P2P 环境下数据管理系统上的 Top-k 查询*)

何盈捷　文继军　冯月利　王　珊
(中国人民大学信息学院　北京 100872)

摘　要　目前大多数 P2P 系统只提供文件的共享，缺乏数据管理能力。基于关系数据库上的关键搜索，本文提出了一种在 P2P 环境下共享数据库的新框架，其中每个节点上的数据库被看成是一个文档集，用户不用考虑数据库的模式结构信息，简化了不同节点数据库模式间的映射过程，能更好地适应 P2P 的分散和动态特性。将基于直方图的分层 Top-k 查询算法扩展到 P2P 环境下的数据库管理系统上，文档集和数据库的查询被统一起来，一致对待。在查询处理期间，直方图可以自动更新，同时根据查询结果，邻居节点可以自调整，具有自适应性。实验结果表明，基于关键词的数据库共享突破了传统的数据库共享模式，简化了数据访问方式，而基于直方图的 Top-k 查询算法提高了查询效率。

关键词　P2P，数据库共享，关键词查询，Top-k 查询，直方图，邻居节点自调整

Top-k Query over Data Management System in P2P Network

HE Ying-Jie　WEN Ji-Jun　FENG Yue-Li　WANG Shan
(Information School, Renmin University of China, Beijing 100872)

Abstract　Most of existing peer-to-peer (P2P) systems only provide coarsely granular file-level sharing and lack of data management ability. A new framework of database sharing in P2P network is presented, which is based on keyword search over relational databases. In the framework, database on each peer can be seen as a text collection, users needn't care about the schema information of the database, which greatly simplifies the schema mapping between different peers, and is more suitable for the decentralization and dynamicity of P2P network. Hierarchical top-k query processing algorithm based on histogram is applied in the data management system in P2P network to do top-k query, which unifies the query over text collections and relational databases. During the query processing, histogram can be updated automatically by using the returned top-k results, and the neighborhood of the peer can be self reconfigured to let the nodes containing the real top-k results to be its neighbors. Experiment results show that database sharing based on the keyword searching breaks through the traditional database sharing schema and simplifies the data accessing method, and top-k query based on histogram improves the search efficiency.

Keywords　Peer-to-peer, Database sharing, Keyword query, Top-k query, Histogram, Neighborhood self reconfiguration

1　引言

目前的 P2P 系统大多只提供文件级的共享，共享粒度较粗，缺乏数据管理的能力，将 P2P 技术和数据库技术相结合是大势所趋[1]。由于 P2P 系统是一个分散、动态的系统，系统中不存在中心控制节点，节点可以随意加入和退出网络，使得原来分布式数据库的一些技术不再适应 P2P 系统，需要新的技术来适应 P2P 的挑战。

Pizza[2]和 PeerDB[3]是两个具有代表性的 P2P 环境下的数据管理系统。Pizza 通过分散的模式映射在一定程度上解决了 P2P 环境下的数据库共享问题，但它要求事先给出节点之间的模式映射关系，随着系统规模的扩大，维护工作量很大；PeerDB 利用信息检索的技术实现不同节点之间的模式映射。节点数据管理系统，需要为每个关系表名和属性名定义一组关键词，不同节点之间的模式映射通过关键词的匹配进行。PeerDB 在一定程度上解决了 P2P 环境下的数据库共享。但 PeerDB 采用的方法是一种不精确的方法，仍然需要用户的干预。

本文提出了一种在 P2P 环境下共享数据库的新框架：基于关键词查询的数据库共享。将每个节点上的数据库看成是一个文档集，用户不用考虑数据库的模式结构信息，简化了不同节点数据库模式间的映射过程，能够较好地适应 P2P 分散和动态的特性。我们将 Top-k 查询扩展到 P2P 环境下的数据管理系统上，将文档集和数据库的查询统一起来，一致对待。实验证明，文档集上基于直方图的分层 Top-k 查询算法[4]同样适用于 P2P 环境下数据库上的 Top-k 查询。随着查询的进行，直方图可以自动更新，同时根据查询的结果对邻居节点进行调整，把那些真正包含 Top-k 结果的节点作为查询发起节点的邻居，具有自适应性。基于关键词的数据库共享突破了传统的数据库共享模式，简化了数据访问方式，而基

*)本文得到国家自然科学基金项目(60473069)，国家自然科学基金重大项目(60496325)，北京市科技计划重点项目(H030130060011)和 863 专项(2003AA4Z3030)的支持。何盈捷　博士研究生，主要研究领域为数据库，信息检索。文继军　博士研究生，主要研究领域为数据库，信息检索。冯月利　硕士研究生，主要研究领域为数据库。王　珊　博士生导师，主要研究领域为数据库与知识工程，网格数据管理与信息检索等。

于直方图的 Top-k 查询算法提高了查询的效率。

2 P2P 环境下的数据管理系统

2.1 SEEKER 简介

文本文档和网页的关键词检索已经是成熟的研究领域。近年来，结构化和半结构化数据的关键词查询开始引起研究人员的兴趣。已经有几个关系数据库关键词查询系统开发出来：BANKS[5]，DBXplorer[6]，DISCOVER[7]。这些系统的基本思路是一致的，即将数据库看作是由数据库中的元组（顶点）通过主码-外码关系（边）连接而成的图。当用户给出一个关键词查询，通过全文索引从图中找出含全部关键词的最小子图作为查询结果。

我们实现了一个基于关键词的关系数据库信息检索系统 SEEKER[8]，与现有的关系数据库关键词查询系统相比，它不仅可以检索关系数据库里的文本属性（CHAR，VARCHAR 等），还可以检索数据库的元数据（关系名、属性名等）以及数字属性（DATE，TIME，INT，FLOAT，NUMERIC 等）；并且，SEEKER 采用了更合理的评分公式对查询结果排序，将 Top-k 结果返回给用户。

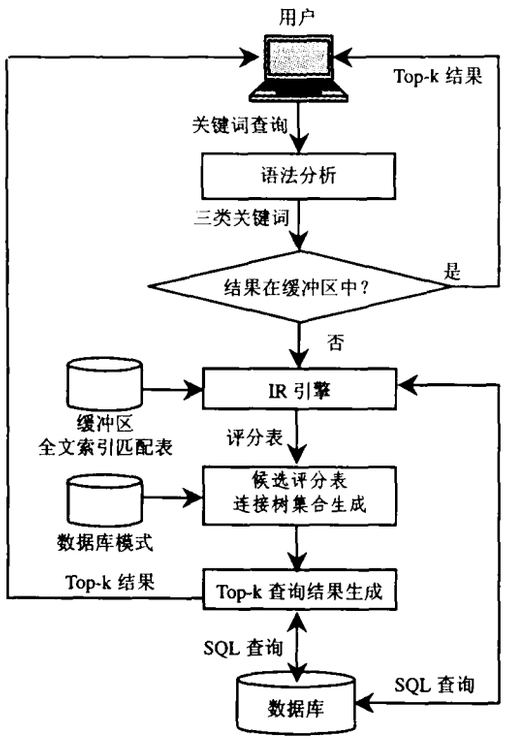


图 1 SEEKER 的体系结构

SEEKER 的体系结构如图 1 所示。SEEKER 的查询结果是以数据库中的元组为结点的一棵树：连接元组树。树中任意两个相邻的元组间的边是主码-外码连接关系，连接元组树包含至少一个查询中的关键词，且每个叶结点都包含至少一个查询中的关键词。

2.2 P2P 环境下基于关键词查询的数据库共享

我们提出了一种 P2P 环境下共享关系数据库的新框架：基于关键词查询的数据库共享。当用户提交一个关键词查询，搜索 P2P 网络中与关键词最为匹配的 Top-k 个结果返回给用户。P2P 系统中每个共享数据库的节点都提供一个关键词查询的接口，不同节点返回的关键词查询结果合并生成最终的结果。图 2 给出了 P2P 环境下数据管理系统的体系结构。系统共分为四层：P2P 层、关键词检索层、Top-k 查询层和数据库管理层。数据库管理层采用 RDBMS（Oralce、DB2 或者 SQL Server）来管理数据；关键词检索层在数据库管理层之上，提供本地数据库上关键词的查询，我们用 SEEKER 作为关键词检索层；Top-k 查询层主要负责将各个节点（包括本地节点）的查询结果汇集生成全局的 Top-k 结果；P2P 层主要负责节点之间的网络通讯。

基于关键词查询的关系数据库共享，实质上是将 P2P 网络中的每一个数据库都看成是一个文档集，用户使用关键词查询来获取分布在 P2P 系统中各处的数据库中的数据。采用这种框架，用户在对 P2P 网络中的共享关系数据库进行信息检索时，不需要任何 SQL 语言和底层数据库模式的知识，不需要知道数据库位于何处，不需要不同数据库间的模式映射过程，非常好地适应了 P2P 系统分散和动态的特性。

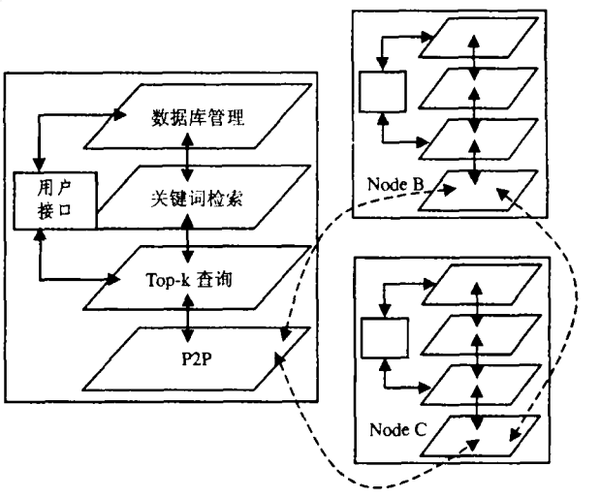


图 2 P2P 环境下数据管理系统的体系结构

3 P2P 环境下数据库上的 Top-k 查询

我们首先介绍本地数据库上基于关键词的 Top-k 查询，然后应用基于直方图的分层 Top-k 查询算法进行 P2P 环境下的数据管理系统上的 Top-k 查询。

3.1 本地数据库上基于关键词的 Top-k 查询

本地数据库的 Top-k 查询由 SEEKER 来完成，下面是 SEEKER 的一些基本定义。

一个包含 n 个关系 R_1，…，R_n 的数据库 DB，可以将它看成一个有向图 G。DB 中的每一个关系就是 G 的一个顶点，每一个主码-外码关系就是 G 的一条有向边，方向由主码所在的关系 $R_i \in G$ 指向外码所在的关系 $R_j \in G$，记为 $e(R_i, R_j)$，将 G 称为模式图（Schema Graph）。一个关键词查询 Q 由一组关键词 kw_1，…，kw_m 组成，表示为 $Q(kw_1, \cdots, kw_m)$。

定义 1 连接元组树 TT 是以数据库 DB 中的元组为结点的一棵树，树中任意两个相邻的元组 $t_i \in R_i$，$t_j \in R_j$，它们之间的边 $e(t_i, t_j) = e(R_i, R_j) \in G$ 且 $(t_i \bowtie t_j) \in (R_i \bowtie R_j)$。TT 的大小是它拥有的元组的数量，记为 sizeof(TT)。

定义 2 查询结果是含至少一个 Q 中的关键词且每个叶结点都含至少一个 Q 中的关键词的连接元组树。

SEEKER 只将 Top-k 查询结果返回给用户，因此需要给查询结果评分，然后根据分数高低来对查询结果排序。查询结果所含关键词的数量不尽相同，显然含关键词越多，得分应

该越高。SEEKER 采用了以下的评分公式：

$$\text{Score}(TT,Q)=\frac{1}{\text{sizeof}(TT)}\left(\frac{m'}{m}\right)^{a}\sum_{i=1}^{\text{sizeof}(TT)}\sum_{j=1}^{m}\text{Score}(T_i,kw_j) \quad (1)$$

其中，TT 是构成查询结果的一个连接元组树；sizeof(TT)是 TT 中所含元组的个数，它与该结果的得分成反比；T_i 是 TT 中的元组；Q 是一个关键词查询，m 是 Q 中关键词的个数，$kw_j \in Q$，m' 是 TT 中所含关键词的数量；a 是常数，取大值(例如 10)来保证含关键词多的查询结果比含关键词少的查询结果的得分高；$\text{Score}(T_i, kw_j)$是元组 T_i 对于关键词 kw_j 所得的分数。

3.2 P2P 环境下数据管理系统上的 Top-k 查询

我们应用基于直方图的分层 Top-k 查询算法进行 P2P 环境下的数据管理系统上的 Top-k 查询。给定一组关键词，搜索网络中与查询最为匹配的 Top-k 个元组连接树。本文只考虑针文本属性的关键词查询，暂不考虑元数据和数值属性的关键词查询。

3.2.1 *分层的 Top-k 查询* 纯的 P2P 网络的搜索是一种广播式的搜索，节点除了进行本地搜索，还将查询被广播发送给它的所有邻居节点。搜索范围由查询跳数 TTL 给出。查询每转发一次，TTL 减 1。当 TTL 等于 0 时，就停止传播查询。为了防止回路的产生，每个查询消息都有一个唯一的编号。如果节点先前已经收到查询，则说明发生了回路，节点不再继续传播查询。我们可以把这一查询过程用一棵查询树来表示，如图 3 所示。图中节点 a 发起查询，为查询树的根节点。查询被广播发送给节点的所有邻居节点，每次转发 TTL 减 1，TTL 为 0 或者没有邻居的节点为叶子节点。

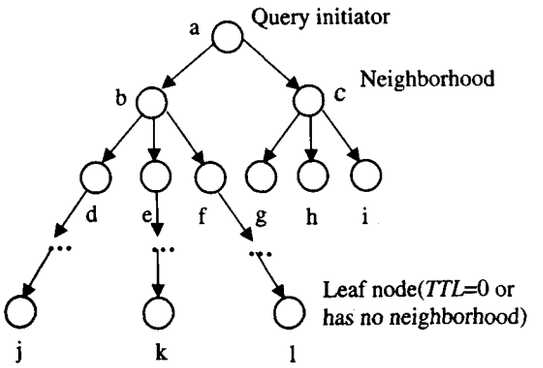


图 3 查询树

我们定义 P2P 环境下数据管理系统上的 Top-k 函数为：

$$\text{Top-k}(P,Q,TTL)=\max_k(\{\text{Local-Top-k}(P_i,Q)\mid P_i\in\text{QueryTree}(P,Q,TTL)\}) \quad (2)$$

其中，Local-Top-k(P_i, Q)为节点 P_i 的本地 Top-k 函数，采用 SEEKER 的评分公式(式 1)作为本地的 Top-k 函数。$\max_k$ 为全局的 Top-k 函数，比较查询树中所有节点的 Top-k 结果，取其中分数最大的 k 个元组连接树作为最终的 Top-k 结果返回。SEEKER 需要使用 RDBMS 给出的分数(即式(1)中 $\text{Score}(T_i, kw_j)$)，而不同的 RDBMS 采用的评分方法可能是不一样的。本文假设网络中所有的节点都采用一样的 RDBMS。如果存在不同的 RDBMS，可以考虑加一个中间层来统一它们给出的分数。

基于查询树，我们分层计算 Top-k 函数，将结果的排序和合并分布到网络中的各个节点上，实现分布式的 Top-k 查询。父亲节点除了进行本地的 Top-k 查询，还要聚集儿子节点返回的 Top-k 结果，产生最优的 Top-k 结果。叶子节点(TTL 为 0 或者没有邻居的节点)仅仅进行本地的 Top-k 查询，将 Top-k 结果返回给它的父亲节点。整个查询过程自底向上逐层进行，直到根节点得到最终的 Top-k 结果。例如，图 3 中，节点 j 查询本地信息，返回本地的 Top-k 结果给父亲节点。节点 b 聚集节点 d，e，f 返回的 Top-k 结果以及本地的 Top-k 结果，产生最优的 Top-k 结果，继续返回给节点 a。节点 a 聚集节点 b 和节点 c 返回的 Top-k 结果以及本地的 Top-k 结果，产生最终的 Top-k 结果，返回给用户。

3.2.2 *直方图的建立* 广播式的搜索是低效的，我们采用直方图的方法提高搜索的效率。根据 Peer(节点)返回的 Top-k 结果为 Peer 构建直方图，直方图中存储的是查询关键词的分数上限。利用直方图为将来的查询估计 Peer 可能的分数上限，即查询树中以 Peer 为根的子树所包含的元组连接树的分数上限。对 Peer 进行选择，裁剪查询树中的一些无用分枝(不含真正的 Top-k 结果)，以提高搜索的效率。

观察 SEEKER 的评分公式(式(1))，我们可以在计算 Score(TT, Q)的过程中为每个查询关键词 kw_j 计算一个分数，计算公式如下：

$$\text{Score}(TT,kw_j)=\frac{1}{\text{size}(TT)}\sum_{i=1}^{\text{size}(TT)}\text{Score}(T_i,kw_j) \quad (3)$$

$$\text{Score}(TT,Q)=\left(\frac{m'}{m}\right)^{a}\sum_{j=1}^{m}\text{Score}(TT,kw_j) \quad (4)$$

$\text{Score}(T_i, kw_j)$是 RDBMS 的全文索引给出的一个分数。例如，Oracle9i 给出的是一个位于区间[0，100]的值。对 $\text{Score}(T_i, kw_j)$进行归一化，除以系统的最大分数，将 $\text{Score}(T_i,kw_j)$映射到区间[0，1]。由式(3)，$\text{Score}(T_i, kw_j)$的取值范围一定在区间[0，1]内。

性质 1 SEEKER 的评分公式(式(4))单调递增的，即给定两个元组连接树 TT_1 和 TT_2，假如对查询 Q 包含的所有 kw_j，都有 $\text{Score}(TT_1, kw_j) \geq \text{Score}(TT_2, kw_j)$，则 $\text{Score}(TT_1,Q) \geq \text{Score}(TT_2,Q)$。

假设 Peer 返回的 Top-k 结果不仅包含元组连接树的分数，而且包含查询中每个关键词的分数。根据性质 1，我们可以利用 Peer 返回的 Top-k 查询结果构建 Peer 的直方图。由于 P2P 环境下的查询与查询跳数相关，因此我们为 Peer 建立不同跳数的直方图。设查询 Q 在 Peer A 上的跳数为 TTL，Peer A 返回的与查询 Q 最相似的 k 个元组连接树为{TT_1，TT_2…，TT_k}，$\text{Score}(TT_i, kw_j)$表示元组连接树 TT_i 对关键词 kw_j 的分数，$\text{Score}(TT_i, kw_j)$已经归一化。对 Q 中的每个关键词 kw_j，我们计算 kw_j 在 Top-k 结果中的分数上限，即 $\text{Score}_{kw_j,A,TTL}=\max_{1\leq i\leq k}\text{Score}(TT_i,kw_j)$，构建 Peer A 跳数为 TTL 的直方图为：

$$\text{Histogram}_{A,TTL}=\{(kw_j,\text{Score}_{kw_j,A,TTL}=\max_{1\leq i\leq k}\text{Score}(TT_i,kw_j))\mid kw_j\in Q\} \quad (5)$$

计算 Peer A 相对于查询 Q 的分数上限：

$$\text{UpperScore}_{Q,A,TTL}=\sum_{j=1}^{m}\text{Score}_{kw_j,A,TTL}$$

由性质 1，可以保证

$\text{UpperSocre}_{Q,A,TTL} \geq \text{Score}(TT_i, Q)$，$1 \leq i \leq k$。我们用 $\text{UpperScore}_{Q,A,TTL}$估计以 Peer A 为根的子树 QueryTree(A，Q，TTL)所包含的元组连接树的分数上限。

对将来的查询 Q′(假设查询 Q′在 Peer A 上的跳数为 TTL′)，我们可以利用直方图估计 Peer A 的分数上限，采用

的计算公式为：

$$UpperScore_{Q',A,TTL'} = \left(\frac{m''}{m}\right)^{a} \sum_{kw_j \in Q' \wedge kw_j \in Histogram_{A,TTL} \wedge TTL = \min(TTL_1 | TTL_1 \geq TTL')} Score_{kw_j,A,TTL} + \sum_{kw_j \in Q' \wedge \neg \exists Histogram_{A,TTL}(kw_j \in Histogram_{A,TTL} \wedge TTL \geq TTL')} 1 \quad (6)$$

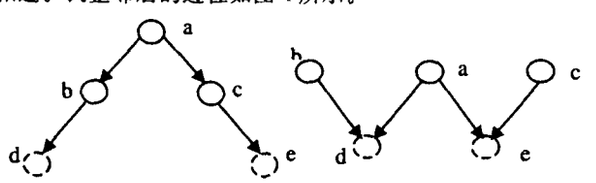
式中，m 表示查询 Q' 包含的关键词个数，m'' 表示 Peer A 跳数大于等于 TTL' 的直方图中包含 Q' 的关键词个数。对 Q' 中的每个关键词 kw_j，假如 Peer A 的直方图中存在跳数大于、等于 TTL' 的直方图并且包含 kw_j，我们取跳数与 TTL' 最接近的直方图中的分数上限 $Score_{j,A,TTL}$ 计算 Peer A 的分数上限；否则，我们取最大分数 1 计算 Peer A 的分数上限。由于经过归一化的 $Score(T_i, kw_j)$ 取值范围在 0 到 1 之间，取最大分数 1 能够保证估计的分数上限一定大于实际的分数上限。

利用直方图估计了 Peer 可能的分数上限，我们可以根据 Peer 的分数上限对 Peer 进行选择。首先对 Peer 进行排序，依次选取分数最好的 Peer 发送查询。假如当前返回的 Top-k 结果的最小分数大于等于余下 Peer 的分数上限，说明余下的 Peer 不包含真正的 Top-k 结果。停止继续发送查询，得到最终的 Top-k 结果。

根据查询的 Top-k 结果，我们可以对直方图进行更新，计算关键词在 Top-k 结果中的分数上限，和直方图中的分数上限进行比较，保存大的分数上限。这样的更新方法可能导致累计效应的产生，因此我们需要定期刷新直方图。

3.2.3 *邻居节点自调整* 利用直方图对 Peer 进行选择，删除了查询树中一些无用的分枝，提高了搜索效率。但只对搜索路径进行了选择，并没有对查询路径进行优化。假如真正的 Top-k 结果在查询树的叶子节点，我们仍然需要遍历从根节点到叶子节点查询路径上的所有节点。为了更快地找到真正含有 Top-k 结果的目标节点，我们需要根据查询结果对查询路径进行优化，这就需要节点能够动态地调整它的邻居节点。

假如返回的 Top-k 结果不光包含结果的分数，还包含结果的源节点（来自哪个节点），则我们可以根据 Top-k 结果调整节点的邻居，使得节点与真正包含 Top-k 结果的节点直接相连。调整邻居的过程如图 4 所示。



(a) 配置前，节点 d, e 包含真正的 Top-k 结果　　(b) 配置后，节点 d, e 成为节点 a 的邻居节点

图 4　根据 Top-k 结果配置邻居节点

图 4 中，节点 d，e 包含真正的 Top-k 结果。配置前，节点 a 要取得节点 d，e 的结果，需要首先访问节点 b，c。根据返回的 Top-k 结果重新配置邻居后，节点 d，e 成为节点 a 的邻居节点，节点 b，c 不再直接与节点 a 相连。这样，查询 q 可以直接发送到目的节点 d，e，减少了访问节点 b，c 的过程。邻居节点的重配置，一方面缩短了查询路径，减少了访问节点的数目，提高了查询的效率；另一方面，由于 P2P 网络中的查询都有一个跳数的限制，调整邻居节点进一步扩展了搜索的范围，原来需要 n 次转发才可以找到节点，现在只需要 1 步就能找到。可以搜索到网络中先前没有访问到的节点，得到更多、更好的 Top-k 结果，提高了查询的精度。

由于对不同的查询，返回 Top-k 结果的节点是不一样的，很可能配置后的邻居节点对查询 q_1 为最优，而对查询 q_2 却很差。因此，我们需要根据查询的频率进行邻居节点的自配置。对经常使用的查询，节点根据查询返回的 Top-k 结果重新配置邻居节点，提高今后该查询的效率；对不经常使用的查询，节点不做邻居节点的重配置。

4　实验

本节我们通过模拟实验验证基于直方图的分层 Top-k 查询算法是有效的。我们对算法的查询效果和查询效率进行实验分析。

4.1　实验设置

所有实验都在一台 PC 机上完成，PC 机的配置为 CPU P4 1.6GHz，内存 1GB，操作系统 Windows XP。模拟程序由 JAVA 编写，网络拓扑结构基于 Power-law，由 PLOD 算法[9] 产生。

表 1　实验参数

参数名称	缺省值	描述
Network Topology	Power-law	网络拓扑结构，每个节点的平均出度为 4
Network Size	1000	网络中的节点数
TTL	5	Time-to-live，查询消息的跳数
K	10	返回的元组连接树的数目

注意：PLOD 算法产生的网络是一个无向图，每个节点平均出度为 4，相当于无向图中有 2000 条无向边。

Top-k 查询的测试集我们采用 DBLP 数据集。我们将 DBLP[10] 网站提供的 XML 文件分解成四个关系表：Author 有 294,062 个元组；Paper 有 446,409 个元组；Write 有 1,000,099 个元组；Cite 有 111,357 个元组。DBLP 的数据库模式图如图 5 所示。

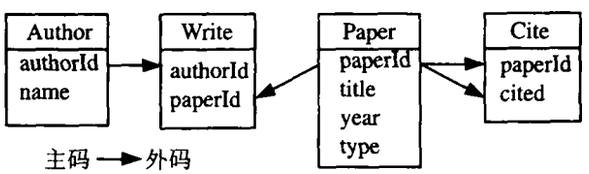


图 5　DBLP 数据库模式图

我们根据 Paper 关系表对数据库进行划分。首先，按照文章的 title（标题）属性，将 Paper 划分为 26 份，title 分别以 'A'（或'a'）到'Z'（或'z'）开头；然后，根据 Paper 对其他三个表（Author，Write 和 Cite）进行划分，将与 Paper 相关的 Author，Write 和 Cite（根据主码-外码关系）放在一个数据库分片中。划分好的 26 个数据库分片随机地分布在网络中的 26 个节点上。

我们构造了 20 组关键词查询对数据库进行测试，查询由论文作者以及他们的研究领域组成，比如："E. F. Codd, database"，"Jim gray, transaction"，"Donald E. Knuth, algorithm"等。

4.2　实验效果

查询效果，我们采用查准率（Precision）进行衡量。将集

中式环境下数据库上的关键词查询结果 R 为测试的基准，比较 P2P 环境下的 Top-k 查询结果与集中式环境下的 Top-k 查询结果，计算查准率。查准率的计算公式如下：

$$Precision = |Ra| / K \quad (7)$$

其中，Ra 表示返回的结果中在 R 中的结果数，K 为返回的结果数。

实验 1 基于直方图的分层 Top-k 查询的查询效果

执行 20 组关键词查询，每次查询从网络中任选一个节点进行，最后计算平均查准率。图 6 给出了 1000 个节点 P2P 网络在返回 Top 10 个结果的情况下的平均查准率。

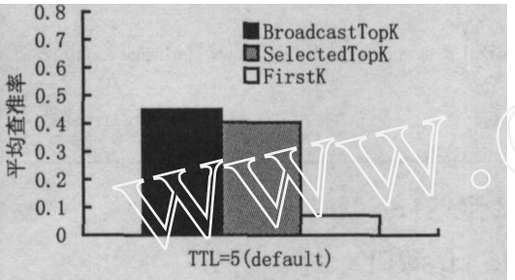


图 6 Top-k 查询的查询效果

图 6 中，我们比较了 BroadcastTopK（基于广播搜索的分层 Top-k 查询算法）、SelectedTopK（基于直方图的分层 Top-k 查询算法）和 FirstK（取到 K 个与查询相关的结果就停止搜索）的查询效果。可以看出，Top-k 查询的效果远远好于 FirstK 的查询效果，说明 Top-k 查询提高了查询的精度。另一方面，BroadcastTopK 的查询效果与 SelectedTopK 的查询效果十分接近，说明基于直方图的 Peer 选择是有效的，基本不会影响 Top-k 查询的查询效果。

实验 2 不同 TTL 情况下的查询效果

由 3.2.1 节的讨论，我们知道纯 P2P 环境下的 Top-k 查询，搜索范围受到 TTL 的限制，得到的 Top-k 结果是一个局部最优的 Top-k 结果，增大 TTL，可以进一步提高查准率。实验结果如图 7 所示。当 TTL=8 时，平均访问了 419 个节点，平均查准率为 0.553。查准率不是很高，一方面由于漏掉了一些节点，另一方面由于数据的划分损失了一些信息，得到的结果可能与集中式的结果不一样。

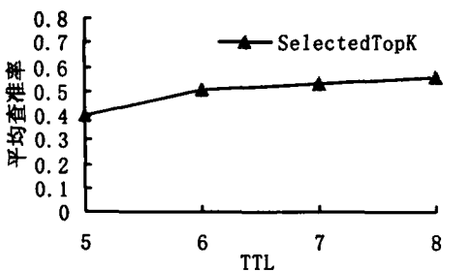


图 7 不同 TTL 情况下的查询效果

4.3 查询效率

我们采用查询访问的节点数作为衡量查询效率的指标。

实验 3 基于直方图的分层 Top-k 查询的效率

执行 20 组关键词查询，每次查询从网络中任选一个节点进行，最后计算平均访问的节点数。结果如图 8 所示。图中，我们比较了 BroadcastTopK 和 SelectedTopK 所访问的节点数。在执行 SelectedTopK 之前，我们首先为每个查询建立直方图。可以看出，基于直方图的 Peer 选择有效地裁剪了搜索空间，减少了节点的访问数目，大大提高了查询的效率。

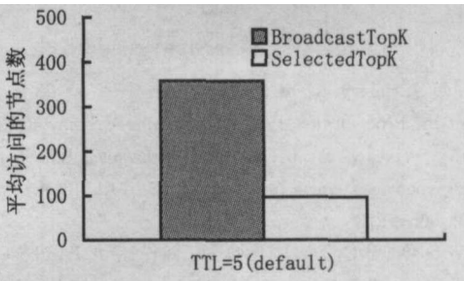


图 8 Top-k 查询平均访问的节点数

实验 4 邻居节点自调整

我们通过调整邻居节点进一步提高查询效率和查询效果，查询路径上的每个节点根据返回的 Top-k 结果动态调整自己的邻居。针对查询 3，实验结果如图 9 所示。

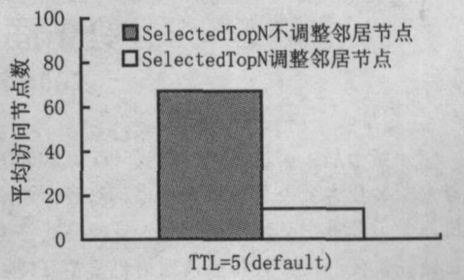


图 9 邻居节点自调整的查询效率

我们根据查询 3 的结果对邻居节点进行调整，把那些包含真正 Top-k 结果的节点作为查询发起节点的邻居。在网络中任意选择 10 个节点执行查询 3，计算平均访问的节点数。可以看出，根据查询结果调整邻居节点能够提高查询的效率。由 3.2.3 节的讨论，调整邻居节点还可以进一步扩展查询的搜索范围，使得原来在跳数 TTL 范围内不能访问到的节点现在可以访问到，提高了查询的精度。

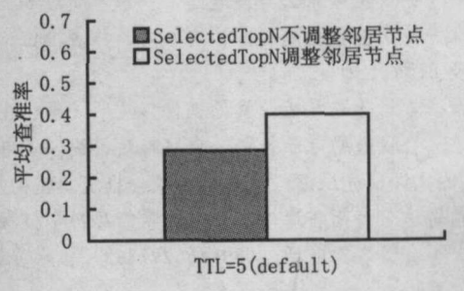


图 10 邻居节点自调整的查准率

图 10 给出了调整邻居节点前后平均查准率的比较（针对查询 3）。可以看出，根据查询结果调整邻居节点提高了查询的效果。

结论 P2P 环境下的数据库共享是 P2P 目前研究的热点。数据库上的关键词检索标志着数据库与 IR 的结合，用户查询数据库不再需要关心数据库的模式结构，只需要输入一组关键词，就能获得与查询相关的元组连接树。基于关系数据库上的关键词检索，我们提出了一种 P2P 环境下共享数据库的新框架，大大简化了不同节点之间的数据库模式映射。接着，我们将 Top-k 查询扩展到 P2P 环境下的数据管理系统上，将文档集和数据库的查询统一起来，提供统一的查询接口。下一步，我们将把 XML、多媒体等数据的查询统一起来，

提供统一的接口。

参 考 文 献

1 Gribble S, Halevy A, et al. What can database do for Peer-to-Peer? In: Proc. of the 4th WebDB. Santa Barbara, 2001. 31～36

2 Halevy A, Ives Z, Suciu D, et al. Schema mediation in peer data management systems. In: Proc. of the 19th ICDE, Bangalore, 2003. 505～518

3 Ng W S, Ooi B C, Tan K L, et al. PeerDB: A P2P-based system for distributed data sharing. In: Proc. of the 19th ICDE, Bangalore, 2003. 633～644

4 何盈捷，王珊. 纯 Peer to Peer 环境下有效的 Top-k 查询. 软件学报，2005，16(4)：540～552

5 Halotia G, Hulgeri A, Nakhey C, et al. Keyword searching and browsing in databases using BANKS. In: Proc. of the 18th ICDE, San Jose, 2002. 431～440

6 Agrawal S, Chaudhuri S, Das G. DBXplorer: a system for keyword-based search over relational databases. In: Proc. of the 18th ICDE, San Jose, 2002. 5～16

7 Hristidis V, Papakonstantinou Y. DISCOVER: keyword search in relational databases. In: Proc. of the 28th VLDB, Hong Kong, 2002. 670～681

8 文继军，王珊. SEEKER：基于关键词的关系数据库信息检索. 软件学报，2005(已录用)

9 Palmer C, Steffan J. Generating network topologies that obey power law. In: Proc. of GLOBECOM, San Francisco, 2000. 434～438

10 DBLP Home Page. http://www.informatik.uni-trier.de/～ley/db/

第十届中国机器学习会议

2006 年 10 月 13-15 日，海口

第十届中国机器学习会议（CCML2006）由中国人工智能学会机器学习专业委员会和中国计算机学会模式识别与人工智能专业委员会联合主办，海南大学承办，海南软件学院协办。该系列会议每两年举行一次，现已成为国内机器学习界最主要的学术活动。此次会议将为机器学习及相关研究领域的学者交流最新研究成果、进行广泛的学术讨论提供便利，并且将邀请国内机器学习领域的著名学者做精彩报告。

征稿范围（征求但不限于如下主题）

机器学习的新理论、新技术与新应用；人类学习的计算模型；计算学习理论；监督学习；非监督学习；强化学习；多示例学习；半监督学习；集成学习；多策略学习；基于案例的推理；增量学习与在线学习；对复杂结构数据的学习；增强学习系统可理解性；数据挖掘与知识发现；神经网络；神经网络集成；进化计算；人工生命；模糊集与粗糙集；多 Agent 系统中的学习；模式识别；信息检索；生物信息学；语音、图像处理与理解；自然语言解。

投稿要求

·论文必须未公开发表过，一般不超过 6000 字；中、英文稿均可接受

·论文应包括题目、作者姓名、作者单位、摘要、关键字、正文和参考文献；另附作者地址、邮编、电话或传真及 E-mail 地址

·参选优秀学生论文的稿件请注明（须由在校博士生、硕士生或本科生）为第一作者

·会议鼓励电子投稿，也可邮政投稿：若电子投稿，请将 Word 格式的文件发到：ccml06@hainu.edu.cn（超过 1M 的文件请先压缩；请注意接收会议组织机构发出的收稿确认电子邮件）。若邮政投稿，请将三份打印稿于截稿日期前寄达：海南省海口市海南大学信息科学技术学院 曾水香 收 邮编：570228

·会议咨询电话：0898-66272862（曾水香），66288382（雷景生）

论文出版

所有录用论文按评审结果分别发表在《Journal of Computational Information Systems》（英文稿件，全部 EI 检索）、《计算机研究与发展》（EI 检索源）、《计算机科学》（中文核心期刊）、《计算机工程》（EI 检索源）和《广西师范大学学报》（中文核心期刊），以上论文发表期刊均为正刊；会议还将评出 3 篇优秀学生论文，颁发证书并给予奖励。

重要日期

全文投稿：2006 年 3 月 15 日　　录用通知：2006 年 5 月 15 日　　修改定稿：2006 年 6 月 15 日