ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.17, No.4, April 2006, pp.782−793
DOI: 10.1360/jos170782

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

混合 P2P 环境下有效的查询扩展及其搜索算法*

张 骞+, 张 霞, 刘积仁, 孙 雨, 文学志, 刘 铮

(东北大学 计算机软件国家工程研究中心,辽宁 沈阳 110179)

Query Expansion and Its Search Algorithm in Hybrid Peer-to-Peer Networks

ZHANG Qian+, ZHANG Xia, LIU Ji-Ren, SUN Yu, WEN Xue-Zhi, LIU Zheng

(National Engineering Research Center for Computer Software, Northeastern University, Shenyang 110179, China)

+ Corresponding author: Phn: +86-24-83661102, E-mail: zhangqian@neusoft.com, http://www.neu.edu.cn

Zhang Q, Zhang X, Liu JR, Sun Y, Wen XZ, Liu Z. Query expansion and its search algorithm in hybrid peer-to-peer networks. *Journal of Software*, 2006,17(4):782−793. http://www.jos.org.cn/1000-9825/17/782.htm

Abstract: Query expansion has long been suggested as a technique for dealing with the fundamental issue of word mismatch in information retrieval and it has gained great success in Web searching. However, processing query expansion is very challenging in hybrid P2P network because a P2P system is a decentralized and dynamic system. First, the LEM query expansion method, which is constructed by analyzing correlation between queries and documents, is presented. And then, the HEM query expansion method is proposed by establishing the correlation between queries and documents terms directly. Next, an efficient search algorithm is constructed based on the query expansion algorithms. It is proved by experiments that the query expansion methods and search algorithms can greatly improve the search efficiency.
Key words: query expansion; peer-to-peer; query note; similarity; search

摘 要: 查询扩展是解决信息获取领域中用词歧义性问题的关键技术,并被广泛应用于搜索引擎中,获得了巨大的成功.然而,由于 P2P(peer-to-peer)系统是一个分散的、动态的系统,在 P2P 环境下进行有效的查询扩展具有一定的挑战性.首先,利用查询与文档的关联关系构建了 LEM(local expansion method)查询扩展方法;然后,基于查询与文档用词的直接关联,提出了 HEM(history_based expansion method)查询扩展方法.在此基础上,提出了一种基于查询扩展的混合 P2P 环境下的搜索算法.实验及分析结果表明,查询扩展及其搜索算法能够极大地提高搜索的效果.
关键词: 查询扩展;peer-to-peer;查询记录;相关度;搜索
中图法分类号: TP311 文献标识码: A

Peer-to-Peer(P2P)是当前研究的热点.目前,很多 P2P 系统支持基于关键词的搜索[1,2].然而,由于大量同义词和多义词的存在,用户提交的查询用词往往与文档索引使用的词有很大差别,这就是所谓的“词典问题(dictionary problem)”[3].“词典问题”的存在,限制了 P2P 系统的应用.

Furnas 就“词典问题”所做的实验表明:两个人使用同样的关键词描述同一事物的概率通常小于 20%.同样

* Supported by the National High-Tech Research and Development Plan of China under Grant No.2002AA4Z3120 (国家高技术研究发展计划(863))

Received 2005-06-28; Accepted 2005-10-10

有研究表明:网络用户用于检索的查询 85%是短查询,该类查询一般包括 3 个或更少数目的查询单词[4].因此,用户提交的查询通常不能充分表达出检索相关文档所需的信息.目前,Web 搜索引擎广泛采用自动查询扩展方法来解决这个问题,并获得了成功.即在原来查询的基础上,加入与用户用词相关联的词组成新查询,这在一定程度上弥补了用户查询信息的不足.我们希望使用 P2P 系统也能像使用 Web 搜索引擎那样,当输入一个查询时,系统能自动进行查询扩展,返回最相关的结果.因此,在 P2P 系统中进行查询扩展优化十分必要.

由于 P2P 系统中不存在中心控制节点,网络中的节点也可以随意加入或退出系统,因此,在 P2P 环境下进行查询扩展优化是十分具有挑战性的.本文提出了两种 P2P 环境下的查询扩展方法:局部查询扩展方法 LEM(local expansion method)和基于历史查询信息的查询扩展方法 HEM(history_based expansion method).设定的场景为非结构化文档上的基于关键字的查询.

本文第 1 节介绍现有的查询扩展技术,随后依次介绍局部查询扩展方法和基于历史查询信息的扩展方法,并在此基础上提出一种新的查询路由算法.最后是仿真实验及分析.

1 相关工作

据我们所知,目前还不存在有效的 P2P 环境下的查询扩展方法.因此,这里对传统的查询扩展技术作简要介绍.查询扩展技术主要分为全局分析与局部分析两类.

全局分析的基本思想是统一对全部文档中的词或词组进行相关性分析,计算词或词组与查询之间的相关度.当查询到来时,使用与查询相关度最高的文档用词作为新生成的查询用词.LSI(latent semantic indexing)[5]等是常见的全局分析方法.该类方法的主要缺陷在于:对于非常大的文档集,构建全局性的词关系词典通常在时间和空间上是不可行的,并且词典的更新将产生较高的代价.显然,这不适合文档数量巨大及高度动态的 P2P 网络.

局部分析可以进一步分为相关反馈(relevance feedback)[6]和局部反馈[7]两类.相关反馈需要用户对初次检索的结果进行评判,然后从用户认为相关的文档中选择扩展用词.局部反馈不需要用户的参与,它从初次检索的前 N 篇文档中选择扩展用词.但局部反馈的查询精度高度依赖于排在前面的文档与查询的相关度:当相关度不大时,局部反馈会把大量无关的词作为扩展用词,从而降低了查询精度.在 P2P 网络中,由于节点在计算能力、网络带宽等方面的异质(heterogeneity)性,并非每个节点都适合直接采用局部分析的方法进行查询扩展.

Xu 将全局分析的技术应用于局部反馈,提出了局部上下文分析方法[8],研究表明,该方法的检索效果优于传统的全局分析和局部分析方法.因为本文提出的 LEM 方法是对局部上下文分析方法的修改,因此,这里对局部上下文分析方法进行详细描述.该方法的基本思想是:从初始检索得到的前 N 篇文档中选择与原查询相关度最高的文档用词作为新的查询用词,相关度依据下式计算

$$simi(Q,c)=\prod_{t_i\in Q}[\log(cf(c,t_i)+1)\times idf_c/\log(N)+\delta]^{idf_i} \tag{1}$$

$$cf(c,t_i)=\sum_{j=1}^{j=N} ft_{i,j}\,fc_j,$$

$$idf_i=\max(1.0,\log10(N/N_i)/5.0);\quad idf_c=\max(1.0,\log10(N/N_c)/5.0).$$

其中:$simi(Q,c)$为查询与词或词组的相关度;$cf(c,t_i)$表示查询用词和文档用词共同出现的频率;N_i 和 N_c 分别是包含查询用词 t_i 和文档用词 c 的文档数目.选取 m 个相关度最高的文档用词作为新的查询用词.

2 混合 P2P 网络

混合 P2P 网络中存在两类节点:目录节点(directory node)和叶节点(leaf node).相对于叶节点,目录节点具有更高的可信度,在网络中担负更大的责任.叶节点以“星型”方式连接到目录节点上,叶节点发起的查询首先被路由到目录节点,由目录节点依据一定的策略路由查询,最后有目录节点收集查询结果并返还给叶节点,叶节点在结果中选择并通过点击连接直接下载文档.如图 1 所示,叶节点 C 发出查询请求,图中的箭头表明查询结果的返回路径,目录节点 D 负责对叶节点 F,G 及目录节点 E 返回的查询结果进行合并,并将合并后的结果返回给 C,C 依据自己的判断点击连接直接下载文档.下面给出 P2P 环境下查询记录的定义.

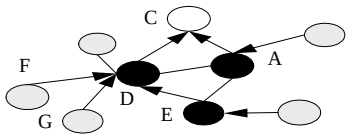

Fig.1 Retrieval in hybrid P2P networks

图 1 混合 P2P 环境下的信息获取

定义 **1**. 令 *Note* 为节点保存的历史查询记录,即 *Note*=(*query text*)[(D_i,ID_i,D_WORD_i)]…其中,(*query text*)为 *Note* 对应的查询;D_i 为用户从标识为 ID_i 的 Peer 节点下载的文档集.将每个文档中诸如 a,the,is 等出现频率极高的词删除,组成文档词集 D_WORD_i.每次查询用户可能从多个节点下载文档,因此每条查询记录可能包含多个(D_i,ID_i,D_WORD_i)对.

基于上述定义,易得下述假定是合理的:

(1) D_i 代表的文档集与(*query text*)具有很强的相关性.用户浏览搜索结果并点击下载文档,这一行为本身就包含着用户对文档和查询之间相关性的判断;

(2) (*query text*)中包含的查询单词与 D_WORD_i 包含的文档单词具有很强的相关性.这种相关性可以作为选择查询扩展词的依据;

(3) 若历史记录中标识为 ID_i 的节点与查询单词及其扩展词有较强的关联关系,则认为该节点拥有的文档与查询具有很强的相关性.

基于上述假定,可以建立查询空间、文档空间、文档用词和 Peer 节点间的关联关系,利用这些关系可以进行有效的查询扩展及搜索.后续章节将对此详加论述.

3 LEM 扩展方法

本节试图对局部上下文分析方法进行修改,使之适用于混合 P2P 环境.由于弱 Peer 节点的存在,该类节点一般具有弱的计算力和存储力,因而通常不能在该类节点上直接应用局部上下文分析方法;另一种可能的方法是充分利用目录节点的计算及存储优势,在目录节点上应用局部上下文分析方法,即由叶节点发送查询到与之连接的目录节点,由目录节点采用局部分析方法对查询进行扩展.然而,该方法仍然面临以下挑战:

(1) 由第 1 节的分析可知,局部上下文分析方法的应用效果高度依赖于初次检索得到的文档.因此,问题的关键是如何在目录节点上确定与查询最相关的 N 篇文档;

(2) 每次都需要先进行初始检索,然后利用初始检索得到的前 N 篇文档进行查询扩展.这不仅浪费了网络资源,而且不利于提高用户满意度.

问题转化为在不进行初次检索的情况下,如何在目录节点上获取与查询最相关的 N 篇文档.我们的方法是:每次查询完成后,叶节点将查询记录发往相应的目录节点.目录节点利用保存的历史查询记录构造查询用词空间和文档空间的关联关系(如图 2 所示),当新查询到来时,利用这种关联关系获取与查询最为相关的 N 篇文档.

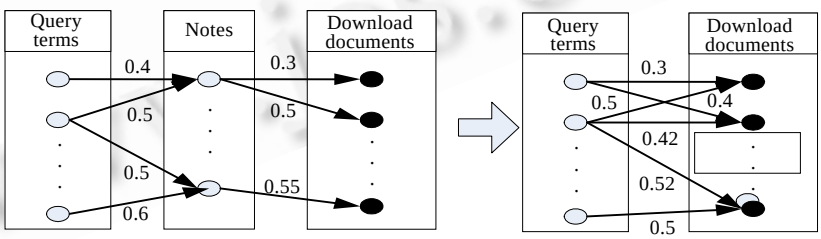

Fig2. Correlation between query terms and download documents

图 2 查询用词空间与文档空间的关联关系

在查询记录中,若查询用词 t_i 与文档 D_c 间至少存在一条可达路径,则创建一条 t_i 与 D_c 的直接连接,并依据 t_i 与 D_c 的相关程度为该连接赋予一定的权重.对任意查询用词 t_i 与文档 D_c,关联权重记为 $Weight_{i,c}$,可以通过两者在历史查询记录中出现的频率来估算.

$$Weight_{i,c}=\frac{f_{i,c}(t_i,D_c)}{f(t_i)}\times T_{i,c}\times I_i \tag{2}$$

$$T_{i,c}=\frac{qf_{i,c}}{qf_{i,c}+(qw_c/avg\_qw)},\qquad I_i=\frac{\log((n+0.5)/nf_i)}{\log(n+1.0)}.$$

其中,$f_{i,c}(t_i,D_c)$是查询用词 t_i 与文档 D_c 同时出现的历史记录数目;$f(t_i)$是 t_i 出现的历史记录数目;$qf_{i,c}$ 是 t_i 在文档 D_c 中出现的频率;qw_c 是 D_c 包含的单词数目;avg_qw 是历史记录中全部文档包含的平均单词数目;I_i 类似于反排文档索引;nf_i 是历史记录中包含 t_i 的文档数目;n 是历史记录中包含的总的文档数目.$f_{i,c}(t_i,D_c)$的含义是:如果 t_i 与 D_c 以较高的频率共同出现,则认为 t_i 与 D_c 高度相关;$T_{i,c}$ 和 I_i 分别类似于 CORI[9]资源选择算法中的 T_i 和 I_i,即 t_i 相对于 D_c 的权重与 t_i 在 D_c 中出现的频率成正比,与包含 t_i 的文档数目成反比.当新查询到来时,利用查询用词空间与文档空间的关联关系计算查询与文档的关联度,计算方法借鉴了文献[9]中的方法.

$$score(Q,D_c)=\frac{1}{|Q|}\sum_{i=1}^{|Q|}Weight_{i,c} \tag{3}$$

其中,$score(Q,D_c)$是查询 Q 与文档 D_c 的关联度.根据关联度,可以选择出与 Q 最为相关的 N 篇文档,然后利用局部上下文分析方法,从 N 篇文档中选取与 Q 最为相关的 m 个词或词组加入原查询组成新查询.

显然,历史记录信息越丰富,其包含的用户关于查询用词与文档的关联性判断就越多,就越有利于提高查询扩展的效果.但对于某些新加入的节点,这类节点通常只有少量的历史查询记录,因而影响到扩展的效果.另外,查询记录是随时间推移递增的,这可能会给某些弱节点带来严重的存储负担.因此,在每次查询完成之后,叶节点需要将查询记录发往相应的目录节点,由目录节点构建查询用词空间与文档空间的关联关系,查询扩展在目录节点上进行.

4 HEM 扩展方法

第 3 节在目录节点上建立了查询用词空间与文档空间的关联关系.实际上,可以借助文档空间将查询空间与文档用词空间直接关联起来,查询单词与文档单词的关联权重可作为选择查询扩展词的依据[10].

设历史记录中包含的文档集为 D,查询用词 A 与文档用词 B 的关联权重设为 B 相对于 A 的条件概率[10],

$$P(W_B|W_A)=P(W_B,W_A)/P(W_A)=\frac{\sum_{d_i\in D}P(W_A,W_B,d_i)}{p(W_A)} \tag{4}$$

其中,$P(W_B,W_A,d_i)=P(W_B|W_A,d_i)\times P(W_A,d_i)=P(W_B|W_A,d_i)\times P(d_i|W_A)\times P(W_A)$.$P(W_B|W_A,d_i)$可根据 d_i 在记录中出现的频率及 A 和 B 在 d_i 中的频率得出 $P(W_B|W_A,d_i)=\frac{FD_{i,B}}{size(d_i)}\times\frac{FD_{i,A}}{size(d_i)}\times\frac{count(d_i)}{count(D)}$.其中,$FD_{i,A}$ 和 $FD_{i,B}$ 分别为单词 A 与 B 在文档 d_i 中的频率;$size(d_i)$为文档 d_i 的大小.$count(D)$为整个历史记录中总的文档数目.$count(d_i)$为 d_i 在全部历史记录中出现的数目;$P(d_i|W_A)$表示在 A 出现的情况下 d_i 出现的概率,可以根据 d_i 与 A 在历史记录中同时出现的频率计算得出 $P(d_i|W_A)=\frac{f(W_A,d_i)}{f(W_A)}$.其中,$f(W_A)$为包含 A 的记录数目;$f(W_A,d_i)$为同时包含 d_i 和 A 的记录数目.于是

$$P(W_B|W_A)=\sum_{d_i\in D}\frac{FD_{i,B}}{size(d_i)}\times\frac{FD_{i,A}}{size(d_i)}\times\frac{count(d_i)}{count(D)}\times\frac{f(W_A,d_i)}{f(W_A)} \tag{5}$$

然后,可以计算单词 B 与查询 Q 的相似度 $score(Q,B)$.

$$score(Q,B)=\frac{1}{|Q|}\sum_{A\in Q}P(W_B|W_A) \tag{6}$$

对任意查询 Q,针对历史记录中的全部文档用词,根据式(6)计算用词与 Q 的相似度,并将结果按降序排列,选择相似度大于某一阈值 σ 的前 m 个结果作为 Q 的新扩展词.与 LEM 方法一样,HEM 方法也要求在每次查询完成后,叶节点需要将查询记录发往相应的目录节点,查询扩展在目录节点上进行.

5 查询记录的获取及更新

每次查询完成之后,LEM 和 HEM 方法都要求叶节点将查询记录发往目录节点,而查询记录有查询单词、

文档单词及 Peer 标识组成,因而产生较高的通信开销.这里采用如下方法来减少这种开销:

(1) 叶节点发起的查询路由到达与之连接的目录节点时,目录节点保存该查询的一个副本.这样,可以避免查询单词的重复发送;

(2) 叶节点下载的文档可能是其邻居节点已经下载过的,因而该文档所包含的单词在目录节点上已经存在,所以叶节点没有必要重复发送这些文档单词.具体方法是:目录节点考察返回的查询结果 R,并检查保存的查询记录 N,若存在非空文档集合 D,满足 $D \in R \cap N$,则将 R 中包含的与 D 中文档相关的结果作上已"获取过"标识"*retrieval*",并对所有结果进行编号(记为 $result\_id_i$).然后,将 R 返回给查询发起节点;

(3) 有研究表明[11],文档单词的重要性与其频率的排序往往满足 Zipf 定律.因此,在文档中仅出现 1 次的单词数量往往占据总单词数量的一半,抛弃对这些单词的传输可以使通信代价大为减少;

(4) 叶节点计算本次查询涉及到的查询用词和文档用词的相关度,针对每个查询词,选出与该词最为相关的 k 个文档用词发往目录节点.方法如下:

I) 叶节点利用本地维护的历史查询记录建立查询用词和文档用词的关联关系,查询用词和文档用词的连接权重表示两者的相关程度,权重可以根据式(4)计算得出;

II) 对于本次查询叶节点下载的文档中没有"*retrieval*"标识的文档,记经过方法(3)处理后的文档所包含的词集为 *download_word*.对于查询用词 A,根据式(4)计算 $P(W_B|W_A)(B \in download\_word)$,据此选取 k 个与 A 最为相关的文档用词,记为 $word_{A,k}$;

III) 计算 $word_{All,k}=\cup_{A \in Q}(word_{A,k})$,将 $word_{All,k}$ 发往目录节点.

然而,由于受到计算力等方面的限制,方法(4)对于弱 Peer 节点而言并不适用,该类节点通常不往目录节点发送任何查询记录.如图 3 所示,D 是目录节点,由于 C 是弱节点,因此 C 不往目录节点 D 发送任何查询记录,查询记录的发送由叶节点 F 和 G 承担.

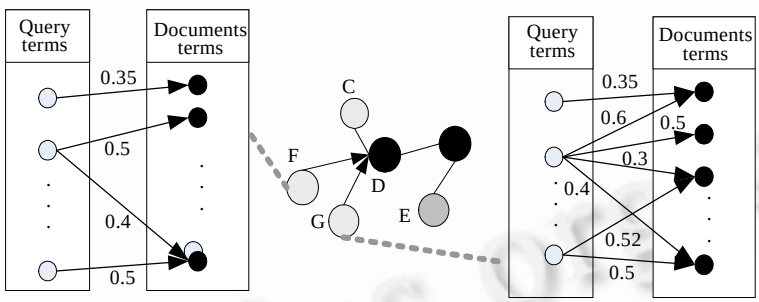

Fig.3 Sending of query notes

图 3 查询记录的发送

综上,每次查询完成之后,查询发起节点需要考察与所下载文档相关的查询结果是否有"*retrieval*"标识,然后执行以下步骤:

(1) 针对没有"*retrieval*"标识的结果所对应的文档执行方法(3)和(4),即从出现频率大于 1 的文档用词中选出$|word_{All,k}|$个,连同文档来源节点标识 ID_i 一起发送到相应的目录节点;

(2) 对于有"*retrieval*"标识的结果,仅需将结果编号及源节点标识发往目录节点.目录节点收到该类消息后,通过 $result\_id_i$ 确定相应的文档单词.

目录节点收到查询发起点的全部反馈消息后,需要更新目录节点中保存的关联权重.限于篇幅,这里仅讨论 HEM 方法.不妨记 *pre_word* 和 *c_word* 分别为更新前目录节点保存的文档词集和本次查询记录包含的文档词集,*c_file* 和 *all_file* 分别为本次查询记录包含的文档集和更新后目录节点保存的文档集,*pre_query* 和 *c_query* 分别为更新前目录节点保存的查询词集和本次查询记录包含的查询词集.对任意查询用词 A 与文档用词 B 而言,记更新前的关联权重为 $P(W_B|W_A)$,更新后的关联权重为 $P'(W_B|W_A)$,则有

$$P'(W_B|W_A)=\begin{cases}P(W_B|W_A)\times\dfrac{count(D)}{count(D')}, & \text{if } (B\in pre\_word-c\_word) \text{ and } (A\in pre\_query-c\_query)\\ P(W_B|W_A)\times\dfrac{count(D)}{count(D')}\times\dfrac{f(W_A)}{f'(W_A)}, & \text{if } (B\in pre\_word-c\_word) \text{ and } (A\in pre\_query\cap c\_query)\\ 0, & \text{if } (B\in pre\_word-c\_word \text{ and } A\in c\_query-pre\_query)\\ & \text{or } (B\in c\_word-pre\_word \text{ and } A\in pre\_query-c\_query)\\ \sum\limits_{d_i\in c\_file} f(A,B,d_i), & \text{if } (B\in c\_word-pre\_word) \text{ and } (A\in c\_query)\\ \sum\limits_{d_i\in all\_file} f(A,B,d_i), & \text{otherwise}\end{cases} \quad (7)$$

其中, $f(A,B,d_i)=\dfrac{FD_{i,B}}{size(d_i)}\times\dfrac{FD_{i,A}}{size(d_i)}\times\dfrac{count(d_i)}{count(D)}\times\dfrac{f(W_A,d_i)}{f(W_A)}$.$f(W_A)$和 $f'(W_A)$分别为查询用词 A 更新前后出现在历史记录中的频率,$count(D)$和 $count(D')$分别为更新前后历史记录中包含的文档数目.显然,LEM 和 HEM 方法在目录节点上聚集众多叶节点的历史查询记录并执行查询扩展,使得各个叶节点的查询信息得以共享,避免了单个节点因为自身信息的有限性而可能导致查询扩展的低效性.

6 基于查询扩展的搜索算法

P2P 环境下的搜索算法已成为研究热点之一[11,12].限于篇幅,我们在此仅讨论与本文相关的混合 P2P 环境下的搜索算法.混合 P2P 环境下的搜索算法主要有以下几类:

(1) 基于文件名的搜索.叶节点将所包含文档名字的 *hashing* 发送到与之连接的目录节点,查询到来时,目录节点根据保存的文件名信息将查询路由到满足条件的叶节点上,而目录节点之间的路由则采用泛洪(flooding)的方式,如 Gnutella 0.6[12]采用的方法.

(2) 基于文档词集的搜索.文档词集指的是叶节点所包含的全部文档用词集合.叶节点将自身的文档词集摘要发往与之连接的目录节点,查询到来时,目录节点通过判断查询用词与摘要的匹配程度来选择到叶节点的路由.目录节点之间的路由仍然采用泛洪的方式.

(3) 基于内容的搜索.在该类方法中,叶节点的路由选择采用了修改后的 *K-L divergence*[11]资源选择算法,该算法需要叶节点将自身包含的文档用词及其频率信息发送到目录节点.目录节点之间的路由选择则通过观察邻近目录节点对以往查询的响应,计算新查询与历史查询的相似程度来判定.文献[11]是这类算法的典型代表.

这里,我们提出了一种新的混合 P2P 环境下的搜索算法,称为 SE(search based on query expansion)算法.SE 算法利用了历史查询记录信息,查询记录不仅蕴涵了查询与文档的关联关系,也包含了查询、文档与源 Peer 的关联关系,因而可以建立查询与 Peer 的直接关联.这种关联为路由查询提供了一定的指引信息,如图 4 所示.

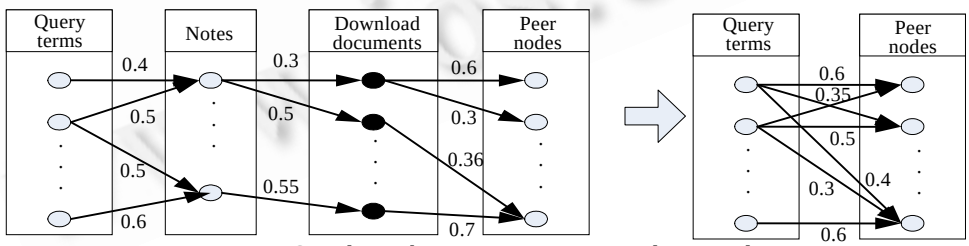

Fig.4 Correlation between query terms and peer nodes

图 4 查询用词空间与 Peer 节点的关联关系

SE 算法实质上是计算扩展后的查询与 Peer 节点的相关度,选择相关度高的节点转发查询.由图 4 可知,若查询与节点的相关度很高,则意味着该节点包含的文档与查询相关度很高,因而能为路由查询提供指引.相关度可以转化为查询用词和节点间的关联权重.在目录节点 ID_j 上,对其历史查询记录中包含的任意查询用词 t_i 和节点 $ID_{j,k}$,记其关联权重为 $Weight_{i,j,k}$.可以通过查询用词和节点在历史记录中共同出现的频率来估算.

$$Weight_{i,j,k}=\frac{f_{i,k}(t_i,ID_{j,k})}{f(t_i)}\times T_{i,k}\times I_{i,j} \quad (8)$$

$$T_{i,k}=\frac{qf_{i,k}}{qf_{i,k}+(qw_k/avg\_qw)},\qquad I_{i,j}=\frac{\log((n_j+0.5)/nf_i)}{\log(n_j+1.0)}.$$

其中,$f_{i,k}(t_i,ID_{j,k})$是查询用词 t_i 与节点 $ID_{j,k}$ 同时出现的历史记录数目;$f(t_i)$是 t_i 出现的历史记录数目;$qf_{i,k}$是 t_i 在 $ID_{j,k}$ 的文档(这里以历史记录中 $ID_{j,k}$ 对应的文档用词近似)用词中出现的频率;qw_k 是 $ID_{j,k}$ 包含的总文档用词数目;avg_qw 是历史记录中全部节点包含的平均文档用词数目;$I_{i,j}$ 类似于反排文档索引,其中 nf_i 是历史记录中包含 t_i 的节点数目;n_j 是历史记录中总的节点数目.

$f_{i,k}(t_i,ID_{j,k})$的含义是:若 t_i 与 $ID_{j,k}$ 以较高的频率共同出现,则 t_i 与 $ID_{j,k}$ 高度相关;$T_{i,k}$ 和 $I_{i,j}$ 的含义是:t_i 相对于 $ID_{j,k}$ 的权重与 t_i 在 $ID_{j,k}$ 包含的文档用词中出现的频率成正比,与包含 t_i 的节点数目成反比.计算出关联权重之后,可以进一步计算查询 Q 与节点 $ID_{j,k}$ 的相似度成绩 $score(Q,ID_{j,k})$:

$$score(Q,ID_{j,k})=\frac{1}{|Q|}\sum_{i=1}^{|Q|}Weight_{i,j,k} \tag{9}$$

综上,当查询 Q 到来时,目录节点执行 SE 算法,步骤如下:

(1) 首先,采用 LEM 或 HEM 方法将 Q 扩展为新查询 Q';

(2) 然后,依据式(9)计算各个邻居节点与 Q'的相似度成绩,选取 K 个与查询最为相关(成绩最高的)的邻居节点转发查询.

需要着重指出的是,SE 算法集中于目录节点间的路由,而目录节点之间的组织形式完全是无结构的,这种组织形式与无结构 P2P 网络的主要区别在于目录节点本身通常并不共享文档.即使将目录节点所保存的叶节点文档信息视为其本身共享的文档,但这些文档信息量通常较大,因而导致较高的通信和维护代价,使得无结构 P2P 网络中广泛存在的基于节点内容的启发式路由算法并不适用于目录节点间的路由.文献[13]提出了基于查询聚集的无结构 P2P 网络路由算法,但这类算法与上述文献[11]中的目录节点之间的路由算法本质上是一致的,即路由决策的依据完全来源于历史查询文本信息.与其不同的是,SE 算法是基于历史查询记录的,这里所说的历史查询记录既包含了历史查询文本信息,也包含了下载的文档信息;另外,SE 算法采用扩展后的查询参与路由决策,因而相对于上述算法,SE 算法能够更好地为路由查询提供指引.

值得一提的是,P2P 网络的一个重要特性就是节点可以自由加入和退出网络.特别地,由于混合 P2P 网络采用的是层次化的架构,因而频繁加入和退出的节点通常是叶节点,目录节点一般相对稳定.而且,叶节点的加入和退出不会对目录节点上的历史查询记录产生影响,即使某一目录节点所属的全部能够发送查询记录的叶节点同时退出,其他弱 Peer 节点仍然可以通过目录节点进行查询扩展;若是目录节点退出,其所属的叶节点也可以连接到新的目录节点.因此,节点的自由加入和退出对本文提出的查询扩展及其搜索算法影响不大.另外,叶节点可以根据自身的能力(如接入带宽等)设定每次返回查询结果的最大数目,而目录节点则可以根据叶节点设定的最大数目逐步返回查询结果.这里,目录节点起到一个缓存的作用.

7 性能分析

7.1 存储代价

这里考虑目录节点和叶节点上存储历史记录的代价.由第 5 节可知,叶节点利用查询记录构建查询用词、文档用词和源节点的关联关系.假定记录中包含 $N_{query\text{-}term}$ 个查询用词、$N_{doc\text{-}term}$ 个文档用词和 N_{file} 个文档,这些文档来源于 N_{count} 个节点.同时,设每个文档用词和查询用词是字符,占 1 个字节,则叶节点所需的存储空间为

$$STORAGE_{leaf\text{-}node}=N_{doc\text{-}term}+N_{query\text{-}term}+N_{doc\text{-}term}\times N_{query\text{-}term}\times 8+(N_{doc\text{-}term}+N_{query\text{-}term})\times N_{file}\times 8+N_{file}\times 8+N_{count}\times 8,$$

其中:每对查询用词和文档用词对应一个关联权重,为 double 型,占 8 个字节;每个查询用词或文档用词在文档中出现的频率为 double 型,占 8 个字节;文档大小及文档来源节点标识也为 double 型,各占 8 个字节.相对于文档用词而言,查询用词、文档大小和节点标识占用的存储空间可以忽略不计,因此有

$$STORAGE_{leaf\text{-}node}\approx N_{doc\text{-}term}\times(N_{file}+N_{query\text{-}term})\times 8.$$

对于目录节点,,假定其叶节点数目为 $N_{neighbor}$,由于计算力等方面的差异,并非全部叶节点都可以构建上述

关联关系,由此易得 $STORAGE_{super\text{-}node} \le N_{neighbor} \times STORAGE_{leaf\text{-}node} + N_{neighbor} \times N_{query\text{-}term} \times N_{neighbor} \times N_{count} \times 8$.

考虑 SE 算法,其中每对查询用词和节点对应一个关联权重,为 double 型,占 8 字节.由上述分析可知: $N_{neighbor}$ 一般不大,随着查询数目的增多,$N_{doc\text{-}term}$,$N_{query\text{-}term}$ 和 N_{count} 逐渐变大.我们可以规定 $N_{doc\text{-}term}$,$N_{query\text{-}term}$ 和 N_{count} 的最大数目,并采用 LRU(least recently used)策略进行淘汰,保证历史记录中存储常用的查询用词.

7.2 网络通信代价

通信代价用需要传输的字节数来衡量.这里,我们仅讨论查询结果返回之后,由叶节点向目录节点发送消息所产生的通信开销.假定扩展后的查询包含 N_{query} 个查询用词,下载的文档数目为 N_{doc},这些文档来源于 N_{node} 个节点,根据第 5 节的讨论,针对每个查询用词 A 选取 k 个最相关的文档用词,记为 $word_{A,k}$,即 $word_{All,k}=\cup_{A\in Q}(word_{A,k})$,同时假定每个文档用词占用 1 字节.于是有

$$communication \approx |word_{All,k}| + N_{doc} \times 4 + N_{node} \times 8 \approx N_{query} \times k + N_{doc} \times 4 + N_{node} \times 8,$$

其中,目录节点为每个查询结果做的编号为 int 型,占 4 个字节;节点标识为 double 型,占 8 字节.

因为用户提交的查询多为 1~3 个单词的短查询,且有研究表明[8],向原查询加入 30 个扩展词时查询性能达到最高,由此可得,$N_{query} \times k \approx 30 \times 30 = 900$;$communication \approx 900 + N_{doc} \times 4 + N_{node} \times 8$.

显然,下载文档越多,通信代价就越高.而对于弱节点而言,这类节点通常不保存查询记录,因而这类节点发送消息的开销为 0.

7.3 查询响应时间

查询响应时间包括网络延迟和访问节点的查询处理时间.查询处理时间又进一步分为 Peer 选择的时间、本地查询的时间、结果合并的时间和查询扩展的时间.在此,仅讨论在目录节点上进行查询扩展所需的时间及 Peer 选择的时间.假定目录节点保存的文档用词数目为 N_{term},因为原查询多为短查询,因而根据式(3)和式(6)易得查询扩展的时间复杂度为 $O(N_{term})$.假定目录节点保存的邻居节点数目为 $N_{neighbor}$,扩展后的查询长度为 Len_{query},由式(9)可得,Peer 选择的时间复杂度为 $O(N_{neighbor} \times Len_{query})$,其中查询长度一般为 30 左右.因此,总的时间复杂度为 $O(30 \times N_{neighbor}) + O(N_{term})$.随着查询记录的增加,时间复杂度也随之增大.可以规定最大的文档单词数目和邻居节点数目,使用 LRU 策略淘汰不经常使用的文档单词,并计算每个邻居节点被选择转发的频率,淘汰转发频率低的节点.

7.4 更新代价

查询完成之后,查询发起节点和目录节点需要对各自保存的查询记录进行更新.限于篇幅,这里仅讨论目录节点上的更新代价.以 HEM 方法为例,由式(7)得到更新代价为

$$UpdateCost_{supernode} = |pre\_word - c\_word| \times |pre\_query| + |c\_word - pre\_word| \times |c\_query| \times |c\_file| + |c\_word \cap pre\_word| \times |all\_file| \times |post\_query|.$$

显然,$|pre\_word - c\_word| + |c\_word \cap pre\_word| + |c\_word - pre\_word| = |post\_word|$ 且 $|C\_file| \le |all\_file|$,因此,

$$UpdateCost_{supernode} < |post\_word| \times |all\_file| \times |post\_query|.$$

随着查询数目的增多,*post_query*,*post_word* 和 *all_file* 都可能逐渐变大.对于 *post_query* 和 *post_word*,可采用 LRU 策略淘汰历史记录中不常用的查询用词和扩展用词;对于 *all_file*,可规定最大的文档数目 N_{file},当超过这个数目时,淘汰文档用词使用频率低的文档.另外,目录节点的更新可离线计算,并设定更新周期,这可以极大地降低更新的代价.

8 实验及分析

本节通过一系列实验来验证查询扩展及其路由算法的有效性.目前,没有标准的文档测试集用于混合 P2P 环境下查询扩展的测试.因此,我们单独开发了一个文档测试集,约为 2.5G 大小,文档来源于 NEUSOFT 等计算机类网站,将 HTML 标题作为文档的名字,并删除文档中诸如 a,the,is 等出现频率极高的词.因为网络用户提交的查询多为短查询,因此,我们从文档名中随机提取关键字来构造查询测试集,测试集包含的查询多为 2~3 个单词的

短查询,目录节点间的拓扑结构基于 Power-Law[14]构造.测试时,随机选择叶节点发送查询,并假定该叶节点不包含查询对应的文档.这里,采用精度-查全率(precision-recall)指标评价查询性能,查询精度和查全率定义为

$$Precision=\frac{\sum_{qualified} Answer}{\sum_{retrieved} Answer},\qquad Recall=\frac{\sum_{retrieved} Answer}{\sum_{system\ \ avaliable} Answer} \tag{10}$$

采用标准的向量空间模型 VSM(vector space model)[15]作为叶节点的检索算法,同时实现了文献[11]中的搜索算法(history_based search,简称 HS)作为本文算法的比较.表 1 总结了实验参数,表 2 为查询扩展的测试集.

Table 1 Parameters of experiments

表 1 实验参数

Parameter	Default value	Description
Network topology	Power-Law	The topology of network
Network size	2 550	150 directory nodes and 2 400 leaf nodes
Queries number	8 000	The number of queries
Query size	2~3	Average terms number for original queries
TTL	5	The time-to-live of a query message
Expansion terms	30	The number of expansion terms for a query
Selected documents	15	The number of selected documents for LEM

Table 2 Queries for test

表 2 测试查询列表

Number	Query
1	Storage training
2	Quality management
3	Content management
4	Solution
5	E-Learning business
6	Network security
7	Education forum
8	Call center
9	Human resource management
10	Digital medicine

8.1 实验1

该实验的目的是检验 LEM 和 HEM 查询扩展算法能否提高现有 HS 搜索算法的性能.首先执行前 4 000 个查询积累查询记录.随机选择叶节点发送查询,中间路由的节点依据查询与邻居节点的相似度选择 K 个节点转发查询;随后,将扩展后的查询应用于 HS 算法,以观察查询性能的变化.比较的结果如表 3(TTL=5,K=3,*query number*=4000)及图 5 所示.

Table 3 Comparison of query efficiency

表 3 查询性能比较

Precision / Recall	HS	LEM_based HS (improvement %)	HEM_based HS (improvement %)
10	55.56	66.67 (+20.00)	71.42 (+28.55)
20	45.45	54.05 (+18.92)	58.82 (+29.42)
30	41.66	48.38 (+16.13)	50.84 (+22.04)
40	33.61	39.22 (+16.69)	41.67 (+23.98)
50	26.45	35.46 (+34.06)	35.21 (+33.12)
60	20.00	29.85 (+49.25)	28.30 (+41.50)
70	15.52	19.50 (+25.64)	20.95 (+34.99)
80	12.28	15.78 (+28.50)	17.05 (+38.84)
90	9.47	12.89 (+36.11)	14.78 (+56.07)
100	7.40	10.91 (+47.43)	12.33 (+66.62)
Total averaged	26.74	33.27 (+29.27)	35.14 (+31.41)

从表 3 可以看出:在测试数据集上,查询扩展后的 HS 算法确实获得了比较好的结果,相对于基本的 HS 算法,基于 HEM 方法扩展的 HS 算法在精度方面提高百分比最大可以达到 66.62%,平均为 31.41%;而基于 LEM 方法扩展的 HS 算法则在精度方面平均提高了 29.27%.当 7 000 个查询执行完毕时,查询性能的比较如图 6 所示.显然,随着历史记录的积累,查询精度随之会有所提高,查询扩展的优势也越加明显.这主要是由于历史记录的积累,同样也意味着用户关于查询用词和扩展用词间相关性判断的积累,因而扩展更加有效.

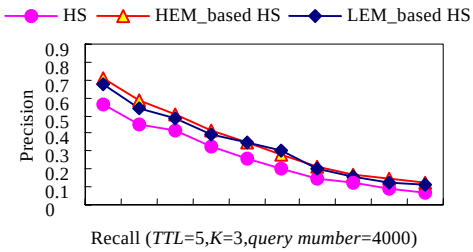

Fig.5 Comparison of query efficiency

图 5 查询性能比较

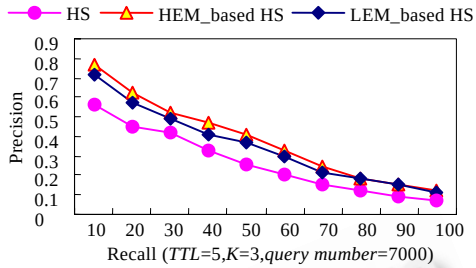

Fig.6 Comparison of query efficiency

图 6 查询性能比较

8.2 实验2

该实验对现有 HS 算法的查询结果、基于 HEM 方法扩展的 SE 的结果和基于 LEM 方法扩展的 SE 的结果进行了比较(见表 4 和图 7(*TTL*=5,*K*=3)).实验表明:与 HS 算法相比,基于 HEM 方法扩展的 SE 算法在精度方面提高百分比最大可以达到 67.03%,平均为 42.60%.即使与基于 HEM 方法扩展的 HS 算法相比,我们的算法同样获得 8.51%的提高;基于 LEM 方法扩展的 SE 算法在精度方面提高的百分比平均为 41.58%,在与基于 LEM 方法扩展的 HS 算法的比较中,算法也获得了 13.80%的提高.在 SE 算法中,除查询文本信息以外,用户点击下载的文档信息也被作为查询路由的指引,因此算法获得了较好的查询效果.

Table 4 Comparison of query efficiency

表 4 查询性能比较

Precision / Recall	HS	LEM_Based SE (improvement %)	HEM_Based SE (improvement %)
10	55.56	83.34 (+50.00)	76.92 (+38.44)
20	45.45	60.61 (+33.36)	64.52 (+41.95)
30	41.66	55.56 (+33.37)	57.69 (+38.48)
40	33.61	43.47 (+29.34)	44.45 (+32.25)
50	26.45	36.49 (+37.96)	38.76 (+46.54)
60	20.00	29.85 (+49.25)	31.58 (+57.90)
70	15.52	21.27 (+37.05)	22.44 (+44.59)
80	12.28	17.51 (+42.59)	17.66 (+43.81)
90	9.47	15.17 (+60.19)	15.02 (+58.61)
100	7.40	12.33 (+66.62)	12.36 (+67.03)
Total averaged	26.74	37.86 (+41.58)	38.13 (+42.60)

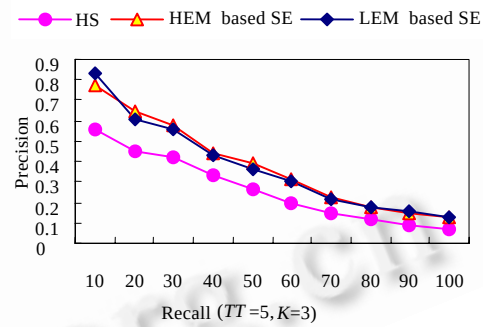

Fig.7 Comparison of query efficiency

图 7 查询性能比较

另外,由上述实验可知:HEM 方法的查询效果略优于 LEM 方法,主要是因为 LEM 方法仅从与查询最相关的 *N* 篇文档中选择查询扩展词,这样可能导致部分查询用词与文档用词关联信息的损失.而 HEM 方法则从全部的历史查询记录中选择扩展词,因而相对于 LEM 方法涵盖有更多的关联信息.然而,由第 7.4 节可知:HEM 方法的更新代价最大为|*post_word*|×|*all_file*|×|*post_query*|,与文档用词的数目成正比.而 LEM 方法的更新主要涉及到 *N* 篇最相关文档的重新选择,更新代价增加缓慢,且通常情况下,比 HEM 方法的代价要小.

8.3 实验3

该实验的目的是检验增大 *K* 对查询效果的影响.HS 算法和 SE 算法搜索的范围受到选择节点个数(*K*)的影响,增大 *K* 可以提高查询的效果.限于篇幅,在此仅列出 HS 算法和基于 HEM 方法扩展的 SE 算法在不同 *K* 值下的查询结果.基于 LEM 方法的 SE 算法和 HS 算法的比较结果与此类似.实验结果表明:增大 *K* 能够提高查询精度,在 *K*=4 的情况下(见表 5 和图 8(*TTL*=5,*K*=4)),SE 算法较 HS 算法在精度方面最大百分比可达 38.05%,平均为 17.37%;在 *K*=5 的情况下(见表 6 和图 9(*TTL*=5,*K*=5)),SE 算法较 HS 算法在精度方面最大百分比可达 47.58%,平均为 18.61%.图 10 表示了在不同 *K* 值下,SE 算法和 HS 算法平均精度的变化.显然,SE 算法的表现明显优于 HS 算法.值得一提的是,增大 *TTL* 能够取得与增大 *K* 类似的效果.虽然增大 *K* 或 *TTL* 能够提高查询的效果,但也会影响查询的效率,因为这意味着搜索将访问更多的节点.因此,一般情况下,*K* 取值为 3,*TTL* 为 5.

Table 5 Comparison of query efficiency (k=4)

表 5 查询性能比较(k=4)

Recall \ Precision	HS	HEM_based SE (improvement %)
10	71.42	83.33 (+16.68)
20	68.96	76.92 (+11.63)
30	65.21	71.42 (+9.52)
40	55.55	66.67 (+20.02)
50	39.37	54.35 (+38.05)
Total averaged	60.10	70.54 (+17.37)

Table 6 Comparison of query efficiency (k=5)

表 6 查询性能比较(k=5)

Recall \ Precision	HS	HEM_based SE (improvement %)
10	75.00	90.91 (+21.21)
20	71.42	80.00 (+12.01)
30	68.18	75.00 (+10.00)
40	62.50	67.79 (+8.47)
50	41.32	60.98 (+47.58)
Total averaged	63.68	74.94 (+18.61)

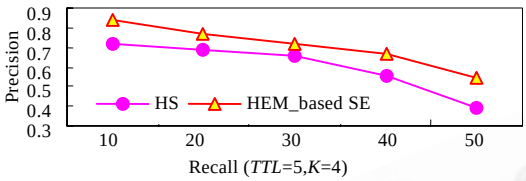

Fig.8 Comparison of query efficiency

图 8 查询性能比较

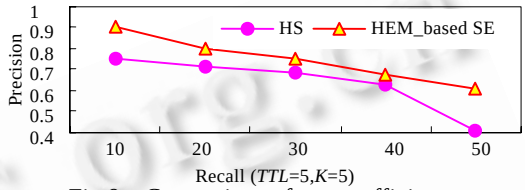

Fig.9 Comparison of query efficiency

图 9 查询性能比较

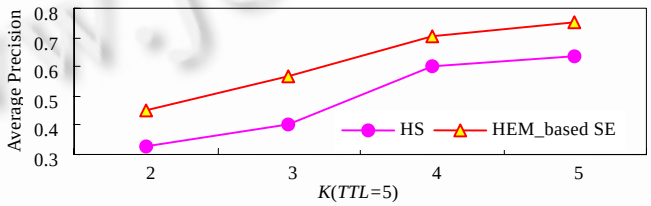

Fig.10 Change of average precision

图 10 不同 K 值下平均精度的变化

8.4 实验4

实验的目的是评价不同算法的搜索效率.取查询访问的节点数(如图 11 所示)和传递的查询消息数(如图 12 所示)作为衡量搜索效率的指标.访问的节点越少,传递的消息数就越少,搜索的效率就越高,付出的代价就越小.

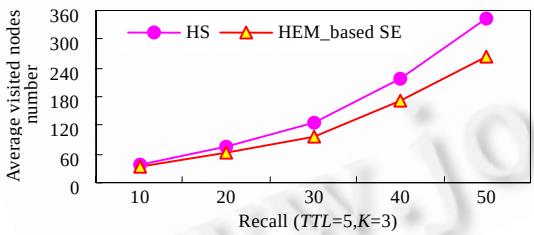

Fig.11 Comparison of average visited nodes number

图 11 平均访问节点数目的比较

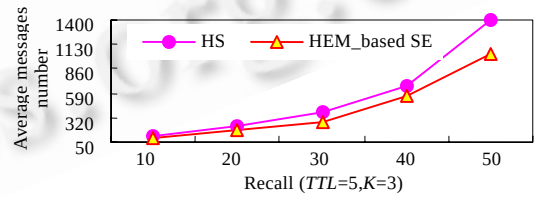

Fig.12 Comparison of average message number

图 12 平均发送消息数目的比较

从图 11 和图 12 容易看出:获取同等数目的相关结果,SE 算法需要访问的节点数和发送消息的数目都明显低于 HS 算法;相关结果获取的数目越多,SE 的优势也越加明显.这主要是因为用户提交的短查询无法提供检索出相关文档的足够信息,而查询扩展弥补了这一缺陷.基于 LEM 的 SE 算法与 HS 算法的比较结果类似,在此不再详述.

9 结 论

本文讨论了如何在混合 P2P 环境下进行有效的查询扩展,并在此基础上提出了一种新的混合 P2P 环境下的搜索算法.分析和仿真表明,查询扩展及其搜索算法能够提高搜索的效果.如何聚集相同领域的叶节点,使得目录节点上的查询扩展更具针对性,是我们下一步的研究目标.

References:

[1] Zhang Q, Sun Y, Liu Z, Zhang X, Wen XZ. Design of a P2P-based grid content management architecture. In: Iiow J, ed. Proc. of the 3rd Communication Networks and Services Research Conf. New York: IEEE Press, 2005. 339−344.

[2] Clarke I, Sandberg O, Wiley B, Hong TW. Freenet: A distributed anonymous information storage and retrieval system. In: Federrath H, ed. Proc. of the Workshop on Design Issues in Anonymity and Unobservability. Berlin: Springer-Verlag, 2001. 46−66.

[3] Furnas GW, Landauer TK, Gomez LM, Dumais ST. The vocabulary problem in human-system communication. Communications of the ACM, 1987,30(1):964−971.

[4] Jansen BJ, Spink A, Saracevic T. Real life, real users, and real needs: A study and analysis of user queries on the web. Information Processing and Management, 2000,36(2):207−227.

[5] Deerwestr S, Dumai ST, Furnas GW, Landauer TK, Harshman R. Indexing by latent semantic analysis. Journal of the American Society for Information Science, 1990,41(6):391−407.

[6] Salton G, Buckley C. Improving retrieval performance by relevance feedback. Journal of the American Society for Information Science, 1990,41(4):288−297.

[7] Buckley C, Salton G, Allan J, Singhal A. Automatic query expansion using SMART: TREC3. In: Harman DK, ed. Proc. of the 3rd Text Retrieval Conf. Gaithersburg: NIST Press, 1994. 69−80.

[8] Xu JX, Croft WB. Improving the effectiveness of information retrieval with local context analysis. ACM Trans. on Information Systems, 2000,18(1):79−112.

[9] Callan JP, Lu ZH, Croft WB. Searching distributed collections with inference networks. In: Fox EA, Ingwersen P, Fidel R, eds. Proc. of the 18th Annual Int'l ACM SIGIR Conf. on Research and Development in Information Retrieval. New York: ACM Press, 1995. 21−28.

[10] Cui H, Wen JR, Nie JY, Ma WY. Query expansion by mining user logs. IEEE Trans. on Knowledge and Data Engineering, 2003,15(4):1−11.

[11] Lu J, Callan J. Content-Based retrieval in hybrid peer-to-peer networks. In: Frieder O, Hammer J, *et al*. eds. Proc. of the 12th Int'l Conf. on Information and Knowledge Management. New York: ACM Press, 2003. 199−206.

[12] The gnutella protocol specification v0.6. 2005. http://rfc-gnutella.sourceforge.net

[13] He YJ, Feng YL, Wang S. Intelligent search based on content of documents in peer-to-peer network. Journal of Computer Research And Development, 2004,41(Suppl):112−118 (in Chinese with English abstract).

[14] Palmer CR, Steffan JG. Generating network topologies that obey power laws. In: Gavalas D, Greenwood D, *et al*. eds. Proc. of the IEEE Global Telecommunication Conf. (GLOBECOM 2000). San Francisco: IEEE Press, 2000. 434−438.

[15] Wang ZW, Wong SKM, Yao YY. An analysis of vector space models based on computational geometry. In: Belkin NJ, Ingwersen P, *et al*. eds. Proc. of the 15th Annual Int'l ACM SIGIR Conf. on Research and Development in Information Retrieval. New York: ACM Press, 1992. 152−160.

附中文参考文献:

[13] 何盈捷,冯月利,王珊.Peer-to-Peer 环境下基于内容的智能搜索.计算机研究与发展.2004,41(增刊):112−118.

张骞(1979－),男,山东金乡人,博士生,主要研究领域为数据管理与 P2P 计算.

孙雨(1975－),男,博士生,主要研究领域为数据管理与 P2P 计算.

张霞(1965－),女,博士,教授,CCF 高级会员,主要研究领域为 P2P 数据管理,数据库技术.

文学志(1970－),男,博士生,主要研究领域为网络安全.

刘积仁(1955−),男,博士,博士生导师,主要研究领域为计算机网络技术.

刘铮(1979－),女,硕士,主要研究领域为数据管理.