基于结构化 P2P 的语义查询技术

侯祥松[①] 曹元大[②] 关志涛[③] 张 昱[①]
[①](北京理工大学计算机科学技术学院 北京 100081)
[②](北京理工大学软件学院 北京 100081)
[③](华北电力大学计算机系 北京 112206)

摘 要：由于 P2P 系统可以高效地对资源进行共享而受到关注，但现在的 P2P 仅支持精确查找或者通过洪泛方式进行低效率文本检索。为了解决这个问题，该文提出了一种结构化 P2P 环境中的文本检索系统，使用 LSH 函数将高维语义相关的文本向量映射相近的节点上，并解决了由此带来的负载均衡问题。实验结果显示该系统具有很好的查询准确率和负载均衡性能。

关键词：对等网络；语义查询；位置敏感函数
中图分类号：TP393 **文献标识码**：A **文章编号**：1009-5896(2009)03-0707-04

Semantic Search Based on Structured P2P

Hou Xiang-song[①] Cao Yuan-da[②] Guan Zhi-tao[③] Zhang Yu[①]
[①](*School of Computer Science and Technology*, *Beijing Institute of Technology*, *Beijing* 100081, *China*)
[②](*School of Software*, *Beijing Institute of Technology*, *Beijing* 100081, *China*)
[③](*Department of Computer*, *North China Electric Power University*, *Beijing* 112206, *China*)

Abstract: Peer-to-Peer (P2P) overlays are appealing, since they can aggregate resources of end systems without relying on sophisticated infrastructures. Unfortunately current peer-to-peer systems either offer exact keyword match or provide inefficient text search methods through centralized indexing or flooding. In this paper, a semantic search system is proposed for structured P2P overlays without relying on message flooding. LSH is used to map semantically related text vector to nearly node, and a mechanism is carefully designed to cope with load balancing. Experimental results show that this is a steady system with high recall, good load balance.

Key words: Peer to Peer(P2P); Semantic search; Locality-sensitive hashing

1 前言

随着网络的发展，P2P技术已经成为资源共享的热门技术，结构化P2P[1,2]由于具有良好的查询准确率和可扩展性受到了广泛关注。随着P2P应用的广泛、共享资源的丰富，出现了多维范围[3,4]查询等复杂搜索技术，并且开始朝着语义查询方向发展。语义查询指的是基于内容的全文查询，一个查询请求可以是通过信息检索技术表示的内容，摘要或标题，这些查询系统的一个重要特征是将信息检索技术融入到P2P系统中。

pSearch[5]利用潜在语义索引(Latent Semantic Index，LSI)[6]将文本向量降维，并使用滚动索引(rolling index)方法来解决LSI和CAN维度匹配问题，在实验条件下该方法有较好的查询准确率，但由于P2P是个高度动态的系统，其中不停变化的文档使LSI索引效率大幅度下降，从而导致查询准确率降低。文献[7]采用位置敏感函数作为哈希函数，弥补相容哈希函数在映射时无法保留语义的问题。文献[8]也使用LSH作为哈希函数，并通过适应性的副本解决了负载均衡问题，但由于使用LSH函数效率不高，使其查询成功率很低。文献[9]提出了一种LSH forest，并将其运用在P-GRID之上。文献[10]基于P2P信息检索系统的特性，提出了一种完全分布式的查询结果排序与合并策略，包括元数据管理策略、查询结果的排序与合并的实现。

2007-10-24 收到，2008-03-17 改回

2 系统设计

2.1 文本索引

文档采用传统的向量空间模型(Vector Space Model，VSM)[11]处理，每个文档表示为一个高维向量的形式。对于每个文档向量 $\boldsymbol{v}$ ，利用文献[12]的位置敏感函数(Locality-Sensitive Hashing，简称LSH)获得一个整数 $h(v)$ ， $h_i(v)=\lfloor(a_i\cdot v+b)/h\rfloor$ 然后根据 $\boldsymbol{g}(v)=\{h_1(v),h_2(v),\cdots,h_k(v)\}$ 可以得到一个 k 维向量 $\boldsymbol{g}(v)$ ，这时候就可以把一个高维的文本向量映射为一个 k 维向量。

在CAN网络中，两个节点间的差异程度也可以用欧几里德距离来衡量，路由的过程就是找到和发起节点间距离最小

的节点。在CAN中每个节点负责一个区域，所有落在节点负责的区域内的索引数据都由该节点负责。我们把降维映射后的文本向量利用CAN的算法发布到CAN网络中，由于LSH函数具有位置保持性，因此相似的文档就会索引到相同或相近的节点上。

仅仅使用一个LSH函数不能保证相似的文本能哈希到相近的空间上，为了增加准确率，我们定义函数组 $G=\{g:s\to U^l\}$，这里 $\boldsymbol{g}(v)=(h_1(v),\cdots,h_k(v))$。我们对向量 $\boldsymbol{v}$ 使用 $\boldsymbol{g}_i(v)(i=1,2,\cdots,l)$ 进行处理，将得到的 l 个 k维向量全部映射到CAN的 d维空间中，这样每个文档向量在CAN中会有 l 个副本。

2.2 语义搜索

文档经过索引后，相似的文档索引存储在相同或相近的节点上。节点发起一个查询请求，首先用LSH将查询请求映射为 d维空间的一个点，然后根据CAN的路由算法找到负责该点的节点。只要搜索该节点的本地数据库就可以获得相似度很高的文档。但存在一种情况，两个文档的相似度很高(距离很小)，但负责它的节点不是同一个而是相邻的两个节点。为了增加准确性，我们让查询请求在目标节点的邻居间进行广播查询请求，并将广播请求返回结果和本地结果相似度最高的文档作为最终结果返回给查询节点。

如图1所示，文档 A 经过LSH映射成空间中的点 a，而文档 B,B' 映射为点 b、b'。虽然 a 和 b、b' 的距离相等，但他们映射后又可能由两个不同的节点来负责。当查询 q 路由到节点 n，如果只是搜索节点 n 本地数据库就会丢失很多存储在节点 n 的邻居节点上的相似性很大文档，这时候节点 n 进行有限制的广播查询，获得其邻居节点上的相似性文档。

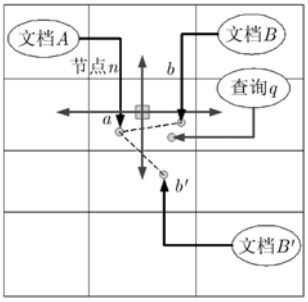

图1 语义搜索过程

3 负载均衡

由于使用LSH有可能在系统中出现大量的内容存储在某个或相邻几个节点的现象，这样就使整个系统的负载非常不均衡。我们的解决方法是在资源发布的时候使用类似Cuckoo Hashing[13]的方法将发布的内容分流到非热点区域，同时采用将新节点加入到自身感兴趣的热点区域来解决负载均衡问题。

3.1 索引的负载均衡

当节点要发布自己的文本时，首先利用VSM获得相应的文本向量，然后以其为Key路由到相应的节点上。如果该节点超过系统设定索引阈值而过载，则从本地踢出一部分内容，并用不同LSH函数重新索引，这时踢出的内容会重新发布到非热点区域，这样就可以保持节点的负载均衡。

我们根据不同LSH参数生成两个相异的哈希函数组 g,g'。假设系统设定的副本数为 l，当节点要发布一个数据时，首先用哈希函数 g 产生 l 个目标节点，然后将这数据发布到这 l 个节点上，如果某个目标节点超过阈值，则该节点从本地数据库中选出一个最近最少使用且没有被踢出数据 x 用函数 g' 来哈希处理，并把该数据发布到相应的节点。如果节点 $g'(x)$ 也达到阈值，则继续在该节点踢出一个数据。在文本索引过程中，有可能出现踢出操作循环而死锁的现象，对此可以设定一个循环次数，当循环到一定次数，可以通过更换踢出的数据 x 来继续本操作。

由于索引过程使用Cuckoo Hashing，使得每个节点所负责的索引数量趋向于平均，在系统中文档的变化不大、文档数量很小的情况下，该索引过程就能达到很好的负载均衡性能。

3.2 兴趣热点加入算法

Cuckoo Hashing在数据发布过程中解决了负载均衡的问题，如果系统一直这么运行下去，数据的存储会由于负载的不均衡而不断被踢出，导致系统查询效率下降，并且会在某些区域产生大量的索引文档，当超过设定的阈值时就称该节点所在的区域为热点。对于一个节点，其兴趣热点定义如下：

定义1 对于节点 N 和节点 H，如果节点 N 自身共享文本中 d 个最高频关键字通过LSH映射后其值落在节点 H 所负责的区域内，则称节点 H 为节点 N 的兴趣节点，如果该节点索引数量超过系统规定的阈值，则节点 H 为节点 N 的兴趣热点。

由定义可知，一个节点和自己的兴趣节点都对某个主题感兴趣，因此可以将节点加入到自己感兴趣的热点区域。当有新节点申请加入时，首先在将本节点共享的文本进行排序，获得 d 个出现频率最高的热点关键字，然后用LSH函数将热点关键字映射为 k维向量，路由到相应的兴趣节点，再以该兴趣节点为中心，获得半径 r 内的超出阈值数最多的兴趣热点，返回给新加入的节点。根据CAN的节点加入算法，将新节点加入到该区域。

假定新节点 Q 要加入到P2P中，其本地要共享的文本集合为 T，则兴趣热点加入算法如下：

步骤1 Q 对本地数据库文本集合 L 根据相似度进行排序，获得本节点的 d 个热点关键字 $\mathrm{Top}_d(T)$，然后用LSH函数组 g 生成 k维向量 $\boldsymbol{V}_k$。

步骤2 路由到负责 $\boldsymbol{V}_k$ 的兴趣节点，然后获得以该节点为中心半径 r 内的超过阈值数最多的兴趣热点 I。

步骤3 节点 I 和节点 Q 根据CAN的节点加入算法分裂区域，并更新路由表。

通过控制节点加入的区域，可以向热点区域加入新的节

点，并且该区域是新节点自身所感兴趣的区域，这样在解决负载均衡问题的同时也可以改善搜索的效率。

当节点离开的时候，节点从邻居节点中选择一个负载最小的执行合并操作，如果合并后负载超过阈值，则利用Cuckoo Hashing算法踢出若干索引。

4　文本检索算法

当节点q发起查询Q时，首先根据两组LSH哈希函数组g,g'得到p_1，p_2个q在LSH空间的d维向量$\boldsymbol{g}(Q),\boldsymbol{g}'(Q)$。对于每一个$d$维向量，根据CAN的路由算法找到目标节点$N_i$，然后以$N_i$为中心，寻找半径$r$内的节点，每个收到查询的节点将本地数据库中相似度最高的文档返回给节点N_i；节点N_i将所有和查询相似度最高的结果返回给节点q，节点q将所有收到的结果和Q比较，然后根据相似度的结果从相应节点获得文档。

节点q要查询关键字Q的文本，算法如下：

步骤1　节点q将关键字Q通过VSM转换为d维向量$\boldsymbol{Q}_d$，然后将向量$\boldsymbol{Q}_d$用LSH函数g,g'处理，生成p_1个k维向量$\boldsymbol{g}(Q_d)_i$，p_2个k维向量$\boldsymbol{g}'(Q_d)_i$。

步骤2　对于每一个k维向量，根据CAN的路由机制找到负责该向量的节点N_q。

步骤3　以N_q为中心，r为半径，进行广播查询，收到查询的节点检查本地数据库，将相似度最高的返回给N_q，N_q将所有接受的结果再次进行相似度排名，将相似度高的d个结果返回给发起节点q。

步骤4　节点q将所有返回的数据进行相似度排名，获得查找结果。

我们可以通过控制p_1，p_2的值来使搜索在LSH函数g, g'的哈希空间中进行不同程度的搜索，当系统负载较大而使数据分布严重不均衡时，数据被踢出到g'哈希空间的次数增大，我们可以通过增加p_2的值来使搜索在g'空间中进行更多的搜索，反之，通过减少p_2增大p_1来使搜索更多地在g空间中进行，这样就可以提高搜索准确率。

5　实验

为了验证系统的性能，建立了一个仿真的 CAN 环境。随机生成的1000 维数据作为 VSM 处理后的文本数据集。系统最重要的性能指标是返回的文档和查询的匹配程度，对于查询 q，首先在文档数据集中根据传统的文本检索技术找到相似度最大的 A 个，然后通过本文的 P2P 语义检索系统找到 B 个相符的文本，则系统的查询准确率可以定义为

$$R=\frac{A\bigcap B}{A}$$

实验中的缺省参数如表 1，没有特别注明，均采用表中参数。

5.1 准确度

图 2，图 3 给出了参数 k、l对准确率的影响。图 2 反映了 CAN 的维度对查询准确率的影响，可以看出，在目标节点广播范围一定时，随着维度的增加，准确率上升。在维度大约 50 左右，准确率为 75%左右，再增加维度对查询准确率的影响很小。这样就可以将高维的文本向量在 50 维左右的 CAN 网络中进行处理，并且可以获得不错的准确率。

表 1 实验参数

参数	数值	说明
D	50000	文档数
N	1024	节点数
h_1	10	LSH 函数 g 映射的区间
h_2	5	LSH 函数 g' 映射的区间
k	50	CAN 的维数，同时也是 LSH 函数组中的函数个数
l	5	文档在 CAN 中的副本数目，也是 LSH 函数组的个数
p_1	6	每次查询的并发数目
p_2	2	每次查询的并发数目
r	2	目标节点对周围节点广播范围
Tr	250	节点负载的最大文档数

图 3 给出了系统中每个数据的副本数和准确率的关系。随着副本数的增加，查询的准确率基本呈线性增加。因为副本数的增加意味着系统中负责该内容的节点数增加，同时每个节点负责的文档数增加，因此查找的过程中更容易命中目标。当系统的副本数在 8 时，基本可以保证准确率在 80%以上。

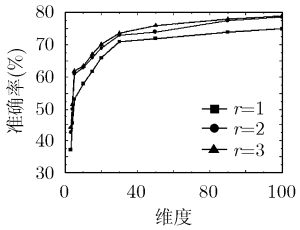

图 2 维度对准确率的影响

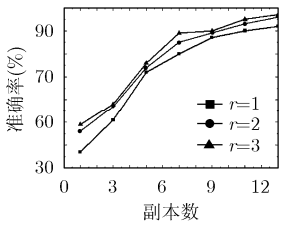

图 3 副本数和准确率的关系

从图 2，图 3 中可以看出，随着目标节点广播半径的增加，准确率也增加，但增加的幅度不是特别大，因为 LSH 已经让相近的内容哈希到相近的节点，很好地反映了语义，在很小的广播半径内就能获得很好的搜索效果。

图 4，图 5 给出了系统规模的变化对查询准确率的影响。如图 4 所示，在系统中文档数一定的情况，随着系统规模的增加，每个节点负责的文档数减少，每次查询获得的文档总数也相应减少，因此准确率有所下降。在图 5 中，当系统中节点数一定，随着文档规模的增加，每个节点所负责的文档增加，相应的准确率也增加。

节点发起一个查询的时候，需要考虑对哈希函数组 g,g'

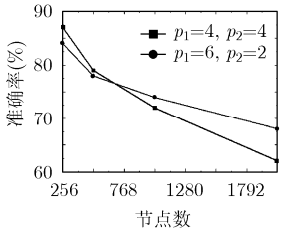

图 4 节点数对查询的影响

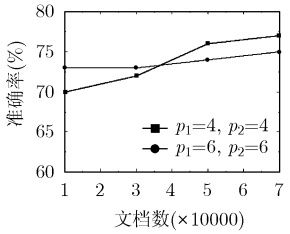

图 5 文档规模对查询的影响

的分配问题。当系统负载比较小的时候，踢出操作很少，增加 g 可以提高查询准确率；而当系统负载变大的时候，系统会有大量的踢出操作，增加 g' 的比例可以一定程度上增大查询的准确率。如图 4 所示，当系统的文档规模一定时，增加节点将导致系统的负载减少，因此使用 $p_1=6, p_2=2$ 的查询组合可以提高查询准确率。而如图 5 所示，当系统节点数一定时，增加文档数将导致系统的负载增加，使用 $p_1=4$, $p_2=4$ 的查询组合可以提高查询准确率。

5.2 负载均衡

为了验证负载均衡的效果，统计系统中节点负责文档数在 $[n\times100,(n+1)\times100]$ $(n=0,1,\cdots,7)$ 的节点数目，图 6 中纵坐标维 100 的点表示文档数在 0–100 间的节点数。从图 6 中我们可以看出，没有使用 Cuckoo Hashing 时，系统存在很多热点，节点负责的文档数很大，实验中我们发现有的节点负责的文档数最大达到 4342 个，这将导致这些热点成为系统的瓶颈。使用了 Cuckoo Hashing 后，节点的负载在 140–230 之间，有效地减少了热点的产生。

如果只采用 Cuckoo Hashing 作为负载均衡的手段，随着系统的运行，文档将不停被踢出到其他节点，导致文档将越来越分散，失去了 LSH 函数的聚类特性。实验中节点数开始为 1024，每次加入 256 个新节点，在加入节点的时候使用热点分裂技术，将其加入到系统的热点区域。图 7 中给出了使用热点分裂后系统中满负荷的节点数，可以看出，使用热点分裂技术后，系统中超过阈值的节点有明显的下降。

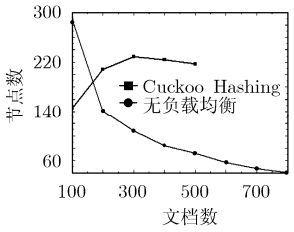

图 6 Cuckoo Hashing 对负载均衡的影响

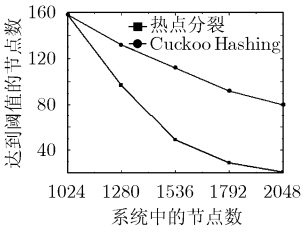

图 7 热点分裂对负载均衡的影响

6 结束语

本文提出了一种结构化 P2P 环境中的语义查询模型。使用 LSH 函数将高维的文本向量映射到低维的空间中，并且在映射过程可以保持语义相关性。为了解决非相容哈希带来的负载均衡问题，本文采用 Cuckoo Hashing 在发布数据的时候将数据调整到非热点区域；当有新节点加入时，将其加入到自己感兴趣的热点区域内。实验结果表明该系统具有很好的查询准确率，并且解决了负载均衡问题。

参 考 文 献

[1] Ratnasamy S, Francis P, and Handley M, *et al.*. A scalable content-addressable network. Proceedings of ACM SIGCOMM 2001. San Diego, CA: 2001, Vol. 31: 161–172.

[2] Stoica I, Morris R, and Karger D, *et al.*. Chord: a scalable peer-to-peer lookup service for Internet applications. ACM SIGCOMM 2001. San Diego, CA, USA: 2001, Vol. 31: 149.

[3] Shu Y, Ooi B C, and Tan K L, *et al.*. Supporting Multi-dimensional Range Queries in Peer-to-Peer Systems. Fifth IEEE International Conference on Peer-to-Peer Computing, P2P 2005. Konstanz, Germany: 2005: 173–180.

[4] 侯祥松, 曹元大. 一种支持结构化 P2P 的多维范围查找方法. 北京理工大学学报, 2007, 27(6): 517–520.
Hou Xiang-song and Cao Yuan-da. Structured P2P search method to support multi-dimensional range queries. *Journal of Beijing Institute of Technology*, 2007, 27(6): 517–520.

[5] Tang C, Xu Z, and Mahalingam M. pSearch: Information retrieval in structured overlays. *Computer Communication Review*, 2003, 33(1): 89–94.

[6] Berry M W, Drmac Z, and Jessup E R. Matrices, vector spaces, and information retrieval. *SIAM Review*, 1999, 41(2): 335–362.

[7] Zhu Y, Wang H, and Hu Y. Integrating semantics-based access mechanisms with P2P file systems. Proceedings of the third International Conference on Peer-to-Peer Computing, 2003. (P2P 2003). Sweden: 2003: 118–125.

[8] Bhattacharya I, Kashyap S R, and Parthasarathy S. Similarity Searching in Peer-to-Peer Databases. Proceedings of the 25th IEEE International Conference on Distributed Computing Systems, 2005. ICDCS 2005. Columbus, Ohio, USA: 2005: 329–338.

[9] Bawa M, Condie T, and Ganesan P. LSH forest: Self-tuning indexes for similarity search. Proceedings of the 14th international conference on World Wide Web. Chiba, Japan: 2005: 651–660.

[10] 凌波, 周水庚, 周傲英. P2P 信息检索系统的查询结果排序与合并策略. 计算机学报, 2007, 30(3): 405–414.
Ling Bo, Zhou Shui-Geng, and Zhou Ao-Ying. A strategy of query result ranking and merging for P2P information retrieval systems. *Chinese Journal of Computers*, 2007, 30(3): 405–414.

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

[12] Datar M, Indyk P, and Immorlica N, *et al.*. Locality-sensitive hashing scheme based on p-stable distributions. Proceedings of the Twentieth Annual Symposium on Computational Geometry (SCG'04). Brooklyn, NY, United States: 2004: 253–262.

[13] Pagh R and Rodler F F. Cuckoo hashing. Proceedings of the 9th Annual European Symposium. Algorithms-ESA 2001. Arhus, Denmark: 2001: 121–133.

侯祥松： 男，1977 年生，博士生，研究方向为分布式计算、信息安全.

曹元大： 男，1944 年生，教授，博士生导师，研究方向为信息安全、分布式计算.

关志涛： 男，1979 年生，讲师，研究方向为信息安全，对等计算.

张 昱： 男，1978 年生，博士生，研究方向为分布式计算.