采用 k-均值聚类算法的资源搜索模型研究

邵必林[1]，边根庆[2]，张维琪[2]，闫瑾[2]
(1. 西安建筑科技大学管理学院，710055，西安；2. 西安建筑科技大学信息与控制工程学院，710055，西安)

摘要：针对当前海量信息存储对等网络系统中资源搜索技术效率较低的问题，提出了一种采用 k-均值聚类分析的高效搜索模型. 该模型利用资源描述框架(RDF)描述的元数据进行聚类分析，使得资源的搜索由全局变为局部，从而有效地提高了资源搜索效率；采用动态优化排序技术显著提高了查询的速度. 通过子网分裂算法和节点备用算法增强了模型的可扩展性、安全性和可靠性. 仿真结果表明，所提模型在查找时延和平均路径方面均比传统搜索模型更加高效、便捷.

关键词：海量信息存储；聚类分析；元数据；搜索技术
中图分类号：TP311.52 **文献标志码**：A **文章编号**：0253-987X(2012)10-0055-05

A Resource Search Model Using k-Means Clustering Analysis

SHAO Bilin[1], BIAN Genqing[2], ZHANG Weiqi[2], YAN Jin[2]
(1. School of Management, Xi'an University of Architecture and Technology, Xi'an 710055, China;
2. School of Information and Control Engineering, Xi'an University of Architecture and Technology, Xi'an 710055, China)

Abstract: An efficient search model using k-means clustering analysis is proposed to improve the low efficiency of resource retrieval technology in peer-to-peer net with mass information. The metadata described by RDF framework is used to perform cluster analysis of resources and the search range of resources is narrowed from global to local so that the model can enhance the efficiency of resource retrieval effectively, a dynamic optimization technique is adopted to significantly improve the inquiry speed. Moreover, the use of the subnet division algorithm and the node backup algorithm enhances the scalability, safety and reliability of the model. Simulation results and comparisons with traditional retrieval models show that the proposed model is convenient and has higher resources searching efficiency in search delay and average path.

Keywords: mass information storage; clustering analysis; meta data; retrieval technology

随着大规模因特网的应用，对等网络作为一种新的通信模式蓬勃发展起来了. 对等网络中的计算机具有相同的功能，一台计算机既是服务器也是客户机，无主从之分. 根据结构的不同，对等网络可以分为结构化和非结构化2种. 非结构化网络[1]又分为完全分散的 Gnutella、FreeNet，部分分散的 Kazza 以及混合分散的 Napster；结构化网络主要有 Chord、CAN、Pastry 等. 结构化系统采用分布式哈希表(DHT)技术[2]来构造，不同的 DHT 算法决定了存储网络具有不同的逻辑拓扑. 目前，主流的基于 DHT 的 Chord 协议是为对等网络应用而设计的环状拓扑结构，在 Chord 基础上的关于对等网络中海量信息的高效定位问题已经进行了大量的研究.

Chord 是由美国的加利福尼亚大学伯克利分校和麻省理工学院共同提出的一种分布式可扩展的查找算法，在 Chord 网络中，每个节点要维护一张 m 个表项的路由表，表中有近 1/3 的项是重复的，相应的查找效率比较低. 为了解决这个问题，很多文献对 Chord 做了进一步研究，结果是每个节点仍然需要维护大量的路由信息，据此本文提出了一种基于 k-

收稿日期：2012-04-01. 作者简介：邵必林(1965—)，男，教授. 基金项目：国家自然科学基金资助项目(61073196)；陕西省自然科学基础研究基金资助项目(2011JM8026).
网络出版时间：2012-08-06 网络出版地址：http://www.cnki.net/kcms/detail/61.1069.T.20120806.1808.001.html

均值算法的资源搜索模型，即先通过聚类算法将资源划分到不同的逻辑子网中，使得资源的搜索由全局变为局部，然后在各个子网中进一步定位目标资源节点，以减少平均查找跳数，提高查找效率.

1 相关研究

1.1 经典 Chord

在 Chord 环中，所有节点按照节点标识符从小到大沿着顺时针方向排列成一个逻辑环，环中每个节点维护一张最多有 m 个表项的路由表. 在对等网络中每个资源 R 赋予了一个关键字 V，利用 Chord 就可以在该关键字映射的节点上存储或读取相应的 (R,V)对.

Chord 路由搜索过程：①对要查询的资源 R 进行哈希运算 $h(R)$；②如果一个节点的 $h(R)$落在该节点标识及其后继节点标识之间，那么该节点的后继节点就拥有关于 R 的资源，后继节点的 R 将返回给资源请求节点，搜索停止，反之该节点查询自己的路由表，找出路由表中拥有节点标识符不超过 $h(R)$ 的最大节点标识符的第一个节点并将请求转发给它. 重复上述过程，最终可定位资源，完成搜索.

1.2 改进的 Chord

1.2.1 高阶 Chord 高阶 Chord[3] 中节点维护的路由表项是将 Chord 环的地址空间不间断地进行 k 划分，相应的指针分布比经典 Chord 更为密集，这样请求可以更快地转发到目标节点，但不足之处是：路由表的长度随 k 的增大线性增加，这不仅会引起更大的内存消耗，而且会使得节点消耗更多的网络带宽来维持路由信息的准确性.

1.2.2 改进路由表的 Chord 改进路由表的 Chord[4] 是每个节点在原来 Chord 路由表的基础上再增加 2 个表：最近访问节点表和邻居节点表. 由于每个节点要维护 3 张表，所以空间效率降低，当节点越来越多时，查询效率必然受到严重的影响.

1.2.3 双向 Chord 最初的 Chord 中资源查找路径都是顺时针进行的，不管目标节点距离源节点有多远，查找都是从源节点开始. 这就带来了一个问题：当目标节点在源节点的逆时针一侧时，由于查找方向是顺时针进行的，所以查找代价相应增加. 针对此种情况，文献[5]提出了一种基于节点异构的双向查询 Chord 系统. 该系统修改了路由信息，即在路由表无法到达的另一半的 Chord 环上选取若干个节点信息加入到路由表中，使得查找跳跃进行，从而减少了评价路由的跳数，提高了查询效率. 但是，该系统有 2 点不足：①节点仍然需要维护较多的路由信息；②修改路由信息时选取的若干个节点的信息是随机的，不能保证大规模对等网络的查找效率.

通过上述分析可知，改进的 Chord 查询性能比经典 Chord 有所改进，但仍然存在问题. 据此，本文在综合考虑了时间和空间性能的基础上，提出一种基于 k-均值聚类算法的聚类分析模型(CAM). 该模型通过定义排序优化调整函数，结合三元组和动态链表二分搜索结构，使得搜索时间缩短，资源搜索定位效率提高.

2 资源分类

2.1 资源描述框架

资源描述框架(RDF)[6] 是表达关于 Web 资源的元数据. 一个 RDF 文档包含了多个资源描述，一个资源描述由多个语句组成，一条 RDF 语句是由资源、属性类型、属性值构成的三元组，对应于自然语言中的主题、谓语、宾语. 网络节点中存在着多个文档，也就是说网络节点中包含了多个“主题”. 本文将由主题、属性类型、属性值构成的一条 RDF 语句记为 RDF(s,p,v)，对 s、p、v 这 3 个字段分别量化后再进行资源 k-均值聚类分析.

2.2 k-均值算法

k-均值算法是划分聚类算法[7-9] 的一个典型的算法. 假设有 N 个数据点的集合 $I=\{\boldsymbol{x}_1,\boldsymbol{x}_2,\cdots,\boldsymbol{x}_N\}$，每个 $\boldsymbol{x}_i$ 代表一个特征向量，则根据某个评分函数不断地使用类簇的均值法，可将 N 个数据点划分到 k 个分类中. 划分过程的关键是选取合适的评分函数，使得每一个数据点到它所属的聚类中心的距离最小. 本文定义聚类间距为欧氏距离，并用协方差来定义评分函数. k-均值算法的伪代码如图 1 所示.

```
for k=1,…,K 令r(k)为从D中随机选取的一个点;
while 聚类C_k中发生变化 do
形成聚类:
for k=1,…,K do C_k={x∈D | d(r(k),x)=d(r(j),x)
    对所有 j=1,…,K,j!=k};
end;
计算新聚类中心:
for k=1,…,K do R_k=C_k内点的均值向量
end;
end;
```

图 1 k-均值算法

3 搜索模型构建

3.1 CAM 体系结构

CAM 由中心的一级服务器和各个子网组成，

如图 2 所示.

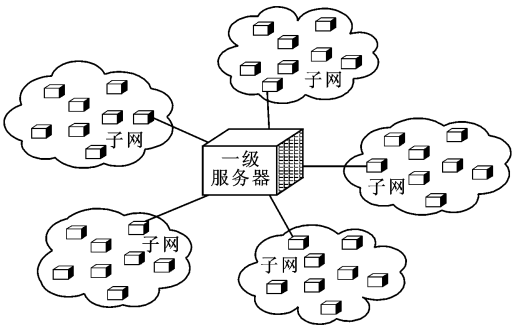

图 2　CAM 体系结构

通过聚类算法将具有相同主题的节点划分到同一个子网中. 一级服务器用一个三元组来记录各个子网的主题 s，各子网分别拥有一台二级服务器，服务器利用动态链表存储 RDF 元组中的 p 和 v 字段. 为了进一步提高搜索效率，在每台服务器上开辟了一个缓冲区，用来记录资源查询的路由. 服务器在查询子网节点时，首先检查缓冲区，如果缓冲区中存在相应的查询记录，将查询记录直接返回目的子网，否则搜索三元组或动态链表. 缓冲区由队列机制来实现，并采取最近最久未使用(LRU)算法来更新缓冲区的内容.

由于各节点信息和元数据均集中在一级、二级服务器上，其中一台服务器出现故障，整个网络可能瘫痪，因此本文建立了备用服务器机制，即当某台服务器出现故障时启动备用服务器进行数据迁移，以提高系统的安全性和可靠性.

3.2　数据结构

一级服务器设置了一个三元组，其字段值分别为 IP、s_i 和子网 S_j，其中 i、j 的取值是随机的，且满足 $1\leqslant i,j\leqslant N$. 三元组按 s_i 值有序排列，查找时相当于二分查找，搜索时依据 s_i 来匹配子网.

搜索过程中需满足的属性不唯一，为此采用图 3 所示动态双向链表来存储 p_i、v_j 字段，其中 i、j 的取值是随机的，且满足 $1\leqslant i,j\leqslant N$.

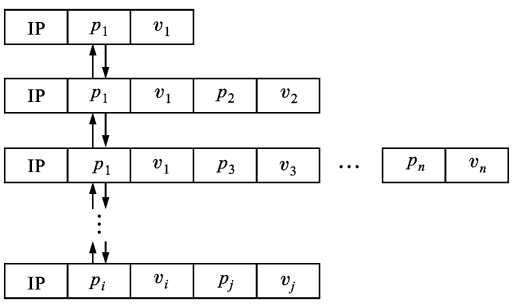

图 3　动态链表

设置头、尾 2 个指针，使得搜索从两边沿着指针方向移动，对 2 个字段逐一进行匹配. 为了进一步提高搜索效率，根据参考文献[10-11]，本文设计了一个动态排序函数

$$F=\alpha_1 f_1(\boldsymbol{x})\oplus\alpha_2 f_2(\boldsymbol{y})\oplus\alpha_3 f_3(\boldsymbol{z})$$

式中：$\boldsymbol{x}$ 表示查询的关键词与文档的相似度；$\boldsymbol{y}$ 表示文档点击率；$\boldsymbol{z}$ 表示二元组个数；α_1、α_2、α_3 为常数，且满足 $\alpha_1+\alpha_2+\alpha_3=1$.

用一组按相似度由大到小排序的向量 $\boldsymbol{x}$ 来表示 $f_1(\boldsymbol{x})=(\boldsymbol{x}_1,\boldsymbol{x}_2,\cdots,\boldsymbol{x}_n)$，则文档间相似度

$$d_{\mathrm{Sim}}(D_1,D_2)=\cos\theta=$$

$$\sum_{k=1}^{n}W_{1k}W_{2k}/\left(\sum_{k=1}^{n}W_{1k}^2\sum_{k=1}^{n}W_{2k}^2\right)^{1/2}$$

式中：W_{1k}、W_{2k} 分别表示文档 D_1 和 D_2 第 k 个特征项的权值，$1\leqslant k\leqslant N$.

文档点击率被记载在日志文件中，用一组按文档点击率由大到小排序的向量 $\boldsymbol{y}$ 来表示 $f_2(\boldsymbol{y})=(\boldsymbol{y}_1,\boldsymbol{y}_2,\cdots,\boldsymbol{y}_n)$，$\boldsymbol{y}_i=\max\{\boldsymbol{m}_1,\boldsymbol{m}_2,\cdots,\boldsymbol{m}_n\}$，$i=1,2,3,\cdots$ 且 $\boldsymbol{m}_i\neq\boldsymbol{y}_1,\cdots,\boldsymbol{y}_{i-1}$.

同理，按二元组个数由大到小排序的向量 $\boldsymbol{z}$ 来表示 $f_3(\boldsymbol{z})=(\boldsymbol{z}_1,\boldsymbol{z}_2,\cdots,\boldsymbol{z}_n)$，$\boldsymbol{z}_i=\max\{\boldsymbol{t}_1,\boldsymbol{t}_2,\cdots,\boldsymbol{t}_n\}$，$i=1,2,3,\cdots$ 且 $\boldsymbol{t}_i\neq\boldsymbol{z}_1,\cdots,\boldsymbol{z}_{i-1}$.

服务器按照上述参数将链表中的内容重新排序，然后通过调用插入函数来调整节点.

3.3　资源搜索

假设没有节点申请加入或退出网络，第 n 个节点提交了一个查询请求，即书名：数据结构，作者：严蔚敏，类别：计算机，经 RDF 分析提取出查询请求的“主题”是“数据结构”，同时满足的属性是“严蔚敏”、“计算机”. 该查询过程如下.

(1)节点通知一级服务器要查找的主题是“数据结构”，一级服务器查看自己的缓冲区，如果找到与“数据结构”匹配的索引，直接返回源节点所在子网的地址，否则查找自己的三元组，查找拥有主题的子网 S_j，同时按照 LRU 算法更新自己的缓冲区.

(2)找到 S_j 之后，节点通知 S_j 中的二级服务器，并要求查找满足属性(严蔚敏、计算机)的字段. 二级服务器首先查看自己的缓冲区，如果缓冲区中存在匹配项，直接返回目标节点的 IP 地址，否则指针沿着链表移动，结合动态排序函数 F 高效地进行查找，直至查到目标节点；然后按照 LRU 算法适时地更新缓冲区；最后发起查询请求的源节点与目标节点建立连接，直接进行资源的交互. 搜索算法如图 4 所示.

```
Search 函数
{
 源节点向一级服务器发送搜索请求;
 一级服务器响应请求，扫描自己的缓冲区，若缓冲区中存在匹配项,该函数返回1,否则返回0;
  if  缓冲区中没有匹配项
       {
          更新缓冲区并扫描其一级服务器三元组 ;
          定位目标子网并返回目标子网的IP;
       }
  一级服务器通知二级服务器查找拥有给定属性的匹配项;
    if   缓冲区中没有匹配项
       {  二级服务器更新缓冲区同时扫描链表;
            if   链表为空
            提示重新输入关键词;
        }
 返回目标节点的IP;
 源节点和目标节点建立连接并进行资源交互;
}
```

图4 搜索算法

3.4 节点加入和退出网络

3.4.1 节点加入 考虑到子网规模和服务器维护等问题,设置了一个子网规模值阈值 L. 当节点申请加入子网时,先检查现有子网规模 S,如果 $S<L$,则同意加入. 经过聚类分析之后,如果 s_i 存在于一级服务器之中,直接将节点加入拥有 s_i 的子网,然后在子网的二级服务器中记录该节点的 p_i、v_j;否则,将新节点的 s_i 插入到一级服务器的三元组的第一个位置上,同样地在二级服务器中记录新节点的 p_i、v_j. 如果 $S\geqslant L$,则触发子网分裂机制. 子网分裂机制是节点在满足加入子网要求且 $S\geqslant L$ 时,按子网中各节点的被搜索频度运行子网分裂机制,即频度高的节点分到一个二级子网中,剩下的分到另外一个二级子网中,以此类推. 新加入的节点优先插入到被搜索频度高的二级子网中,如果 s_i 存在于该二级子网,只需记录新节点的 p_i、v_j;否则,在一级服务器中记录新节点的 s_i,在二级服务器中记录新节点的 p_i、v_j. 新节点请求加入子网的算法如图5所示.

```
Addnode  函数
{
  新节点请求加入服务器，并告知自己的s_i,p_i,v_i;
     if  (S>L)
          运行子网分裂机制;
     采用k-均值算法将新节点加入网络;
}
```

图5 新节点加入算法

由于频繁地加入节点会造成网络颠簸,同时费力耗时,因此本文开辟了一块缓冲区,即先将申请加入的节点放入缓冲区中,间隔一段时间后再进行动态加入. 搜索资源时先搜索服务器,若服务器中没有相应的信息,再进行缓冲区遍历. 具有目标资源的节点应优先加入子网.

3.4.2 节点退出 节点退出网络时,一级服务器和二级服务器应删除相应的信息,三元组移动元数据,动态链表修改指针,信息删除时采用排序调整函数.

4 模型分析

采用CAM模型,服务器只需要维护少量的信息,节点加入或退出网络时,服务器仅需更新自己的路由表,这样空间维护代价大大降低. 实际上,子网中的所有节点均退出(相当于删除一个 s_i)的概率非常小,因此一级服务器的信息维护代价较小,但安全性和可运行性需要保证.

采用节点分裂机制可将网络划分为二级子网、三级子网等,以保证系统的可扩展性.

临时缓冲区可存放申请加入子网的节点,并根据需要进行动态加入,这样避免了节点频繁加入造成的网络颠簸现象,同时节省了处理的时间.

在网络初始化时执行聚类算法需要一定的时间,但在节点加入后聚类过程会很快完成,且不会影响资源的搜索过程. 采用动态排序函数可使得数据查找始终处于高效状态.

一台服务器一旦出现问题,即可触发备用服务器节点机制来替代故障机器,以保证系统的安全性和可靠性.

5 仿真实验

5.1 实验环境

仿真实验使用了 BRITE 拓扑生成软件,并采用 bottom-up 参数建立两层拓扑结构. 仿真环境为: Radhat Linux, CPU; Intel Core(TM)2 Duo, 2.00 GHz; 内存 2.00 GB.

5.2 实验结果及分析

实验为节点数从0增加到9 000时的9 000次查询. 图6是双向 Chord 和本文 CAM 的查找时延比较. 图6显示,当节点数较少时,CAM 的时延比双向 Chord 的高,但随着节点数的增加,CAM 的时延较双向 Chord 略有降低. 这是因为: 节点数较少时,双向 Chord 中节点存储的路由信息冗余度比较低,丰富的路由信息使得时延较低; 当节点数较多时,双向 Chord 的路由信息冗余度较高,而 CAM 的冗余度很低的路由信息都存储在服务器上,使得路由信息无效转发的可能性降低,加之 CAM 中采用

动态排序函数显著提高了查询速度，因而 CAM 的时延比双向 Chord 的有所下降.

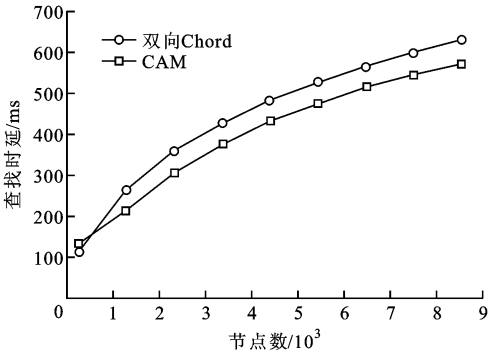

图 6　查找时延比较

图 7 是双向 Chord 和 CAM 的平均路径比较. 图 7 显示，CAM 的平均路径长度较双向 Chord 的短. 原因在于：双向 Chord 中的每个节点要维护一个顺时针和一个逆时针的路由表，这样有效信息增多，查询速度加快，平均查询路径为 $(1/3)\lg N$（N 为网络节点总数），所以路由信息冗余量较大；CAM 中节点加入需执行 k-均值算法，该算法将节点资源划分为几个逻辑子网，从而使得资源的搜索由全局变为局部，搜索路径相比需要进行全局搜索的双向 Chord 显著下降.

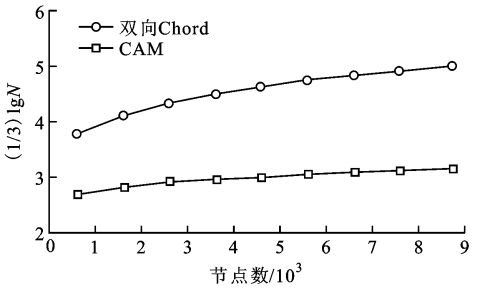

图 7　平均路径比较

6　结　论

在 k-均值聚类算法的基础上，构建了一种高效的资源搜索模型 CAM，即先利用 RDF 框架描述的元数据进行聚类分析，再采用简单、易行的数据结构和动态排序函数来提高查询效率. 仿真结果表明，所提模型可以降低查找时延，缩短平均路径长度，提高资源定位效率，同时具有较好的搜索性能.

参考文献：

[1] EDITH C, SCOTT S. Replication strategies in unstructured peer-to-peer networks[C]// Proceedings of the 2002 SIGCOMM Conference. New York, USA: ACM Press, 2002:177-190.

[2] 肖波，聂晓文，侯孟书. DHT 网络规模估计算法的定量分析与设计[J]. 电子科技大学学报，2011，40(2)：261-266.
XIAO Bo, NIE Xiaowen, HOU Mengshu. Quantitative analysis and design of an estimating algorithm on DHT network size [J]. Journal of University of Electronic Science and Technology, 2011, 40(2):261-266.

[3] 郝杰. 基于 DHT 的结构化 P2P 路由协议 Chord 的研究与改进[D]. 北京：北京邮电大学，2009.

[4] 王必晴. Chord 路由算法的研究与改进[J]. 计算机工程与应用，2010，46(14)：112-114.
WANG Biqing. Research and improvement of Chord routing algorithm [J]. Computer Engineering and Applications, 2010, 46(14):112-114.

[5] 周伟平，刘卫国. 基于节点异构的双向查询 Chord 系统[J]. 计算机工程，2009，35(2)：95-97.
ZHOU Weiping, LIU Weiguo. Bidirectional search Chord system based on heterogeneity of peers [J]. Computer Engineering, 2009, 35(2):95-97.

[6] W3C. Resource description framework [EB/OL]. (2003-10-20) [2012-01-15]. http://www.w3.org/RDF/.

[7] YU Zhiwen, WONG Hausan. Quantization-based clustering algorithm [J]. Pattern Recognition, 2010, 43(8):2698-2711.

[8] QIU Dingxi. A comparative study of the k-means algorithm and the normal mixture model for clustering: bivariate homoscedastic case [J]. Journal of Statistical Planning and Inference, 2010, 140(7):1701-1711.

[9] TSAI C W, YANG C S, CHIANG M C. A time-efficient pattern reduction algorithm for k-means based clustering [C]// IEEE International Conference on Systems, Man and Cybernetics. Piscataway, NJ, USA: IEEE, 2007:504-509.

[10] BLOEHDORN S, CIMIANO P, HOTHO A. Learning ontologies to improve text clustering and classification[C]// Proceedings of the 29th Annual Conference of the From Data and Information Analysis to Knowledge Engineering. Berlin, Germany: Springer, 2006: 334-341.

[11] 周博，刘奕群，张敏，等. 一种基于文件相似度的检索结果重排序方法[J]. 中文信息学报，2010，24(3)：19-36.
ZHOU Bo, LIU Yiqun, ZHANG Min, et al. A document relevance based search result re-ranking [J]. Journal of Chinese Information Processing, 2010, 24(3):19-36.

（编辑　苗凌　武红江）