律分层排布,资源定位通过对查询语义的逐步匹配来达到,节点的动态行为遵循概率平衡的原则,有效降低了维护开销.系统不要求任何中心服务的存在,以全分布式形态运行,但所有节点不需要维护任何全局信息,只需同时承担索引、存储、中继的功能,以均摊系统运行的负荷.

2 系统结构及路由机制

2.1 匹配路径

结构化系统和无结构化系统的一个重要区别是路由的确定性(即在此路由下资源必然会被搜索到).结构化系统大多带有内容编址的特性,而后者即使使用智能动态路由,其作用也旨在减少候选路由,并不能保证路由的确定性.我们采用一种关键字匹配路径的方法实现内容编址,类似于前缀匹配,但它是直接将资源关键字映射到资源索引位置,查询关键字采用与资源关键字相同的语义表达形式.这样,查询关键字就与索引位置产生对应关系.如图 1 所示,节点 1~节点 7 分别存放一些字母相关的资源,节点的路由表中只可以看到相邻节点的资源内容.

所不同的是:节点 1,2,3 和节点 1,4 的资源内容存在前缀相关性;节点 1,5,6,7 则内容相互无关.当一个针对 ABC 的查询到达节点 3 时,我们能够确信它一定曾通过节点 1,2,因为内容的相关性使得节点 1→节点 3 的路由唯一而确定;反之,该查询到达节点 1 时,虽然节点 1 并不与节点 3 相邻,但确定性路由使节点 1 知道应该中转到节点 2;而当一个针对 GH 的查询到达节点 1 时,节点 1 无法通过相关性推算出节点 5 可以到达节点 7,所以只能泛洪;反之,该查询到达节点 7 的路由也是不确定的.因此,如果我们把资源的相关性扩展到整个网络,那么全部的路由都将是确定的,又这种相关性是和查询的语义相对应的,那么语义路由就能够实现.

我们先给出一些形式化定义,称查询的内容为查询关键字(query key,简称 QK),称资源的命名为资源关键字(resource key,简称 RK).RK 与 QK 采用相同的语义表达,存在直接映射关系;称节点的命名为节点关键字(node key,简称 NK),每个节点在进入时会得到一个它的 NK,且只存放与此 NK 相同或以之为前缀的 RK 索引.根据上面的模型,如果在一条路径上所有相邻节点的 RK 都存在相关性,那么任一节点就能够推算到达其他节点的下一步路由,且该路径不存在环路,称该路径为匹配路径(match path).易见,由于资源的相关性和语义映射关系,匹配路径上的节点能够直接根据 QK 进行路由.例如图 1 中节点 1↔节点 3、节点 1↔节点 4 都是匹配路径.

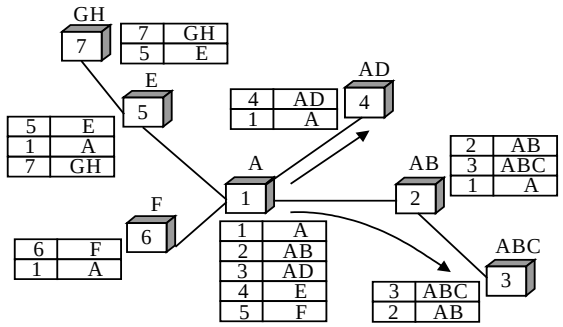


Fig.1 Deduce forward path by content correlation

图 1 通过相关性推算语义路由

2.2 系统结构

MPPBTree 类似于 trie 树,采用层次化模型,树中的每个节点被标记字母符号.查询的消耗只取决于 QK 的长度,而任何 QK 都是有限长度,因此耗时不是不可预知的.MPPBTree 的根节点是网络的发起者,不标识任何 NK,只承担路由功能.① NK 分配:每个节点进入时被父节点标记一个字母,该字母与父节点的 NK 组成本节点的 NK.字母由父节点随机标识,父节点由进入节点自己随机选择.② 索引存储:节点只存放自己 NK 所对应的 RK 的资源索引,包括与 NK 完全一致的 RK 的资源索引及以 NK 为前缀的 RK 资源索引.③ 资源发布:节点对外发布包含 RK 的通告,通告根据 RK 进行语义路由,找到与 RK 对应的 NK 节点的位置.系统结构如图 2 所示.

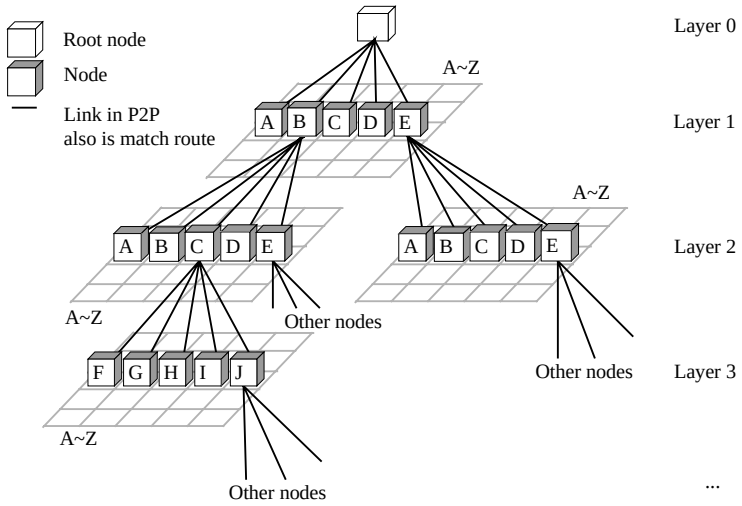


Fig.2 System structure

图 2 系统的总体结构

图 2 中,处于 layer 3 的被标记为 F 的节点,其父节点 NK 为 BC,因此,它的 NK 是 BCF,存放 RK 为 BCF 的资源的索引.如果它没有子节点,则它同时存放以 BCF 为前缀的 RK 的资源索引;如果它有子节点 A,则该子节点的 NK 为 BCFA,存放 BCFA 及以其为前缀的 RK 的资源索;如果它有全部 26 个子节点 A~Z,则它将只存放 RK 为 BCF 的资源索引.

性质 **1**. 设层 L_i 高度为 i (i=0,1,2,…),则 MPPBTree 中任一非根节点到根节点的路径满足

$$length(NKL_i)-length(NKL_j)=i-j\ (i,j=0,1,2,\ldots) \tag{1}$$

$$prefix(NKL_i,NKL_j)=\begin{cases}NKL_i & (i\le j)\\ NKL_j & (i\ge j)\end{cases}\ (i,j=0,1,2,...) \tag{2}$$

$$length(prefix(NKL_i,NKL_j))=\begin{cases}i & (i\le j)\\ j & (i\ge j)\end{cases}\ (i,j=0,1,2,...) \tag{3}$$

性质 **2**. MPPBTree 中节点的度(degree)最大为 27.

2.3 语义路由的形式证明

MPPBTree 是基于分布式环境的查找树,节点是高度自治的,路由的发起点可以是网络中任何一个节点,而所有节点都只拥有局部信息.在 MPPBTree 中,节点只维护父子节点信息,不需要知道同层或跃层的任何其他信息.因此,怎样利用少量局部信息来实现全局的路由是必须要解决的问题.图 2 中节点 BC 的路由表结构如下:

Node	Layer	Count	Node key	IP address
Self node	2	0	BC	****
Parent node	1	0	B	****
Child node	3	0	BCF	****
		…	…	…

MPPBTree 的路由遵循两个法则:

(1) 最大前缀匹配.节点选择与待查 Key 前缀匹配长度最长的 NK 子节点作为下一跳路由;

(2) 父节点优先.若路由表中所有 NK 与待查 Key 匹配长度相同,则选择父节点作为下一跳路由.

性质 **3**. MPPBTree 中任一非根节点到根节点的任何一段路径都是匹配路径.

证明:设任意节点 NKL_i 要路由到任意 NKL_j 节点的位置,首先取父节点 NKL_{i-1} 和子节点 NKL_{i+1} 与 NKL_j 作前缀匹配,如果 $i<j$,由式(3)得到前缀匹配最大长度是 i+1,根据路由法则(1)选子节点为下一条路由;如果 $i>j$,由式(3)得到前缀匹配最大长度是 j,根据路由法则(2)选父节点为下一条路由.如果 $i=j$,则本节点即为所求.另外,MPPBTree 节点只维护上下游节点信息,不构成任何环路. □

性质 **4**. MPPBTree 中任意两个节点间的路径是匹配路径.

证明:两个节点总有共同的祖先节点,根据性质 3,它们到祖先节点的路径都是匹配路径,两条路径上所有 NK 的共同前缀就是祖先节点 NK.易知:祖先节点能够转发两条路径上的相互请求,这样两条路径就连接成完整的匹配路径.这一性质说明:在 MPPBTree 内,所有连接都是匹配路径,因此形成匹配路径覆盖网络,从形式上证明了语义路由的实现. □

2.4 查询路由算法

在系统中,除根节点以外的所有节点都有可能同时承担存储资源文件、存储资源索引和路由转发的任务,路由的基本过程是查询节点根据 QK 决定下一跳路由,收到查询的节点判断 QK 是否和自己的 NK 一致,若不一致,则继续进行语义路由.以下我们分别针对几种不同情况予以阐述:

算法 **1**. 单 QK 精确查询算法(arithmetic for single QK exact match query).

L: layer of node in MPPBTree, $0\le L\le$height of MPPBTree

n: number of child of node, $0\le n\le 26$

take QK from query message;

strcpy(*p*,*prefix*(*QK*,*NK*));

```
if length(p)>0{
    if p==QK && p==NK              //the node is goal
        return all index of NK resource in node; return;
    if p==QK && length(p)<L
        forward query to parent; return;
    if p==NK && length(p)<length(QK){
        strcpy(p,QK,length(p)+1);
        for i=1 to n{    //search in children of the node
            if child[i].NK==p{    //get child NK with longest prefix
                forward query to child[i]; return;}}
            return all index of QK resource in node; return;}
        forward query to parent; return;}
forward query to parent; return;
```

算法 **2**. 单 QK 模糊查询算法(arithmetic for single QK fuzzy match query).

这种查询是最常用的匹配查询(如 Web 搜索引擎),返回结果是所有满足 $RK \supseteq QK$ 资源的索引,系统还允许用户指定候选 RK 长度,以增强搜索效率.

```
C: length of candidate RK specified by user
take QK and C from query message;
if query from parent
    return all index of RK resource which meet RK⊇QK;
    if length(NK)>=C
        return;
    forward query to all children; return;
forward query to parent; return;
```

算法 **3**. 多 QK 精确查询算法(arithmetic for multi QK exact match query).

查询源节点的同时,就每个 QK 发送精确查询,最后对返回结果作交集,其他节点在路由上无须作任何改动.

算法 **4**. 多 QK 模糊查询算法(arithmetic for multi QK fuzzy match query).

查询源节点的同时,就每个 QK 发送模糊查询,最后对返回结果作交集,其他节点在路由上无须作任何改动.

算法 **5**. 带通配符的 QK 查询算法(arithmetic for wildcard QK match query).

这种查询模式同时带有模糊性和指导性,比普通的模糊查询算法效率更高,更为精确,系统的消耗也较少.根据前面的算法容易推出,限于篇幅这里不再给出伪码.

3 系统维护及负载特性

3.1 概率平衡树

在现有的研究中,树状的节点组织结构在维护时大多要求保持平衡树的形式,这样的结构比较规范整齐,避免诸如单枝树的异态结构产生[16].在 MPPBTree 中,节点要加入时需要知道网络的任一节点,通过该中介节点申请加入网络,如果采用简单的把新节点直接作为中介节点的子节点的策略,就有可能造成异态结构.由马太效应,一个中介节点越有名,就越容易得到新节点去发展分支,而分支越大它就越有名.这样会压制其他分支的正常生长,这种负载的不均衡将极大地降低搜索的效率,并且增加维护成本.然而在 P2P 环境下,维护节点平衡树很困难[16]:首先,中介节点可能是系统中的任何节点,它必须知道新来节点在平衡树中应处的位置,这个位置信息是一种全局信息,而 P2P 节点的自治性和暂态性使得每个节点获得任何全局信息都是困难的;其次,退出节点同样可能是系统中的任何节点(而不仅仅是叶节点),这就涉及寻找退出节点的合适替代者及各种存储信息的转

交,而合适的替代者通常也属于一种全局信息.

考虑到我们的目的是避免异态结构产生的同时尽量减少全局信息的产生,而 MPPBTree 并不需要完全遵守平衡状态,也就是说,只要不过度失衡就是可接受的.设想:如果我们的层次树只是从概率上讲是平衡的,即处于不平衡状态的可能性较小,那么,这种概率平衡树同样能达到目的.概率平衡树有两个特征:(1) 同层 NK 位置被分配到节点的概率是一样的;(2) 层深相差 2 以上的 NK 位置分配到节点的概率相差很大.即满足

$$P(NKL_i)_1=P(NKL_i)_2=\ldots=P(NKL_i)_m \tag{4}$$

$$P(NKL_i)>>P(NKL_{i+k})\ \ (k\geq 2) \tag{5}$$

其中,$P(NKL_i)_j(j=1,2,\ldots,m)$为第 L_i 层的第 j 个 NK 位置被分配到节点的概率.式(4)保证了树在横向增长的平衡性;式(5)则保证了树在纵向增长的平衡性.

3.2 节点的加入

为生成概率平衡树,我们设计一种逐次随机探测(step random probe)算法,其过程如下:

(1) 新节点向网络中任一节点(中介节点)提出加入请求,并随机产生一串足够长的数字 δ 作为它的初始 NK,中介节点转发请求给 NK 为 p 的节点,其中,

$$p=strcpy(p,\delta,length(p)+1) \tag{6}$$

(2) 被请求节点随机选择一个空位子节点,设其 NK 为 β,并安插新节点到该位置,转移相关资源索引并更新路由表.新节点接收相关资源索引并构建路由表,设定 β 为本节点 NK,加入过程终止;若被问询节点已拥有满额的子节点,则它再转发请求给 NK 为 p 的节点,返回到步骤(2).

可以看到,在加入过程中,新节点能够随机选择父节点,而一旦选定父节点,其具体的位置将由父节点决定.这类似于在应聘工作时,单位可以任意选择,但具体的职位则由单位决定,是一种半自愿半强制的作法.算法的显著优点是:节点加入过程不需要知道任何全局信息,而且仅涉及新节点及其父节点的路由表更新,不产生任何批量的状态更新通告消息,因此对整个网络状态的影响非常小.算法还保证整棵树以近似平衡的形态增长,避免了异态结构的形成.

性质 **5**. 新节点的初始 NK 与其实际 NK 的最长匹配前缀就是其父节点的 NK,即满足

$$prefix(\delta,\beta)=NK_{parent} \tag{7}$$

其中:δ 为新节点的初始 NK; β 为实际 NK.

证明:如图 3 所示,假设新节点初始 NK 为 AABB*,AABB 位置已有实际节点存在,而 AABC 是空位节点,当请求到达节点 AAB 时,根据逐次随机探测算法新节点被安放到 AABC 位置,新节点的 NK 变更为 AABC,它与 AABB*的最长匹配前缀就是其父节点 AAB.可以看到:算法的逐次特性使 NK 长度依序增长,保证了节点按层级依次排列的模式. □

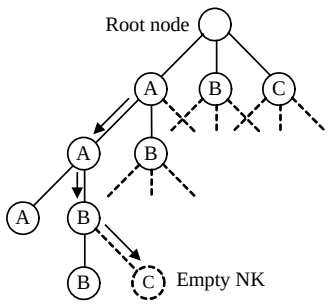


Fig.3 Node with original NK of AABB* being AABC

图 3 初始 NK 为 AABB*的节点成为 AABC

性质 **6**. 按逐次随机探测算法生成的树是概率平衡树.

证明:(1) 设 L_i 层节点数为 n,L_{i-1} 层节点数为 m,$n=26m$,不失一般性,

设父节点$(NKL_{i-1})_1$有空位子节点$(NKL_i)_1,(NKL_i)_2,\ldots,(NKL_i)_{26}$,由于父节点是随机选择一个空位子节点$(NKL_i)_j$分配给新节点,因此可得

$$P((NKL_i)_1|(NKL_{i-1})_1)=P((NKL_i)_2|(NKL_{i-1})_1)=\ldots=P((NKL_i)_{26}|(NKL_{i-1})_1)=p\ (0<p<1).$$

即同一父节点下的空位子节点被分配到新节点的概率是一样的.又父节点 NK 是由新节点随机产生的,则

$$P(NKL_{i-1})_1=P(NKL_{i-1})_2=\ldots=P(NKL_{i-1})_m=q\ (0<q<1).$$

故得

$$P(NKL_i)_j=P(NKL_i)_j(NKL_{i-1})_1)=P((NKL_i)_j|(NKL_{i-1})_1)P(NKL_{i-1})_1=pq\ (j=1,2,\ldots,26).$$

同理可得

$$P(NKL_i)_1=P(NKL_i)_2=\ldots=P(NKL_i)_n=pq.$$

(2) 不失一般性,设$P(NKL_{i+1})_1>0$.这需要满足一个条件,即节点$(NKL_{i-1})_1$具有满额子节点.如图 4 所示中节点 AAA 要得到被分配子节点的机会,则节点 A 一定具有满额子节点.而这个概率是

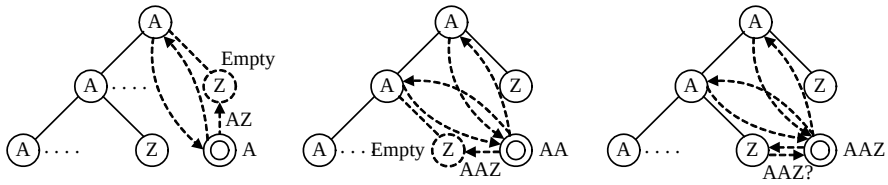
$$P_{full}=P(NKL_i)_1P(NKL_i)_2\ldots P(NKL_i)_{26}=(pq)^{26}.$$

故得到

$$P(NKL_{i-1})=q>>P(NKL_{i+1})=(pq)^{26}.$$

由于$(pq)^{26}<<1$,式(4)、式(5)得到满足.因此,按逐次随机探测算法生成的树是概率平衡树. □

节点加入的例子:如图 4 所示,假设新节点产生的 NK 为 A,通过中介节点与节点 A 联系,节点 A 随机选择一个空的子节点 NK,假设是 AZ,于是新节点成为节点 AZ,如图 4(a)所示;若 A 节点没有空位子节点,则新节点再随机产生一个字母,假设仍是 A,并与前面的 NK 组成新 NK 为 AA,通过中介节点与节点 AA 联系,节点 AA 随机选择一个空的子节点 NK,假设是 AAZ,于是新节点成为节点 AAZ,如图 4(b)所示;若 AA 的子节点也满额,则新节点继续产生 NK 进行探测,如图 4(c)所示.新的节点在加入后,需要保存父节点地址和自己的 NK,接收父节点转移给自己的资源索引,最后将本地资源索引按 RK 通过语义路由通告对应节点.



(a) Node being 19
(a) 节点成为 AZ

(b) Node being 119
(b) 节点成为 AAZ

(c) Node being 119*
(c) 节点成为 AAZ*

Fig.4 Node join

图 4 节点加入

3.3 节点的离开

如果是在平衡树的结构下,叶节点的离开相对容易处理一些;而非叶节点离开的处理就困难很多,通常是通过一些方法将问题转嫁到叶节点上,例如,寻找叶节点替代者或指定某个子节点替代而引起系列节点的更替等,而且都必须以不破坏平衡树的结构为前提.这些方案只是单纯从树的角度考虑问题,而没有结合 P2P 网络的特点进行研究.首先,P2P 节点都是自治的、松散耦合的,因此获得全局信息很困难,联动的信息转移也不现实;其次,MPPBTree 具有特殊性,它并不需要严格平衡,而逐次随机探测算法使新节点可能定位到任何一个位置,因此不需要全局信息来描述节点离开形成的空位.

我们设计了一种随机叶节点替代(random leaf substitute)算法,其过程如下:

(1) 当离开节点是叶节点时,处理过程较简单,它通知父节点并转移资源索引到父节点,父节点更新路由表;

(2) 当离开节点是非叶节点时,它随机选择一个子节点并发出替代问询通告,若被问询子节点是叶节点,则转到步骤(3);若被问询子节点是非叶节点,则随机选择一个子节点并转发替代问询通告,如此向下层转发,直至一个叶节点收到替代问询通告,转到步骤(3);

(3) 叶节点成为替代者,返回替代应答消息.首先,通知父节点自己离开并转移资源索引到父节点;其次,接收离开节点的资源索引及路由信息,构建新的路由表,设定节点 NK;最后,向路由表中的上下游节点发送路由变更通告.上下游节点据此更新路由表,离开过程终止.

节点离开的例子:当离开节点是叶节点时,它通知父节点并转移资源索引到父节点,如图 5 所示中节点 AAC 所示.当离开节点是非叶节点时,如图 5 中节点 AA,它需要找一个叶节点来替代,AA 在以其为根的子树中任取一条从根到叶的路径,如图 5 中是以最左边的子节点作为路径选择.节点 AAAA 在被选中后,通知父节点并转移资源索引到父节点,然后接收 AA 信息的转移,包括节点 NK、资源索引及其父子节点信息.最后,AA 还要通告上下游节点关于自己的变更.在这个过程中,信息转移只涉及 3 个节点,转移发生 2 次,不产生联动反应.

可以看到,在离开过程中,替代叶节点的选择是任意的,它不需要具有在位置或身份上的任何特殊性.算法的显著优点是,离开过程不需要任何全局信息,而且即使网络规模缩小(如因用户活动的时间性限制),没有足够的新节点进入,系统也将按照自然规则收缩而不发生畸变.算法可能会破坏树的平衡结构,但这种破坏引起的失衡非常小而且暂时,下文将证明这种失衡会迅速被新到节点消除.

性质 **7**. 节点离开可能引起失衡,这种失衡导致的层深差异>2 的概率很小.

证明:不失一般性,假设节点离开导致的层深差异要达到 3,则必须满足一个条件:即节点离开前,树的最大层深差异至少已达到 2,根据式(5),MPPBTree 形成这种结构的概率很小,故假设成立的概率也很小. □

性质 **8**. 由节点离开后形成的空位叶节点$(NKL_i)_j$ 分配到新节点的权利一定优先于兄弟节点的任意空位子节点$(NKL_{i+1})_k$.其中,$(NKL_i)_j$ 代表第 L_i 层的第 j 个 NK 节点位置.

证明:设 L_i 层节点数为 n,不失一般性,设节点离开后形成空位叶节点$(NKL_i)_1$,空位兄弟节点 $m-1$ 个,分别是$(NKL_i)_2$,$(NKL_i)_3$,…,$(NKL_i)_m$.又设父节点 $P(NKL_{i-1})_1$ 为 p,则根据逐次随机探测算法可得

$$P(NKL_i)_1=P(NKL_i)_2=\ldots=P(NKL_i)_m=p/m,$$

$$P(NKL_i)_{m+1}=P(NKL_i)_{m+2}=\ldots=P(NKL_i)_{n-m}=0.$$

显然,非空位兄弟节点不能被分配到新节点,故其子节点得到新节点的概率为 0;而其他$(NKL_{i+1})_k$ 节点因其父节点为空也不能得到新节点.因此,$P(NKL_{i+1})_k$ 恒等于 0.例如在图 6 中,原 AA 节点离开后由 AAA 节点代替,这样,AAA 位置成为空位 NK.此时,节点 AA 的 B 子树层深比 A 子树大 2,发生了树的失衡.但是,AAA 空位一定会先于 AAB 的任何子节点空位(如 AABA)得到新节点补充,从而消除了树的暂时失衡,维护树变化的平衡性,这是由逐次随机探测算法的半强制性来保证的. □

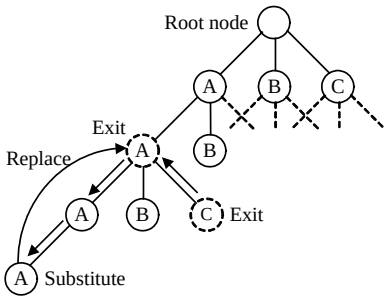


Fig.5 Substitute process in node departure

图 5 节点离开时的替代行为

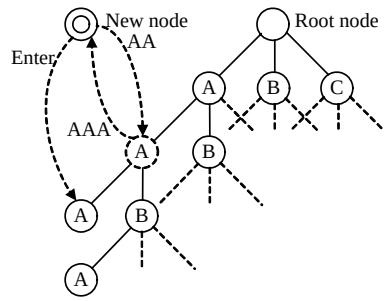


Fig.6 New node prior to enter empty position in higher layer

图 6 新节点优先补充层高的空位 NK

每个节点始终通过“心跳”消息与资源索引节点联系,启用超时机制检测节点的存在,一旦认定节点不存在,就将对应索引删除.因此,节点在退出前不需要对资源索引源进行批量通告.但这可能会导致索引存储节点没有及时探测到源节点的离开而产生一些失效索引,从用户角度来看,极少数的失效索引是可以容忍的.

3.4 异常处理和负载均衡

一个节点的非正常退出,将引起索引丢失和子树离线.若子节点发现其父节点不可达,则执行两个步骤:首

先,它任意选择子树中的一个叶节点作为异常节点替代者,替代者设异常节点 NK 为初始 NK,并启动加入程序.根据随机探测算法,它将代替异常节点,若异常节点已有替代者,则停止加入程序;随后,子节点保持原有 NK 为初始 NK 并重新加入网络,它们将各自进入原有位置,不丢失原有索引.只有替代者需在系统内广播重新建立索引.

实际应用中,关键字出现概率的不同,易引发存储和路由负担的不平衡.上文的分配方式虽然没有详细考虑这个问题,但是留有较大的解决空间.我们考虑的一个方案是:采用集束 NK 和分离 NK 法,对于出现概率较少的关键字,用一个节点负责多个 NK;反之,用多个节点负责一个 NK,使得不同 NK 得到不同的逻辑链路数,减小节点之间的负载差异,这也是我们下一步的研究方向.

4 路由长度分析

路由长度主要包括查询的路由长度、节点加入时用于定位的路由长度和节点离开时查询替代者的路由长度.我们从这 3 个方面综合考察系统的路由性能,着重量化在最坏情况下的一些路由指标,以真实反映 MPPBTree 路由的实质特征,而相关的平均数值将在仿真实验中给出.一些相关定义如下:

L:MPPBTree 的层深,若节点数为 N,则多数情况下 $L=[\log_{26}N]$;

HN:完成一次查询或定位所花费的总的路由跳数;

$HN_{\max}$:完成一次查询或定位所花费的最大路由跳数;

δ:新节点的初始 NK;

β:新节点加入后的实际 NK;

γ:加入过程中中介节点的 NK.

查询的路由长度:查询路由长度同时与 QK 长度和 L 相关,一般来说,考虑两个叶节点通过 root 通信,最长路由长度是 $HN_{\max}=2L$.对于一个具体 QK,其路由长度还要取决于 $length(prefix(QK,NK))$的值,即

$$HN=\begin{cases} length(NK)+L-2i, & length(QK)>L \\ length(NK)+length(QK)-2i, & length(QK)<L \end{cases} \tag{8}$$

其中,i 是前缀匹配的长度,即

$$i=length(prefix(QK,NK)).$$

很多 P2P 网络应用的规模在 50 万节点以下,此时,系统也只有 5 层(Layer 0~Layer 4),L 为 4,整个系统的最长路由长度仅是 8;当节点规模达到 1 000 万时,最长路由长度也仅是 10.因此,系统的可扩展性较强.

节点加入时,用于定位的路由长度:节点加入时要运行逐次随机探测算法,最坏情况下是通过叶节点中介定位到经过 root 的另一端的叶节点,因此,

$$HN_{\max}=2L \tag{9}$$

对于具体的一个加入过程,

$$HN=length(\gamma)+length(\beta)-2i \tag{10}$$

其中,i 是前缀匹配的长度,即

$$i=length(prefix(\delta,\gamma)) \tag{11}$$

大多数情况下,用于定位的路由长度接近 $2L$,因为叶节点占多数,加入过程在它们之间发生的概率较大.MPPBTree 的 L 值增长缓慢,因此,路由性能基本不随规模网络的增大而退化.

节点离开时,查询替代者的路由长度:非叶节点要寻找叶节点作为自己的替代,最坏情况下是 $HN_{\max}=L$,不需要任何全局信息.对一个处于 L_i 层的节点,则有

$$HN=L-L_i-1 \text{ 或 } HN=L-L_i \tag{12}$$

HN 的具体取值与 L 层节点数有关,但影响整体路由性能的关键是离开节点的性质.从概率的角度来考察,对于一个 50 万节点的网络,每分钟近 10 000 个节点离开,网络中叶节点数量占 96%,即离开节点是非叶节点的数量约 400 左右,当中绝大多数与叶节点也仅 1 跳之遥,从而使得平均路由长度很短,这在仿真实验中得到验证.即使是较坏情况下,由于 L 值增长缓慢,$O(L-L_i)$的路由长度也完全可以接受.综上所述,MPPBTree 能够应付频繁

的节点离开,而加入的“严格”过程也有利于系统规模平稳的增长.

5 性能评价

我们通过仿真程序对系统性能进行评估,网络规模在 10000~100000 个节点之间发生变化,每次查询性能测试随机产生 1 000 次查询,取平均值作为实验数据;每次维护性能测试随机产生 500 个节点的动态行为,取平均值作为实验数据.在一些性能指标上,我们使用 chord[3]和 BATON[16]作为比较,其中,chord 是一种基于 DHT 的可扩展结构化 P2P 网络;BATON 是一种基于排序平衡树的支持范围查询的结构化 P2P 网络.两者都具有查询效率高、可扩展性强和完全分布运行的优点.

精确匹配查询:图 7 显示的是 QK 精确匹配查询在 MPPBTree 所需的平均跳数和最大跳数,以及在 chord 所需的平均跳数.可以看到:MPPBTree 比 chord 要少 1 跳左右,而最大跳数在 100 000 节点规模的网络也只达到 8,此时,chord 的平均跳数已经超过 8 了.MPPBTree 的最大跳数就是层深 L 的 2 倍.由于树的组织是概率平衡,L 是未知的,考虑 100 000 节点的网络,第 4 层每个节点平均只能分配 4.65 个叶节点,因此从概率上看,一个节点被分配满额子节点的概率极小,这意味着 100 000 节点网络通常都不超过 5 层,最大跳数也就不会超过 8.BATON 在查询上的效率虽未在图中给出,但通常要低于 chord[3].对于模糊查询我们未作比较,因为三者对它的定义并不一致:chord 本身不支持模糊查询,一些 DHT 改进方案[9]多是从矢量相似度角度去定义;BATON 更侧重于范围查询(range query);而 MPPBTree 则是最简单常用的包含匹配查询.

节点动态维护:图 8 显示的是 MPPBTree 节点加入的平均跳数及最大跳数、节点离开的平均跳数及 chord 节点加入的平均跳数.这里的节点加入指的是新节点从联系中介节点开始到最后定位至对应位置的过程,不包含路由更新、索引转移等步骤;节点离开指的是离开节点寻找到替代节点的过程,也不包含路由更新、索引转移等步骤.可以看到:MPPBTree 节点加入过程较快捷,最大跳数为 $2L$,加入过程对一个上线节点通常只运行一次,而且跳数并不多,如 100 000 节点网络的跳数上限一般不超过 8.系统节点离开的平均跳数仅为 1 跳多,远少于 BATON(参见文献[16]),定位是很迅速的.

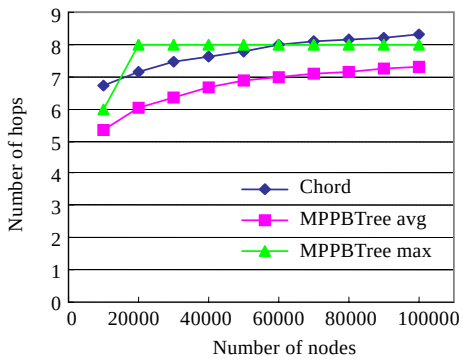


Fig.7 Exact match

图 7 精确匹配查询

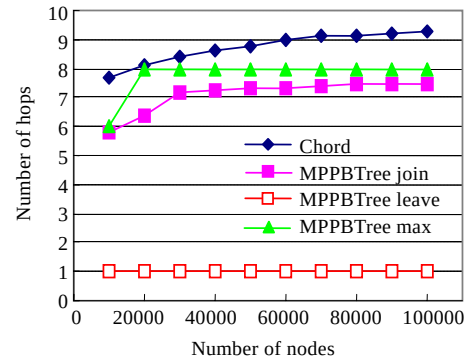


Fig.8 Dynamic maintenance

图 8 节点动态维护

平均路由表大小:影响路由性能的因素除了路由跳数以外,还有节点路由表的大小.图 9 显示了 3 种系统在不同规模网络的平均节点路由表尺寸.BATON 和 chord 的平均节点路由表大小相当,其中 BATON 虽然是一种树型结构,但除了维护上下游节点信息以外,还要维护部分同层节点信息以加快查询过程.BATON 采用类似 chord 的方式组织同层节点的相互关系,这些全局信息的增加使得其路由表尺寸与 chord 相当.MPPBTree 同样是一种树型结构,但每个节点不需要维护任何同层节点信息,只需维护上下游节点信息.考虑叶节点占节点总数的大部分,MPPBTree 的平均节点路由表尺寸只有 2.图 10 显示了平均节点路由表尺寸的分布,考察非满额节点的平均子节点数及这些非满额节点占节点总数的比例.可以看到,不同的网络规模下二者成反比.这是因为通常只有倒数第 2 层节点会成为非满额节点,而其平均子节点数越多,节点总数也越多,非满额节点占的比例就越小.

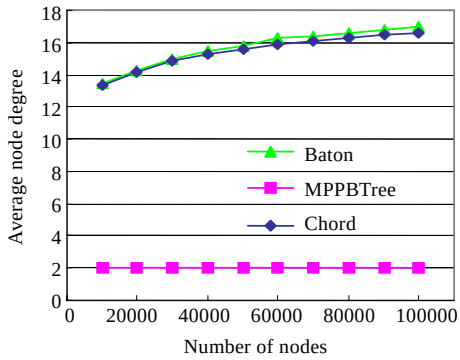


Fig.9 Average routing table size

图 9 平均路由表大小

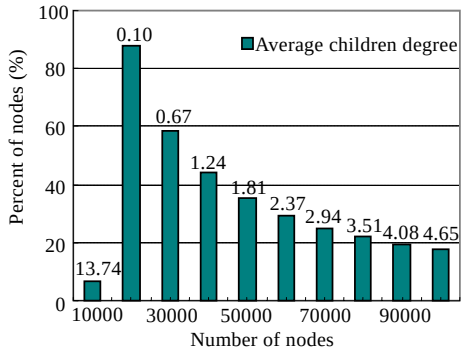


Fig.10 Average children degree and proportion of non-full nodes

图 10 非满额节点的平均子节点度及比例

平均路由表更新开销:这是衡量系统动态维护性能的一个重要指标.图 11 显示了 MPPBTree 和 chord 在节点状态变更后引起的路由更新通告消息的数量.MPPBTree 由于除上下游节点信息外不需要维护其他任何全局信息,因此对非叶节点而言,路由更新只涉及其上下游节点,对叶节点则仅需要更新其父节点的路由表,节点的加入或离开引发的平均路由变更通告消息只有 2 条左右;相比之下,chord 则需要 $O(\log^2 N)$条消息.MPPBTree 的最大路由变更通告消息数是 28,分别是节点离开时替代节点对其父节点的通告和对被代替节点的上下游节点的通告,这样的节点通常只处于第 2 层~倒数第 3 层之间,在任意规模的网络中都只占节点总数的很小比例.

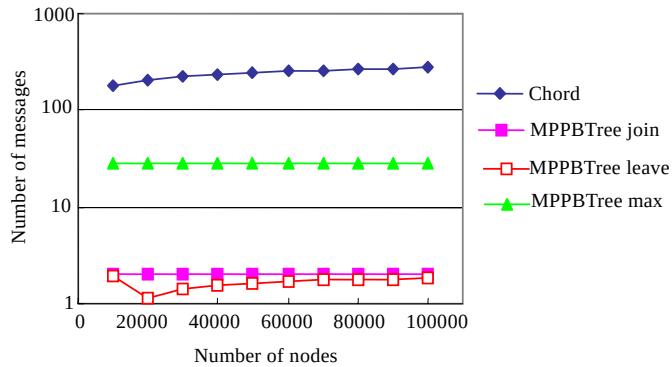


Fig.11 Average routing table update cost

图 11 平均路由表更新开销

6 总结和进一步的工作

本文提出了一种基于匹配路径和概率平衡树的 P2P 语义路由系统,RK 通过匹配路径与资源位置相关并按照一定规则排布,QK 采用与 RK 一致的语义表达,从而实现语义路由过程;系统能同时支持精确、模糊、多关键字及通配符等查询模式;系统通过概率平衡的方法实现节点加入,避免了平衡树的高额维护成本;系统同时从数据存储和路由两方面去考虑负载特性,更好地分配节点资源.与其他 P2P 系统相比,MPPBTree 在支持查询灵活性的同时保持了良好的可扩展性和路由效率,维护成本相对较低.未来的研究工作包括:节点能力自适应的存储组织方法、考虑物理距离的拓扑组织方法及更均衡的负载结构和更简单的维护手段等.

致谢 在此,我们向对本文的工作给予支持和中肯而深刻建议的审稿人及编辑部老师表示感谢.

References:

[1] Gnutella. 2003. http://www.gnutella.com/

[2] Napster. http://www.napster.com/

[3] Stoica I, Morris R, Karger D, Kaashoek MF, Balakrishnan H. Chord: A scalable peer-to-peer lookup service for Internet applications. In: Govindan, ed. Proc. of the ACM SIGCOMM 2001. San Diego: ACM Press, 2001. 149−160.

[4] Ratnasanry S, Francis P, Handley M, Karp R, Shenker S. A scalable content-addressable network. In: Govindan, ed. Proc. of the ACM SIGCOMM 2001. San Diego: ACM Press, 2001. 168−175.

[5] Rowstron A, Druschel P. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In: Guerraoui R, ed. Proc. of the Int'l Conf. on Distributed Systems Platforms (Middleware 2001). Heidelberg: Springer-Verlag, 2001. 135−141.

[6] Zhao B, Kubiatowicz J, Joseph A. Tapestry: An infrastructure for fault-tolerant wide-area location and routing. Technical Report, UCB/CSD-01-1141, Berkeley: Computer Science Division, 2001. 106−115.

[7] Ishikawa N, Sumino H, Omata E, Hjelm J, Yu Y, Zhu ZW. Semantic content search in P2P networks based on RDF schema. In: Proc. of the PACRIM 2003. Victoria: IEEE Computer Society, 2003. 143−148. http://www.ece.uvic.ca/~fayez/pacrim/index.html

[8] Crespo A, Molina HG. Semantic overlay networks for P2P systems. 2003. http://www.db.stanford.edu/~crespo/publication/op2p.pdf

[9] Tang C, Xu Z, Mahalingam M. Peer-to-Peer information retrieval using self-organizing semantic overlay networks. In: Anja F, Martina Z, Jon C, David W, eds. Proc. of the ACM SIGCOMM 2003. Karlsruhe: ACM Press, 2003. 175−186.

[10] Shen HT, Shu YF, Yu B. Efficient semantic-based content search in P2P network. IEEE Trans. on Knowledge and Data Engineering, 2004,16(7):813−825.

[11] Lü Q, Cao P, Cohen E, Li K, Shenker S. Search and replication in unstructured peer-to-peer networks. In: Gupta M, ed. Proc. of the 16th ACM Int'l Conf. on Supercomputing (ICS 2002). New York: ACM Press, 2002. 254−261.

[12] Joseph S. NeuroGrid: Semantically routing queries in peer-to-peer networks. In: Enrico G, Ludmila C, Gianpaolo C, Fabio P, Gian PP, eds. Proc. of the Int'l Workshop on Web Engineering and Peer-to-Peer Computing. Pisa: IEEE Computer Society, 2002. 202−214.

[13] Daniel AM. Scalable P2P search. IEEE Internet Computing, 2003,7(2):83−87.

[14] Sripanidkulchai K, Maggs B, Zhang H. Efficient content location using interest-based locality in peer-to-peer systems. In: Anja F, Martina Z, Jon C, David W, eds. Proc. of the ACM SIGCOMM 2003. ACM Press, 2003. 175−186.

[15] Chawathe Y, Ratnasamy S, Breslau L, Lanham N, Shenker S. Making Gnutella-like P2P systems scalable. In: Anja F, Martina Z, Jon C, David W, eds. Proc. of the ACM SIGCOMM 2003. ACM Press, 2003. 407−418.

[16] Jagadish HV, Ooi BC, Vu QH. BATON: A balanced tree structure for peer-to-peer networks. In: Böhm K, Jensen CS, Haas LM, Kersten ML, Larson P, Ooi BC, eds. Proc. of the 31st VLDB Conf. Trondheim: VLDB Endowment, 2005. 661−672.

许立波(1976－),男,浙江宁波人,博士生,主要研究领域为 P2P 网络,网络流量模型,网络性能评价.

于坤(1972－),男,博士生,主要研究领域为 P2P 网络协议及性能分析,ad-hoc 网络.

吴国新(1956－),男,教授,博士生导师,主要研究领域为分布式计算,网络安全,信息化关键支撑技术,网络性能评价.