ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.18, No.6, June 2007, pp.1443−1455
DOI: 10.1360/jos181443

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

非结构化对等计算系统中多维范围搜索*

徐林昊, 钱卫宁+, 周傲英

(复旦大学 计算机科学与工程系,上海 200433)

Multi-Dimensional Range Search in Unstructured Peer-to-Peer Systems

XU Lin-Hao, QIAN Wei-Ning+, ZHOU Ao-Ying

(Department of Computer Science and Technology, Fudan University, Shanghai 200433, China)

+ Corresponding author: Phn: +86-21-65643921, Fax: +86-21-65643503, E-mail: wnqian@fudan.edu.cn

Xu LH, Qian WN, Zhou AY. Multi-Dimensional range search in unstructured peer-to-peer systems. *Journal of Software*, 2007,18(6):1443−1455. http://www.jos.org.cn/1000-9825/18/1443.htm

Abstract: It is an important problem to efficiently support similarity search for multi-dimensional data spaces in peer-to-peer (P2P) data management environment. Current unstructured P2P data sharing systems provide only a very rudimentary facility in query processing, i.e., matching-based query processing. This paper therefore presents a simple, yet effective index structure called EVARI (extended vector approximation routing index) to address the problem of multi-dimensional range search in unstructured P2P systems, by means of both data approximation and routing index techniques. With the aid of the EVARI, each peer can not only process range queries with its local dataset, but also route queries to promising peers with the desired data objects. In the proposed scheme, each peer summarizes its local content using space-partitioning technique, and exchanges the summarized information with neighboring peers to construct the EVARI. Furthermore, each peer can reconfigure its neighboring peers to keep the relevant peers nearby so as to optimize system resource configuration and improve system performance. Extensive experiments show the good performance of the proposed approach.
Key words: peer-to-peer computing; range search; vector approximation; routing index; query routing

摘 要: 对等计算数据管理中的一个重要问题是如何有效地支持多维数据空间上的相似性搜索.现有的非结构化对等计算数据共享系统仅支持简单的查询处理方法,即匹配查询处理.将近似技术和路由索引结合在一起,设计了一种简单、有效的索引结构 EVARI(扩展近似向量路由索引).利用 EVARI,每个节点不仅可以在本地共享的数据集上处理范围查询,而且还可以将查询转发给最有希望获得查询结果的邻居节点.为了建立EVARI,每个节点使用空间划分技术概括本地的共享内容,并与邻居节点交换概要信息.而且,每个节点都可以重新配置自己的邻居节点,使得相关节点位置相互邻近,优化了系统资源配置,提升了系统性能.仿真实验证明了该方法的良好性能.
关键词: 对等计算;范围搜索;近似向量;路由索引;查询路由

* Supported by the National Natural Science Foundation of China under Grant Nos.60496325, 60496327 (国家自然科学基金); the High Education Doctorial Program of MoE in China under Grant No.20030246023 (国家教育部博士点基金); the Science and Technology Commission of Shanghai Municipal Government of China under Grant No.03DZ15028 (上海市科委重大项目); the National University of Singapore and Info-Communications Development Authority of Singapore under a Grant on Peer-to-Peer Computing Research (新加坡资讯通信发展管理局基金)

Received 2005-07-21; Accepted 2006-08-16

中图法分类号: TP311 文献标识码: A

对等计算(peer-to-peer computing,简称 P2P)已经成为工业界和学术界的研究热点.目前,众多研究工作集中在 P2P 环境下的数据管理[1,2]上,而 P2P 环境下的查询处理问题显得尤为重要.早期的 P2P 系统[3]仅支持匹配查询,查询处理效率低下.为了改善查询处理性能,文献[4–6]提出了一些基于启发式规则的查询路由策略和路由索引技术.虽然这些方法提升了 P2P 系统的查询处理效率,但无法实现基于语义或内容的复杂查询处理.针对这个问题,来自数据库领域的研究人员探索了如何在 P2P 环境下实现关系查询处理和多维范围搜索.在关系查询处理方面,文献[7,8]研究了节点之间的语义映射关系,实现了基于查询重写的关系查询处理,文献[9,10]通过扩展分布式散列表(distributed hash table,简写 DHT),分别实现了基于 DHT 的连接查询处理和基于缓存的查询处理,而 Ng 等人则研究了基于代理技术的 P2P 查询处理[11].

另一个研究热点是如何在 P2P 系统上实现多维相似搜索.在 P2P 数据共享系统中,每个节点共享的数据(如文档、音乐、图像等)一般被抽象为特征向量,而特征向量之间的距离被用于衡量数据之间的相似性.在传统数据库领域中,针对多维数据模型,研究人员设计出了多维索引[12],旨在解决特征向量空间中的相似搜索问题.因此,基于多维索引的相似搜索技术提供了一种基于内容的(语义的)查询处理方法,被广泛应用于地理信息系统和多媒体信息系统.近年来,学术界已经提出了几种 P2P 环境下的相似搜索方法.例如,SCRAP 和 MUCK[13]分别在结构化 P2P 系统 Chord[14]和 CAN[15]上实现了多维范围搜索;SkipIndex[16]将基于 *kd*-树[12]的空间划分方法和 skip graph[17]结合在一起,提出了高维最近邻搜索算法.本质上,这些方法都是考虑如何对数据空间进行划分,在保持数据局部性的前提下,在数据子空间与节点之间建立起映射关系.不同于结构化 P2P 环境下的范围搜索,文献[18]提出了一种分层的索引结构,解决了非结构化 P2P 环境下的 *k*-最近邻搜索问题.

在上述相关工作中,与本文工作最接近的是文献[18],但两者主要存在以下三方面的不同:

首先,本文提出的方法适用于完全分布式的非结构化 P2P 系统;而文献[18]中提出的方法仅适用于由超级节点和客户节点组成的 P2P 系统.因此,文献[18]提出的方法无法应用于完全分布式的非结构化 P2P 系统.

其次,虽然两者都采用了基于 VA-File[19]的空间划分,但在本文中,每个节点仅建立反映局部数据分布的索引;而文献[18]则采用了分层索引.具体而言,每个客户节点负责建立本地索引,将本地索引发送给对应的超级节点;每个超级节点负责将其客户节点的本地索引合并为反映其客户节点上数据分布的局部索引;最后,每个超级节点都将局部索引广播给所有其他超级节点,建立全局索引以反映整个系统的数据分布.因此,这种分层索引结构与本文提出的动态的、自适应的分布式索引结构是完全不同的.

最后,在本文中,每个节点上的索引是随着网络的改变而动态调整的,索引的存储代价和维护代价都不大.而在文献[18]中,所有的超级节点和客户节点需要定期地更新全局索引、局部索引和本地索引,这意味着该方法所需的索引存储代价和维护代价较大.

从资源定位角度看,结构化 P2P 系统[14,15,17,20,21]的路由代价具有理论保证,但在 Internet 上最流行且被广泛使用的却是非结构化 P2P 系统[3,22],这是因为:P2P 系统中的用户具有瞬时性,可以随意地加入或者退出系统;其二,与匹配查询处理相比,基于语义的范围搜索的应用更为广泛且重要;第三,用户查询往往是针对“最流行的”数据,而不是“稀有的”数据[3].因此,本文将研究非结构化 P2P 系统中的多维范围搜索问题.

1) 将传统多维索引技术与 P2P 系统相结合,提出了一种新颖的路由索引结构,给出了多维范围搜索算法,以及索引的建立与维护算法.
2) 针对非结构化 P2P 网络的动态性特征,提出了一种基于反馈的网络自配置算法,优化了系统的资源配置,进一步提升了系统性能.
3) 给出了索引结构的维护代价、查询处理的代价模型以及确定空间划分位数的算法.
4) 在仿真环境下,对本文提出的方法进行了大量的实验测试,证明了这种索引结构非常适合于非结构化 P2P 环境下的范围搜索.

本文第 1 节给出问题描述.第 2 节提出索引结构,给出范围搜索和网络自配置算法.第 3 节讨论索引的建立

和维护算法.第 4 节提出代价模型.第 5 节分析实验结果.最后总结全文.

1 问题描述

一个非结构化 P2P 网络是由大量自治节点相互连接而成.假设所有节点共享的数据对象都属于同一个维度的空间,且每个数据对象都表示为特征向量.特征向量之间的距离决定了数据对象之间的相似性[12,19].本文将研究非结构化 P2P 系统中的多维范围搜索.

定义. 设节点上的数据集 DB 属于维度为 d 的单位超立方体 $\Omega=[0,1]^d$,采用欧几里德距离衡量任意两个特征向量之间相似性,则范围查询 Q 的结果是一个特征向量集合,满足条件 $range(Q)=\{o\in DB|dist(o,c(Q))\leq r(Q)\}$,其中,$dist$ 表示欧几里德距离函数,o 表示特征向量,$c(Q)$表示 Q 的中心点,$r(Q)$表示 Q 的搜索半径.

在一个动态的 P2P 系统中,每个节点都不可能掌握整个系统的数据分布情况,而仅知道哪些节点是自己的邻居.因此,在非结构化 P2P 系统中实现范围搜索的主要难点是:不仅要让每个节点能够处理范围查询,而且还要让每个节点“掌握”局部的数据分布情况,有效地路由查询,减少范围搜索访问的节点数.

问题. 在一个非结构化 P2P 网络中,当任意一个节点发出范围查询后,如何尽可能地提高范围搜索的查全率和查准率,且同时尽可能地降低查询访问的节点数.

为了解决上述问题,可以从两个方面考虑:(1) 在非结构化 P2P 网络中,每个节点都不可能拥有全局的数据分布信息.为了将查询路由到可能包含查询结果的节点,每个节点需要通过邻居节点与其他节点交换一些必要的统计信息,建立起一种局部的数据分布信息.这样,第 1 个问题是每个节点都需要收集并维护什么类型的统计信息;(2) 由于每个节点共享的是多维数据,因此每个节点必须建立并维护一种多维索引,用于处理范围查询.这样,第 2 个问题就是什么类型的多维索引适用于 P2P 环境下的范围搜索.

2 基于 EVARI 的范围搜索

本节首先介绍近似向量技术.然后给出一种基于近似向量的分布式路由索引结构,以及如何利用这种索引实现 P2P 环境下的范围搜索.最后,探讨如何利用网络自配置特性,进一步提升系统性能.

2.1 近似向量技术

近似向量(vector approximation)技术[19]是一种空间划分方法,将多维空间分割为 2^b 个单元格,其中 b 是一个系统参数.具体地,多维空间的第 i 个维度被赋予了一个整数 b_i,用于将该维度划分为间距相等的 2^{b_i} 个区间.所有维度上的划分位数 b_i 的总和等于 b.这样,每个单元格就由一个长度为 b 的位串表示,所有落在该单元格中的数据都由相同的位串表示.例如,图 1 所示的 2 维空间被分割成 16 个单元格,d_5 和 d_7 表示为 0111.这样,一个多维数据集就表示为一个近似向量的集合.在本文的剩余部分,单元格与近似向量具有相同含义,可相互替换.

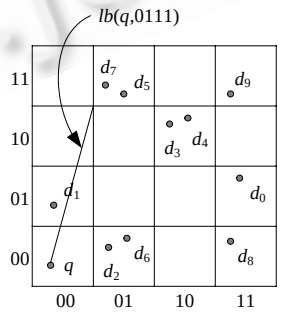

Fig.1 Space partitioning and vector approximation

图 1 空间划分与近似向量

与现有的多维索引结构相比,近似向量技术非常适合非结构化 P2P 系统,这是因为:第一,基于近似向量的范围搜索方法受“维度灾难(curse of dimensionality)”[19]的影响很小;第二,近似向量技术的计算有效性很好,尤其适

合 P2P 系统的动态性特征.即只需在索引中插入或者删除近似向量,而不会招致额外的计算开销.因此,近似向量技术被用于解决第 1 个问题,即近似向量技术不仅被用于生成每个节点的本地内容概要,而且还被用于处理范围查询[19].为了解决第 2 个问题,即将查询路由到包含结果的其他节点,仅使用近似向量是不够的.为此,下一节将设计 EVARI 索引,为每个节点建立起一种局部的数据分布信息,实现有效的查询路由.

2.2 EVARI:一种基于近似向量的路由索引

扩展近似向量路由索引(extended vector approximation routing index,简称 EVARI)是近似向量技术在 P2P 环境下的一种自然扩展.EVARI 不仅利用近似向量来概括节点上的共享数据,实现多维范围搜索,而且还能描述 P2P 系统中的局部数据分布信息,有效地路由相似查询.

EVARI 是一种倒排索引,以近似向量作为每个索引项的键.每个索引项是一个二元组⟨*cell_id*,*peer_list*⟩,其中,*cell_id* 表示近似向量,是一个由“0”和“1”组成的长度为 *b* 的位串,*peer_list* 是一个链表,每个链表项记录了一些邻居节点的标识(如 IP 地址).这样,每个索引项就提供了范围查询所需的路由依据.即给定一个包含结果的近似向量,通过搜索 EVARI,当前节点就可以确定应该向哪些邻居节点发送范围查询.*peer_list* 中的每个链表项是一个二元组⟨*peer_id*,*soi*⟩,其中,*peer_id* 是节点标识,*soi*(scope of indexing)是一个整数,用于限制该链表项对应的近似向量在 P2P 网络中的传播范围.显然,如果对节点之间信息交换的范围不加以限制,那么每个节点用于维护 EVARI 的存储和计算开销会变得过大,同时也会消耗过多的网络带宽资源.因此,需要限定索引范围的最大值 soi_{max},以控制“知识”的传播范围,降低为建立和维护 EVARI 而引起的存储和计算开销.

例如在图 2 中,假设节点的加入次序为 $P_1 \rightarrow P_3 \rightarrow P_2$,且 $soi_{max}=2$.图 3 展示了节点 P_1 建立 EVARI 的过程(算法见第 4 节).当 P_3 加入网络后,它将和 P_1 交换近似向量.P_1 的 EVARI 增加了两个索引项 1101 和 0111,且它们的 *peer_list* 中的 *soi* 值等于 1.这是因为,P_1 从 P_3 获得这些信息只经过了 1 次跳转,即从 P_3 到 P_1.当 P_2 加入系统后,通过 P_3,P_1 的 EVARI 又增加了两个索引项 0100 和 1111,且这两个索引项的 *peer_list* 中的 *soi*=2.这是因为,P_1 从 P_3 获得这些信息经过了 2 次跳转,即从 P_2 到 P_3,再从 P_3 到 P_1.

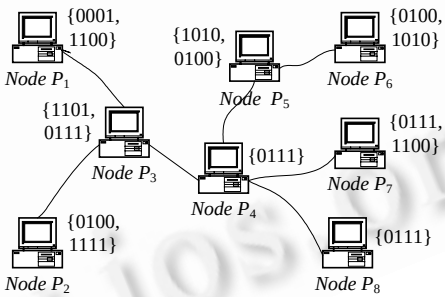

Fig.2 Peers with their vector approximations

图 2 节点与它们共享的近似向量

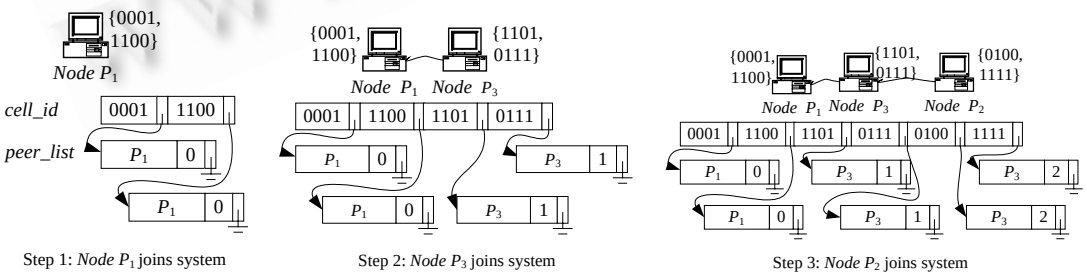

Fig.3 The process of constructing P_1's EVARI

图 3 节点 P_1 的 EVARI 的建立过程

图 4 给出了节点 P_5 的 EVARI(令 $soi_{max}=2$).P_5 的 EVARI 中仅包含 P_4 和 P_6.这是因为 P_5 只知道它有两个邻

居节点 P_4 和 P_6(如图 2 所示),而并不知道其他节点的存在.也就是说,虽然 P_3 拥有近似向量 1101,但 P_5 只能从 P_4 上获得该信息.注意到,在信息交换过程中,信息每经过一次跳转,其 *peer_list* 表项中对应的 *soi* 的值都将被更新.如果某个信息的 $soi=soi_{\max}$,则该信息将不会被发送给其他节点.此外,交换近似向量可以保证路由索引的增长速度小于指数函数的增长速度.这是因为,属于不同节点且属于同一个单元格的数据会被同一个索引项“吸收”.例如,P_5 的 EVARI 包含了 5 个索引项,而这 5 个索引项却包含了 10 个属于不同节点的数据.

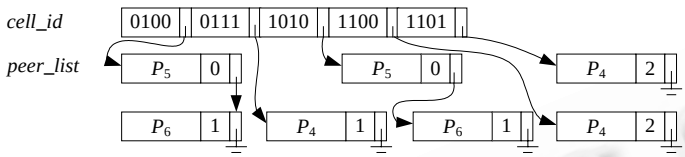

Fig.4 P_5's EVARI

图 4 节点 P_5 的 EVARI

除了继承近似向量本身的优点以外,EVARI 的另一个重要特性在于:由于每个节点不可能掌握全局的数据分布信息,因此通过节点之间的信息交换,每个节点都逐步掌握了自己周围的(局部的)数据分布情况.这样,每个节点就可以依据这种局部的数据分布信息,将查询逐步地路由到拥有查询结果的节点,而不是盲目地选择邻居节点发送查询请求或者在网络中对查询请求进行广播.

2.3 范围搜索算法

假设节点 P 接收到范围查询 Q,则节点 P 使用 EVARI 处理 Q 的过程如下:

1) 节点 P 顺序扫描 EVARI 中的每个单元格 c,计算 c 与查询中心点 q 之间的下界 $lb(q,c)$.如果 $lb(q,c)$小于或等于搜索半径 $r(Q)$,则扫描该索引项的 *peer_list*.如果 $peer\_id=P$,则将 c 放入到候选单元格集合 C 中(P 包含属于该单元格的数据);否则,将 *peer_id* 放入到候选邻居节点集合 N 中.
2) 对于候选单元格集合 C 中的每个单元格 c,节点 P 顺序扫描该单元格所包含的本地数据.如果数据 d 与 q 之间的距离小于或等于 $r(Q)$,即 $dist(d,q)\le r(Q)$,则将 d 放入到查询结果集合 R 中.
3) 节点 P 对范围查询的生命周期加 1.如果候选邻居节点集合 N 不为空,且范围查询的生命周期小于阈值 $TTL_{\max}$,则节点 P 将 Q 路由给集合 N 中的所有邻居节点,否则终止范围搜索.
4) 如果查询结果集合 R 不为空,则节点 P 直接将查询结果 R 返回给查询请求节点.

在范围搜索的执行过程中,所有接收到查询的节点都将执行上述操作.

2.4 网络自配置

非结构化 P2P 系统的另一个优点在于网络自配置.为了充分利用系统资源,每个节点自由选择其他节点作为邻居,以期望在尽可能短的时间内获取尽可能多的查询结果.在理想情况下,每个节点都可以与其他所有节点建立邻居关系.然而在实际的 P2P 系统中,这是不可能的:首先,网络连接需要消耗节点的网络带宽,而节点的网络带宽有限;其次,整个网络将会被用户查询所淹没.这是因为,当任意一个节点发出查询后,所有节点都将接收到该查询.因此,每个节点只能与有限数量的节点建立邻居关系.换句话说,在网络自配置时,如果一个节点与另一个节点建立了邻居关系,则它必须从原有的邻居节点中选择一个,并与之断绝邻居关系.

假设节点 P 发出范围查询,则节点 P 采用网络自配置策略选择邻居节点的算法分为两个步骤.第 1 步,节点 P 依据被访问的节点返回的结果数量,对这些节点进行排序.返回结果数量较多的节点等级较高,而返回结果数量较少的节点等级较低.这里采用了一个合理的假设:如果节点 v 能够从或者通过节点 u 获取最多的查询结果,对于节点 v 将来的查询而言,节点 u 仍旧很有可能为节点 v 提供较多的查询结果.这样,依据反馈调整节点之间的邻居关系,最有希望包含查询结果的节点将会被最先访问到.第 2 步,等级最高的 k 个节点将作为查询请求节点的 k 个新邻居节点,与节点 P 建立连接,其中,k 是一个系统参数,由每个节点依据实际的网络带宽情况而定.节点 P 从原有的邻居节点中选择 k 个节点,并与它们断绝邻居关系.

当一个节点根据反馈与一些最有可能提供较多查询结果的节点建立邻居关系之后,所有这些节点都需要更新自己的 EVARI.这显然增加了 EVARI 的维护代价.然而实验结果将验证,由网络自配置所引起的 EVARI 的维护代价,将会因系统性能的提升而得到补偿.

3 索引的建立与维护

3.1 索引的建立

当一个节点加入 P2P 系统时,它将与邻居节点交换近似向量,建立 EVARI.交换的信息格式是一个三元组⟨*cell_id*,*peer_id*,*soi*⟩,其中,*cell_id* 表示近似向量,*peer_id* 是节点标识,*soi* 表示该信息已经传播的范围.之所以采用这种格式,是因为 EVARI 的作用是处理并路由查询,并不直接提供单元格的实际位置信息.这一方面减少了索引的信息量,同时还保持了非结构化 P2P 系统本身的特性.因此,每个节点只记录从哪些邻居节点获取了哪些近似向量.假设节点 P 加入 P2P 网络,邻居节点为 $P_1,...,P_n$,则 P 建立 EVARI 的过程如下:

1) 节点 P 初始化 EVARI,向邻居节点 $P_1,...,P_n$ 发出获取近似向量的请求,等待邻居节点的回复.
2) 一旦接收到从某个邻居节点 P_i 发送过来的信息,P 立即将近似向量插入到 EVARI 中.
   a) 对于某个单元格 *cell_id*,如果 EVARI 中已经有了该索引项,则检查在该索引项的 *peer_list* 中是否包含 P_i.如果没有,则创建链表项⟨P_i,*soi*+1⟩并插入到 *peer_list* 中;否则,更新链表项⟨P_i,*soi′*⟩的 *soi′*为 min(*soi′*,*soi*+1),其中 *soi′*和 *soi* 分别指已有链表项和新得到的信息的索引范围;
   b) 如果 EVARI 中没有索引项 *cell_id*,则节点 P 创建索引项⟨*cell_id*,*peer_list*⟩,将该索引项插入到 EVARI 中.最后将链表项⟨P_i,*soi*+1⟩插入到 *peer_list* 中.
3) 对于每个索引项,P 检查 *peer_list* 中所有链表项的 *soi* 值,确定哪些单元格可以传播给邻居节点.
   c) 对于某个索引项,如果其 *peer_list* 中所有链表项的 $soi=soi_{max}$,则忽略该近似向量;
   d) 对于某个邻居节点 P_i,如果其标识出现在某个索引项的链表中,则不将该索引项的近似向量传播给 P_i;否则,将所有有效的近似向量⟨*cell_id*,*P*,*soi*⟩发送给邻居节点 $P_1,...,P_n$.

当一个节点加入系统时,它首先从邻居节点获取它们的近似向量,而并不是先将自己的近似向量发送给邻居节点.这是因为一些节点需要通过一个新加入系统的节点来交换彼此的近似向量,以更新各自的 EVARI.此外,通过检查 *soi* 值可以避免因节点构成的环路而引起的无限制的消息循环传递问题.由于节点 P 的加入,所有邻居节点都需要更新各自的 EVARI.下面将讨论节点对 EVARI 的维护算法.

3.2 索引的维护

为了准确地路由和处理查询,每个节点都必须保持本地 EVARI 的正确性,以反映 P2P 系统的动态特征.当节点失效或退出系统以及节点上的数据发生变化时,相关节点需要更新 EVARI.

3.2.1 节点失效或退出

节点退出与节点失效之间的区别在于:对于前者,在一个节点退出系统之前,该节点将通知它的邻居节点“我即将离开系统”;对于后者,由于网络连接不稳定或硬件故障等因素而导致节点断线,已经离开 P2P 网络的节点的邻居节点并不知道该节点已经不在系统中了.在上述两种情况中,虽然节点离开系统的方式不同,但当这些事件发生之后,相关节点执行的 EVARI 更新操作是类似的.下面首先介绍节点退出系统时的 EVARI 更新算法.假设节点 P 退出系统,邻居节点为 $P_1,...,P_n$,那么邻居节点更新各自 EVARI 的过程如下:

1) 节点 P 将“我即将离开系统”这一消息主动通知所有邻居节点 $P_1,...,P_n$,然后离开 P2P 网络.
2) 当某个邻居节点 P_i 收到节点 P 的通知之后,它将从每个索引项的 *peer_list* 中删除包含 P 的链表项.如果所有索引项的 *peer_list* 都不为空,则结束更新操作.这是因为,虽然 P 的离开使得 P_i 失去了一些单元格信息(失去了属于该单元格的一些数据对象),但 P_i 的其他邻居节点上仍然拥有这些单元格.
3) 如果某个索引项 *cell_id* 的 *peer_list* 为空,则 P_i 从 EVARI 中删除该索引项,并创建一个新的 EVARI 更新消息⟨*cell_id*,P_i,*soi*⟩(*soi* 的初始值为 1).如果 $soi<soi_{max}$,则 P_i 将该消息发送给邻居节点.接收到该消息的节点将执行第 2 步操作,更新本地和相关节点的 EVARI;否则,结束更新操作.

更新算法的合理性在于,如果节点 P 的离开使得节点 P_i 认为系统中已经“不存在”某些单元格信息了,那么 P_i 不仅需要从本地的 EVARI 中删除这些单元格对应的索引项,而且还需要通知邻居节点更新它们的 EVARI.对于节点失效情况,更新算法中的第 1 步将被替换为:节点 $P_1,...,P_n$ 定期地探测邻居节点 P 是否已经离开了系统.如果 P 在一段给定的时间内没有响应,则认为 P 已经退出系统,并执行第 2 步和第 3 步操作,更新 EVARI.

3.2.2 数据变化

由于节点上的数据变化可能导致单元格的增加或删除,因此,节点需要对 EVARI 执行相应的更新操作.对于单元格的增加,当节点 P_i 从邻居节点 P 那里得到了所需更新的索引项 *cell_id* 之后,节点 P 执行 EVARI 建立算法中的第 2 步和第 3 步;对于单元格的删除,节点 P_i 执行 EVARI 更新算法中的第 3 步即可.

4 代价模型

4.1 索引维护代价模型

首先,假设所有节点上的数据均匀分布在一个 d 维的单位超立方体空间中.令用于划分数据空间的总位数为 b,则单元格总数 $V=2^b$.若假设所有的维度之间都是相互独立的.那么,任意一个数据对象 d 落入某一个单元格中的概率 $P(d)$将正比于该单元格的体积在整个数据空间中所占的比例,

$$P(d)=\frac{1}{V}=2^{-b}.$$

然后,令每个节点共享的数据对象个数为 k,每个节点的邻居节点个数为 n.给定索引范围的最大值 $soi_{\max}$,则属于该索引范围内的节点总数 N 和所有这些节点上包含的数据对象总数 D 分别为

$$N=\frac{n^{soi_{\max}+1}-1}{n-1},\ D=k\times\frac{n^{soi_{\max}+1}-1}{n-1}=k\times N.$$

最后,给定一个近似向量与索引范围的最大值 $soi_{\max}$,则该索引范围内的所有数据对象都属于同一个单元格,或者具有相同的近似向量,表示的概率 P 为

$$P=1-(1-2^{-b})^D\approx\frac{D}{2^b}.$$

例如,假设 $soi_{\max}=3$,$n=5$,$k=100$,并且 $b=80$,则概率 $P\approx0.31\times10^{-20}$.也就是说,在 P2P 网络中,若干数据对象具有相同的近似向量的概率很小.在最坏情况下,每个节点用于维护 EVARI 的网络带宽和存储资源的开销为 $O(n^s)$,其中,s 表示索引范围的最大值,n 表示每个节点的邻居节点的个数.在一段时间 Δt 内,如果每个节点加入或者退出系统的可能性符合概率函数 $f(t)$,那么每个节点对 EVARI 的维护代价 C 为

$$C=D\times\int_t^{t+\Delta t}f(t)\mathrm{d}t.$$

基于均匀分布,上述结论是正确的.然而,上述分析忽略了节点之间共享数据的相似程度.换句话说,如果节点之间共享的数据相同(多个节点拥有同一数据的副本)或者相似(不同的数据但属于同一个单元格),以及通过网络自配置(共享相同或相似数据的节点邻近),就可以降低用于维护 EVARI 的网络带宽与存储资源.在现实生活中,数据的非均匀分布是一种常见现象,而且在非结构化 P2P 系统中,绝大多数查询是针对“流行”数据的[3].实验将证实,用于维护 EVARI 的网络带宽和存储代价将远小于上述的理论分析值.

4.2 查询处理代价模型

根据查询处理算法,每个节点上处理查询的搜索代价包括两部分:扫描所有单元格的计算代价和扫描候选单元格中数据点的计算代价.一个范围查询 Q 所覆盖的单元格数与其体积 V_Q(是一个球体)和整个数据空间(体积为 1)的比值成正比.由于任意一个数据对象 d 落入某一个单元格中的概率为 $P(d)$,每个节点上包含的数据对象总数为 D,所以在一个节点上执行范围查询的代价 C_Q 为

$$C_Q=V_Q\times2^b+V_Q\times2^b\times P(d)\times D=V_Q\times(2^b+D).$$

假设在一个范围查询访问的节点总数为 N_{Total},那么查询处理代价总和 C_{Total} 为

$$C_{Total}=C_Q\times N_{Total}.$$

4.3 确定空间划分位数

理论上,每个维度上的空间划分位数既不是越小越好,也不是越大越好.一方面,如果每个维度的划分位数为 0,即将整个数据空间作为一个单元格,那么每个节点上只要共享数据,就会被范围查询搜索.这是因为,范围查询一定是属于数据空间的一部分.另一方面,如果每个维度的划分位数足够大,以至于每个单元格仅包含一个数据,那么每个节点的索引项就变成了对单个数据点的索引.基于均匀分布,根据上述分析,每个维度的划分位数至少为 1.假设系统中的数据个数为 N,那么在最坏情况下,整个空间被划分出的单元格数目要小于 N.

$$1<2^b<N\Rightarrow 0<b<\lg N\Rightarrow 0<d\times b_i<\lg N\Rightarrow 1\le b_i<\frac{\lg N}{d}=b_{\max}.$$

因此,每个维度上的划分位数应该是介于 1~$b_{\max}$之间的一个整数.注意到,上述结论适用于均匀分布的情况.在数据倾斜的情况下,每个维度的划分位数的取值要大于 $b_{\max}$.仿真实验将通过选择不同的划分位数来确定最优的取值.此外,实验还将证实在数据倾斜的情况下,每个维度的划分位数确实要大于 $b_{\max}$.

5 性能评价

为了验证基于 EVARI 的范围搜索的性能,实现了一个用 JAVA SDK 1.4 编写的 P2P 模拟器.所有仿真实验都运行在一个具有 Intel Xeon 2.8GHz 和 1.5GB 主存的 Linux 服务器上.实验模拟了 1 024 个节点的 P2P 网络.网络拓扑遵循 Power-Law[23],节点的平均出度为 4.07.表 1 给出了详细的仿真实验参数设置与说明.

Table 1 Simulation parameters and default values

表 1 仿真实验参数设置

Simulation parameters	Default values	Description
Transfer rate of WAN	3.768KB/s	Average transfer rate between peers in WAN
Transfer rate of LAN	1 297.5KB/s	Average transfer rate between peers in LAN
Max user wait time	4 000ms	The maximum time for a user to wait an answer
TTL	6	The time-to-live of a query message

仿真实验使用的数据集遵循正态分布,包含了 10 个簇(cluster).数据集包含 200 000 个数据对象,维度从 8 维变化到 20 维.每个节点随机地从数据集中选取 200 个数据对象作为本地的共享数据集,选取 20 个数据点生成范围查询.范围查询的搜索半径是介于 0.05~0.1 中的任意一个数值.

由于现有的非结构化 P2P 系统并不支持多维数据上的范围搜索,所以采用了 Gnutella[3]系统作为基准测试,与基于 EVARI 的范围搜索方法进行比较.仿真实验采用了查全率、覆盖度、查询响应时间、维护 EVARI 的时间代价以及用于维护 EVARI 的存储与网络带宽消耗来衡量系统性能.前 3 个指标用于衡量基于 EVARI 的范围搜索方法的有效性,后 3 个指标则用于评价 EVARI 的建立与维护代价.

5.1 实验结果与分析

本节将首先测试空间划分位数对搜索性能的影响.然后依次测试查询响应时间、索引的维护时间代价、存储资源和网络带宽消耗以及网络自配置对系统性能的影响.最后,测试了节点的动态性对系统性能的影响.

5.1.1 空间划分位数对系统性能的影响

在这组实验中,查全率和覆盖度被用来评价系统性能.没有考虑查准率是由于范围查询与近似向量之间的下界小于或等于范围查询与近似向量中的任意一个数据对象之间的距离,所以查准率一定是 100%.查全率是指给定一个时间约束(查询的生命周期),系统找到的满足查询条件的结果数量占系统当时所有在线节点维护的满足查询条件的结果总数的比率.覆盖度是指给定一个时间约束,基于 EVARI 的范围搜索方法访问到的节点个数与 Gnutella 搜索方法访问到的节点个数之间的比值.虽然查全率和覆盖度都是反映系统性能的指标,但是它们侧重的角度不同:查全率越高说明范围搜索的效果也就越好;覆盖度越低说明范围搜索所消耗的节点上的计算资源与网络带宽资源也就越少.因此,P2P 系统要求在保证查全率的前提下,尽可能地降低覆盖度.

这组实验运行在从 8 维变化到 20 维的数据集.每个维度分别被划分成 8,16 或 32 个区间.基于 Gnutella 的搜索策略被用作基准测试.图 5~图 7 展示了空间划分位数对查全率的影响.从实验结果中可以观察到两个事实:

首先,基于 EVARI 的范围搜索方法的查全率几乎不受数据维度的影响.这说明 EVARI 受“维度灾难”影响很小;其次,随着 *soi* 取值的扩大,基于 EVARI 的范围搜索方法的查全率提升得非常快,并接近于 Gnutella 的查全率.这是因为,如果 *soi* 取值扩大了,那么 EVARI 反映出的局部数据分布也就更准确了,所以查全率会随之上升.注意到 Gnutella 的查全率是最好的,这是由于 Gnutella 将在整个网络中广播范围查询.

另一方面,横向比较图 5~图 7 中的实验结果可以发现,对于同一个索引范围而言,查全率将随着空间划分位数的增长而迅速下降.例如,取 soi_{max}=2,如果每个维度被划分成 8 个区间,那么查全率为 99%(如图 5 所示);如果每个维度被划分成 16 个区间,那么查全率就会下降为 90%(如图 6 所示);而如果每个维度被划分成 32 个区间,那么查全率将只有 65%(如图 7 所示).

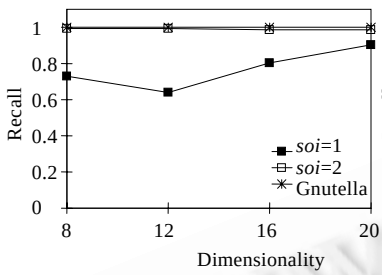

Fig.5 Recall: 8 partitions/dim

图 5 查全率:8 个区间/维

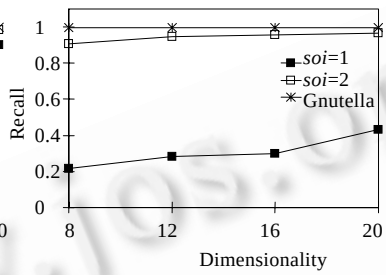

Fig.6 Recall: 16 partitions/dim

图 6 查全率:16 个区间/维

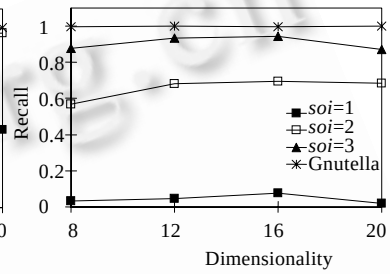

Fig.7 Recall: 32 partitions/dim

图 7 查全率:32 个区间/维

从上述实验分析中似乎可以得出这样一个结论:每个维度的划分位数越少,那么基于 EVARI 的范围搜索的性能越好.然而,我们忽略了一个因素,到底会有多少节点参与了范围搜索,即查询的覆盖度是多少.给定一个适当的查全率,只有尽可能地减少范围搜索访问到的节点个数,才能保证基于 EVARI 的范围搜索方法具备良好的可扩展性.为此,下面将测试空间划分位数对覆盖度的影响.

显然,如果覆盖度越小且系统的查全率越高,那么 EVARI 用于裁剪搜索空间的效果也就越好.图 8~图 10 展示了空间划分位数对覆盖度的影响.随着每个维度上划分位数的增长,覆盖度下降得非常快.例如,取 soi_{max}=2,如果每个维度被划分成 8 个区间,那么覆盖度为 95%(如图 8 所示);如果每个维度被划分成 16 个区间,那么覆盖度就会下降到 70%(如图 9 所示);而如果每个维度被划分成 32 个区间,那么覆盖度将只有 30%(如图 10 所示).

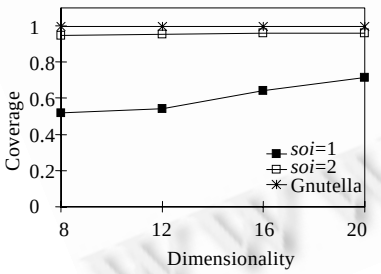

Fig.8 Coverage: 8 partitions/dim

图 8 覆盖率:8 个区间/维

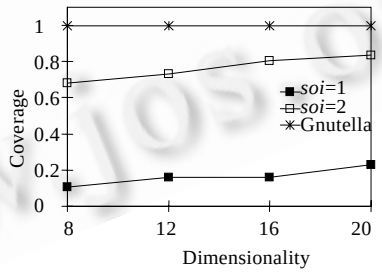

Fig.9 Coverage: 16 partitions/dim

图 9 覆盖率:16 个区间/维

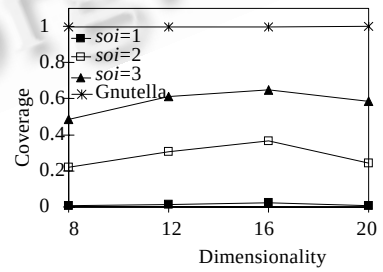

Fig.10 Coverage: 32 partitions/dim

图 10 覆盖率:32 个区间/维

因此,结合图 5~图 10 给出的实验结果,当每个维度被划分成 32 个区间且索引范围等于 3 时,在保证查全率为 90%的前提下,系统能够将覆盖度降低到 50%.很明显,与其他空间划分位数和索引范围的组合相比,这个组合是最优的.在后面的实验中,每个维度均被划分成 32 个区间.

5.1.2 查询响应时间和维护索引的时间代价

除了查全率,在范围搜索过程中,用户关心的另一个指标是查询响应时间.查询响应时间是指用户从提交查询到获得第 1 个结果需要等待的时间.显然,查询响应时间越短,用户的满意程度也就越高.在仿真实验中,假设不会出现网络拥塞,所以 Gnutella 的查询响应时间是最短的.从图 11 中可以看到,随着索引范围的扩大,查询响应

时间下降得很快.这说明,如果 EVARI 反映出的局部数据分布信息越准确,那么范围搜索的效果越好.当索引范围等于 3 时,基于 EVARI 的范围搜索方法的查询响应时间已经非常接近 Gnutella 方式下的查询响应时间了.注意到,图 11 中曲线的波动是由于随机生成范围查询所造成的.

维护索引的时间代价是指在一个节点加入系统之后,该节点建立本地 EVARI 和所有邻居节点更新它们自己的 EVARI 的时间代价总和.从理论上讲,在维护索引的过程中,索引范围越大,新加入系统的节点与邻居节点之间交换的信息也就越多.因此,维护索引的时间代价也越大.这是因为,维护索引的时间代价是由节点之间的信息交换量决定的.此外,数据维度越大,节点之间交换的信息量也越大.因此,维护索引的时间代价也会随着数据维度的增长而增大.从图 12 中可以看出,时间代价是随着数据维度的增长呈线性增长的,而随着索引范围的扩大,时间代价增长得较多.这是因为,需要更新 EVARI 的节点数目会随着索引范围的扩大而增长.然而,时间代价并没有随着索引范围的扩大而呈指数级增长.这是由于仿真实验使用的数据集中存在着簇.因此,在建立和更新索引的过程中,许多近似向量并不会在索引范围以内的所有节点之间进行传播.

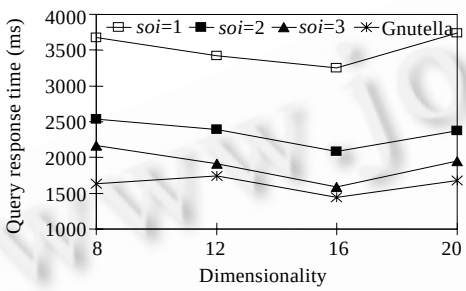

Fig.11 Query response time

图 11 查询响应时间

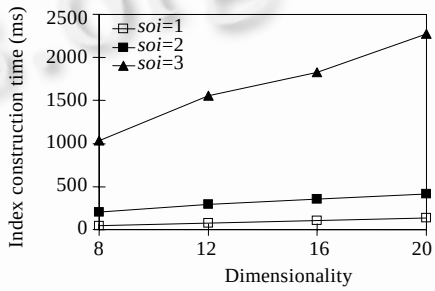

Fig.12 Maintenance cost of EVARI

图 12 维护 EVARI 的时间代价

5.1.3 存储资源与网络带宽消耗

由于节点之间需要通过交换近似向量来建立 EVARI,所以每个节点不可避免地要消耗一些存储资源和网络带宽资源.本节将测试节点在建立与维护 EVARI 的过程中所消耗的这两种资源.

图 13 给出为了维护 EVARI,每个节点消耗的存储资源情况.从实验结果中可以看出,每个节点为 EVARI 提供的存储空间随着数据维度的增长呈线性增长,并没有随着索引范围的扩大而以指数方式增长.这是因为,一些不同的数据对象具有相同的近似向量表示,所以在交换近似向量之后,EVARI 所需的存储空间并不会增长得很多.也就是说,同一个近似向量可以“吸收”很多个节点上的许多不同的数据对象.

图 14 给出了每个节点为建立 EVARI 消耗的网络带宽.对于每个节点,实验统计了该节点发送给每个邻居节点的消息量、邻居节点发送给该节点的消息量以及这两部分消息量的总和.从实验结果中可以看出,当索引范围小于 3 时,每个节点为建立 EVARI 所消耗的网络带宽基本相同,而当索引范围等于 3 时,节点之间交换的消息量增长得较多.即便如此,最大网络带宽消耗也不过 26KB,这对于广域网中的节点而言是可以接受的.

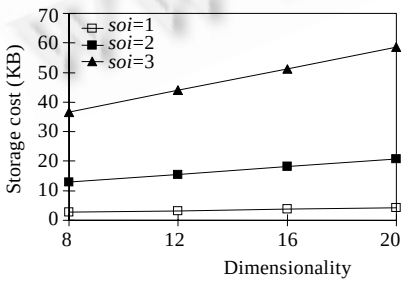

Fig.13 Storage cost of maintaining EVARI

图 13 维护 EVARI 所需的存储消耗

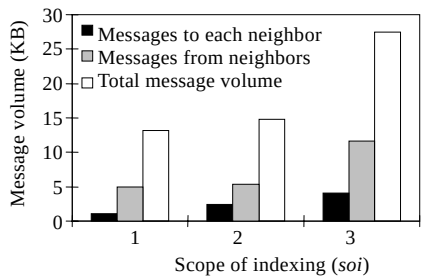

Fig.14 Bandwidth cost of constructing EVARI

图 14 建立 EVARI 所需的带宽消耗

5.1.4 网络自配置对系统性能的影响

本节将评价网络自配置对系统性能的影响.仿真实验使用了 20 维的数据集,并采用查全率和覆盖度两个指标来评价系统性能.图 15 给出了网络自配置对查全率的影响.当查询生命周期小于 4 时,使用网络自配置的查全率要比无网络自配置的查全率好 2~4 倍,甚至远远好于 Gnutella 的查全率.这是因为,通过网络自配置,查询请求节点与那些能够为它提供较多结果的节点建立了邻居关系,而这些节点将最先被查询访问到.另一方面,从图 16 中可以发现,通过网络自配置功能,虽然查全率上升得很快,但是查询访问到的节点数量并没有增加多少,即覆盖度几乎没有增长.因此,网络自配置确实改善了系统资源的利用率.然而,当查询生命周期大于 4 时,即使采用了网络自配置,性能也没有提升很多.这是因为,与查询相关的节点基本上都已经被访问到了.

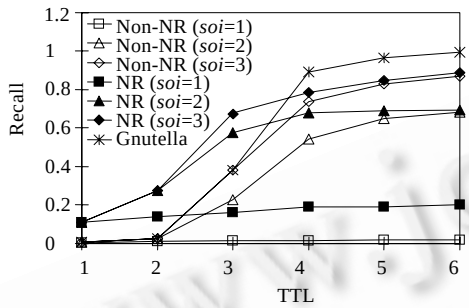

Fig.15 Effect of network reconfiguration on recall

图 15 网络自配置对查全率的影响

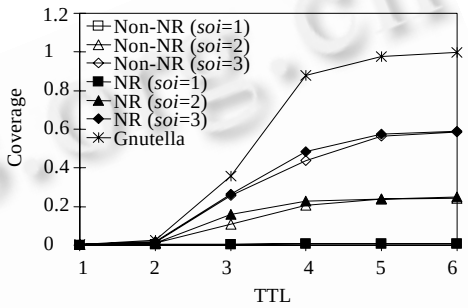

Fig.16 Effect of network reconfiguration on coverage

图 16 网络自配置对查询覆盖度的影响

5.1.5 节点的动态性对系统性能的影响

上述实验结果是在相对稳定的 P2P 仿真环境下得到的,即只有少数节点加入或退出系统.为了评价基于 EVARI 的范围搜索方法对 P2P 系统的动态特征的适应性,这组实验测试了当从 5%~40%节点失效时的性能变化情况.这里,节点失效是指由于网络连接不稳定或硬件故障等因素而导致节点退出系统.实验分别在 12 维、16 维和 20 维的数据集上对系统的查全率和查询响应时间进行了测试,实验结果如图 17~图 22 所示.

从图 17~图 22 中可以看到,随着失效节点个数的增加,查全率逐渐降低,而查询响应时间则逐渐上升.在确定索引范围的情况下,采用网络自配置的查全率要大于无网络自配置的查全率,而查询响应时间则正好相反.这再次证明网络自配置确实能够有效地提高系统性能.此外,系统性能的下降趋势与失效节点个数呈线性关系,而受数据维度变化的影响很小,所以,本文提出的范围搜索方法能够很好地适应 P2P 系统的动态特征.

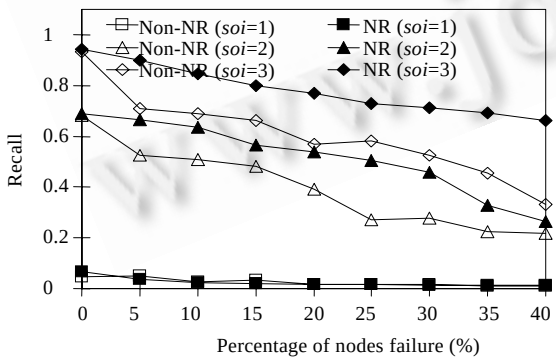

Fig.17 Recall: 12 dim. dataset

图 17 查全率:12 维数据集

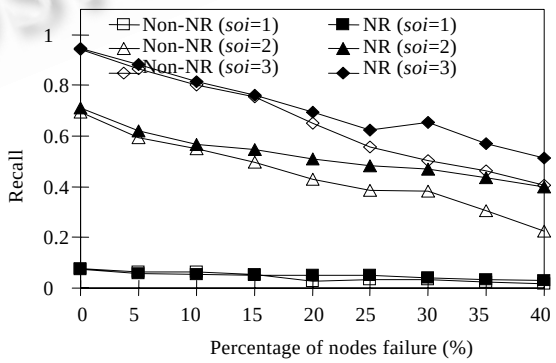

Fig.18 Recall: 16 dim. dataset

图 18 查全率:16 维数据集

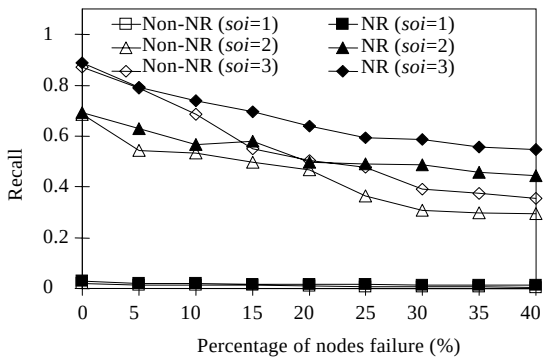

Fig.19 Recall: 20 dim. dataset

图 19 查全率:20 维数据集

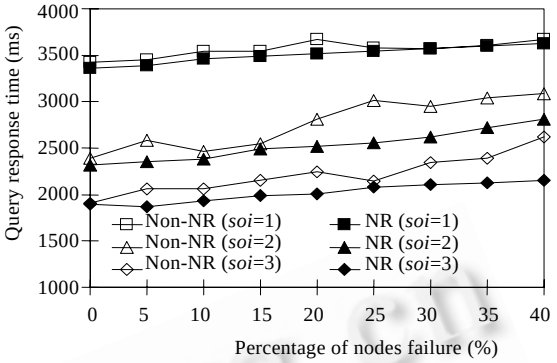

Fig.20 Coverage: 12 dim. dataset

图 20 覆盖度:12 维数据集

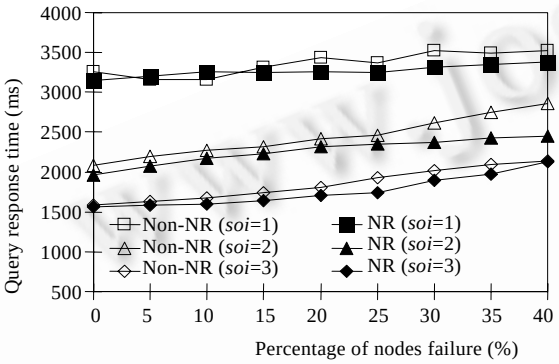

Fig.21 Coverage: 16 dim. dataset

图 21 覆盖度:16 维数据集

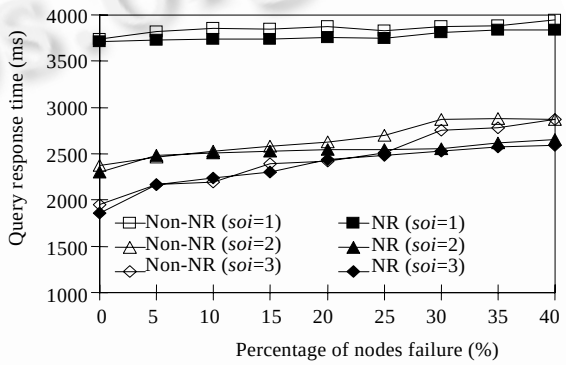

Fig.22 Coverage: 20 dim. dataset

图 22 覆盖度:20 维数据集

6 结束语

本文提出了一种路由索引结构,给出了范围搜索和索引维护算法;仿真实验结果证明了这种索引结构非常适合于非结构化 P2P 环境下的范围搜索.未来的研究工作主要包括:第一,改进 EVARI 使其支持 *k*-最近邻搜索;第二,采用聚类技术,依据单元格中的数据密度,设计出效果更好的网络自配置算法.

致谢 在此,我们感谢新加坡国立大学的黄铭钧(OOI Beng Chin)教授和陈建利教授(TAN Kian-Lee)对本文工作给予的支持和建议.

References:

[1] Gribble SD, Halevy AY, Ives ZG, Rodrig M, Suciu D. What can database do for peer-to-peer? In: Mecca G, Siméon J, eds. Proc. of the Int'l Workshop on the Web and Databases. ACM Press, 2001. 31−36.

[2] Bernstein P, Giunchiglia F, Kementsietsidis A, Mylopoulos J, Serafini L, Zaihrayeu I. Data management for peer-to-peer computing: A vision. In: Fernandez MF, Papakonstantinou Y, eds. Proc. of the Int'l Workshop on the Web and Databases. Wisconsin: ACM Press, 2002. 89−94.

[3] Chawathe Y, Ratnasamy S, Breslau L, Lanham N, Shenker S. Making Gnutella-like p2p systems scalable. In: Feldmann A, Zitterbart M, Crowcroft J, Wetherall D, eds. Proc. of the ACM SIGCOMM Conf. on Applications, Technologies, Architectures, and Protocols for Computer Communication. Karlsruhe: ACM Press, 2003. 407−418.

[4] Lü Q, Cao P, Cohen E, Li K, Shenker S. Search and replication in unstructured peer-to-peer networks. In: Proc. of the Int'l Conf. on Measurements and Modeling of Computer Systems. ACM Press, 2002. 84−95.

[5] Crespo A, Garcia-Molina H. Routing indices for peer-to-peer systems. In: Sivilotti P, ed. Proc. of the Int'l Conf. on Distributed Computing Systems. Arizona: IEEE Computer Society, 2002. 23−34.

[6] Yang B, Garcia-Molina H. Improving search in peer-to-peer networks. In: Sivilotti P, ed. Proc. of the Int'l Conf. on Distributed Computing Systems. Arizona: IEEE Computer Society, 2002. 5−14.

[7] Halevy AY, Ives ZG, Suciu D, Tatarinov I. Schema mediation in peer data management systems. In: Dayal U, Ramamritham K, Vijayaraman TM, eds. Proc. of the Int'l Conf. on Data Engineering. Bangalore: IEEE Computer Society, 2003. 505−516.

[8] Kementsietsidis A, Arenas M, Miller RJ. Mapping data in peer-to-peer systems: Semantics and algorithmic issues. In: Halevy AY, Ives ZG, Doan AH, eds. Proc. of the ACM SIGMOD Int'l Conf. on Management of Data. ACM Press, 2003. 325−336.

[9] Huebsch R, Hellerstein JM, Lanham N, Loo BT, Shenker S. Querying the Internet with PIER. In: Freytag JC, Lockemann PC, Abiteboul S, Carey MJ, Selinger PG, Heuer A, eds. Proc. of the Int'l Conf. on Very Large Data Bases. Berlin: Morgan Kaufmann Publishers, 2003. 321−332.

[10] Sahin OD, Gupta A, Agrawal D, Abbadi AE. A peer-to-peer framework for caching ranges queries. In: Proc. of the Int'l Conf. on Data Engineering. Boston: IEEE Computer Society, 2004. 165−176.

[11] Ng WS, Ooi BC, Tan KL, Zhou AY. PeerDB: A p2p-based system for distributed data sharing. In: Dayal U, Ramamritham K, Vijayaraman TM, eds. Proc. of the Int'l Conf. on Data Engineering. Bangalore: IEEE Computer Society, 2003. 633−644.

[12] Gaede V, Gunther O. Multidimensional access methods. ACM Computing Surveys, 1998,30(2):170−231.

[13] Ganesan P, Yang B, Garcia-Molina H. One Torus to rule them all: Multi-Dimensional queries in p2p systems. In: Amer-Yahia S, Gravano L, eds. Proc. of the Int'l Workshop on the Web and Databases. Paris: ACM Press, 2004. 19−24.

[14] Stoica I, Moris R, Karger D, Kaashoek F, Balakrishnan H. Chord: A scalable peer-to-peer lookup service for Internet applications. In: Proc. of the ACM SIGCOMM Conf. on Applications, Technologies, Architectures, and Protocols for Computer Communication. San Diego: ACM Press, 2001. 149−160.

[15] Ratnasamy S, Francis P, Handley M, Karp R, Shenker S. A scalable content-addressable network. In: Proc. of the ACM SIGCOMM Conf. on Applications, Technologies, Architectures, and Protocols for Computer Communication. San Diego: ACM Press, 2001. 161−172.

[16] Zhang C, Krishnamurthy A, Wang RY. SkipIndex: Towards a scalable peer-to-peer index services for high dimensional data. Technical Report, TR-703-04, Princeton University, 2004.

[17] Aspnes J, Shah G. Skip graphs. In: Proc. of the ACM-SIAM Symp. on Discrete Algorithms. Maryland: ACM Press, 2003. 384−393.

[18] Shen HT, Shu YF, Yu B. Efficient semantic-based content search in P2P network. IEEE Trans. on Knowledge and Data Engineering, 2004,17(7):813−826.

[19] Weber R, Schek HJ, Blott S. A quantitative analysis and performance study for similarity search methods in high dimensional spaces. In: Gupta A, Shmueli O, Widom J, eds. Proc. of the Int'l Conf. on Very Large Data Bases. New York: Morgan Kaufmann Publishers, 1998. 194−205.

[20] 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 PA, Ooi BC, eds. Proc. of the Int'l Conf. on Very Large Data Bases. Trondheim: ACM Press, 2005. 661−672.

[21] Ramabhadran S, Ratnasamy S, Hellerstein JM, Shenker S. Brief announcement: Prefix hash tree. In: Chaudhuri S, Kutten S, eds. Proc. of the ACM Symp. on Principles of Distributed Computing. ACM Press, 2004. 368.

[22] Ng WS, Ooi BC, Tan KL. BestPeer: A self-configurable peer-to-peer system. In: Proc. of the Int'l Conf. on Data Engineering. San Jose: IEEE Computer Society, 2002. 272.

[23] Palmer CR, Steffan JG. Generating network topologies that obey power law. In: Proc. of the Global Internet Symp. 2000.

徐林昊(1976－),男,江苏盐城人,博士,主要研究领域为对等计算系统中的查询处理.

钱卫宁(1976－),男,博士,讲师,主要研究领域为Web数据管理与数据挖掘,对等计算,流数据分析与处理.

周傲英(1965－),男,博士,教授,博士生导师,CCF高级会员,主要研究领域为Web数据管理与数据挖掘,对等计算,流数据管理.