基于分布式范围树的结构化 P2P 多维范围查询*)

傅向华 彭小刚 王志强 明 仲
(深圳大学信息工程学院 深圳 518060)

摘 要 基于分布式哈希表(DHT)的结构化 P2P 网络具有扩展性好、健壮和自组织等优点,但只支持精确匹配的查询。本文提出一种基于分布式范围树的结构化 P2P 范围查询方法(DRT-RQ),该方法将多维索引的分布式范围树分发到已有的结构化 DHT 覆盖网络中,利用 DHT 系统提供的数据查找接口,有效实现数据对象的范围查询。实验结果表明,基于分布式范围树的范围查询(DRT-RQ)比基于前缀哈希树的范围查询(PHT-RQ)需要更短的查询延时。

关键词 分布式范围树,分布式哈希表,结构化 P2P 网络,范围查询

Multidimensionality Range Query Based on Distributed Range Tree in Structured Peer-to-Peer Networks

FU Xiang-Hua PENG Xiao-Gang WANG Zhi-Qiang MING Zhong
(College of Information Engineering, Shenzhen University, Shenzhen 518060)

Abstract Distributed Hash Tables are scalable, robust, and self-organizing Peer-to-Peer systems that support exact match lookups. This paper describes the design and implementation of a distributed range tree based multidimensionality range query (DRT-RQ)in structured Peer-to-Peer networks. This method enables more sophisticated queries over a DHT, which distribute the distributed range tree to existing DHT overlay network, and then uses the lookup interface of a DHT to implement the range query efficiently. Compared the query performance with the prefix hash table based range query (PHT-RQ), the experiment results showe that the DRT-RQ consumes less query latency.

Keywords Distributed range tree, Distributed hash table, Structured peer-to-peer networks, Range query

1 引言

基于分布式哈希表(DHT)的结构化 P2P 系统(如 Chord[1], CAN[2], Pastry[3], Tapestry[4])为每个网络结点和数据对象分配唯一标识 NodeID 和键 key,结点被组织成特定结构的覆盖网络,键被映射到网络中保持有与该键相关数据对象的节点上,从而确保将查询键值在 O(log$N$)(N 为结点数目)跳跃次数内路由到数据对象的存储结点。基于 DHT 的结构化 P2P 网络具有扩展性好、健壮、自组织、无需中心化认证等优点,已被用于构建文件系统、重定向服务、事件通知和内容分发等多种应用。随着应用的越来越广泛,越来越要求 DHT 系统支持多关键词查询、范围查询、等式查询、联合查询、相似性查询等复杂查询方式[5,6]。然而,由于 DHT 系统使用散列来均匀地分发键,用于请求某个值范围内所有数据对象的范围查询在 DHT 系统中特别难以实现[7]。为支持范围查询,一些结构化 P2P 系统不得不使用不同于 DHT 的结构,如 Mercury[8]采用圆形覆盖网络并连续存储数据以支持多属性范围查询;SkipNet[9]基于 SkipGraph 结构,使用名字而不是散列后的标识在覆盖网络中排序结点,以保留基于对象名字的自然局部性,但这些特定设计的结构需要显式地考虑负载均衡问题。前缀哈希树[7](PHT)虽然通过在 DHT 之上附加一个 trie 结构使 DHT 系统能够较好地支持范围查询,但由于 PHT 将键存储在具有相同前缀的叶结点上,导致客户无法得知整个 PHT 的结构,必须花费额外的 DHT 查找时间到达具有最长匹配前缀的叶结点上,使查询响应时间较慢。

范围查询是经常使用的查询方式,为在 DHT 结构化网络上支持范围查询,改善查询效率,本文提出一种基于分布式范围树的结构化 P2P 范围查询方法。该方法将多维索引的分布式范围树分发到已有的结构化 DHT 覆盖网络中,利用 DHT 系统提供的简单 Lookup(key)数据查找接口,有效实现数据对象的范围查询。

2 多维范围树的数据结构

设网络中存储的数据点集合为 P,每个数据点包含 d 个属性,那么任意数据点 p_i 可表示为 d 元组($x_{i,1}$, $x_{i,2}$, …, $x_{i,d}$),所有的数据点集构成一个 d 维的欧几里得空间 E^d。若数据点第 j 维的属性值下界和上界分别为 Low_j 和 $High_j$ 那么欧几里得空间 E^d 中的任意一个区域 $R=[g_1,h_1]\times[g_2,h_2]\times\cdots\times[g_d,h_d]$ 将包含数据点集 P 的一个子集 P',其中 $Low_j\leqslant g_j\leqslant h_j\leqslant High_j$, $j=1,2,\cdots,d$。范围查询问题就是确定给定区域 R 中的所有数据点,包含 d 个属性的数据点,可能需要进行 d 维的范围查询,为简化问题,我们首先考虑一维范围查询的情况。

2.1 一维线段树

线段树 $T(a,b)$[10]是一棵二叉树,其中参数 a,b 是表示某一区间的起点与终点的整数($b\geqslant a$),若根节点用 v 表示,则可记为 $B[v]=a$, $E[v]=b$。线段树 $T(a,b)$可以递归地构造如下:它包含一个根 v,根具有参数 a 和 b,且若 $b-a>1$,则它有左子树 $T(a,\lfloor(B[v]+E(v))/2\rfloor)$和右子树 $T(\lfloor(B[v]+E$

*)基金项目:深圳大学科研启动基金(200648)资助。傅向华 博士,讲师,主要研究领域为对等计算,智能化网络;彭小刚 博士,讲师;王志强 副教授;明 仲 博士,教授。

$(v))/2\rfloor, b)$，其中$\lfloor\ \ \rfloor$表示低限。左右子树的根分别记为$LSON[v]$和$RSON[v]$。结点v所代表的区间由$B[v]$和$E[v]$确定，并满足关系式$[B[v], E(v)] \subseteq [a, b]$。若令$L=b-a$，并用$B_{l,k}$和$E_{l,k}$分别表示$l$层第$k$个节点区间的起点和终点，则线段树具有如下性质：

性质1 线段树的高度$H=\log L+1$。

性质2 线段树中的每个节点表示了一个节点区间$[B_{l,k}, E_{l,k}]$($l\in[0,\log L]$，$k\in[0,2^l-1]$)，区间长度$L_{l,k}=B_{l,k}-E_{l,k}$。

性质3 每个非叶节点有两个孩子，左孩子和右孩子分别表示区间$[B_{l,k}, \lfloor B_{l,k}+E_{l,k}\rfloor/2]$和$[\lfloor B_{l,k}k+E_{l,k}\rfloor/2+1, E_{l,k}]$。

性质4 对于同一层的相邻节点，有$B_{l,k}=E_{l,k-1}+1$，因此，线段树每一层所表示的区间是连续的，并且，同一层的所有节点组成线段树表示的整个范围，则对于任何有$l\in[0,\log L]$有$\bigcup_{k=0}^{k=2^l-1}[B_{l,k}, E_{l,k}]=[B(v), E(v)]$。根据线段树的性质，可以得到如下的定理：

定理1 任意一个查询范围$[g,h]$($a\leqslant g\leqslant h\leqslant b$)都可以划分为一些$T(a,b)$表示的节点区间的并集，即$[g,h]k=\bigcup[B_{l,k}, E_{l,k}]$。

定理1可以根据性质4直观地得到，但对于任意的查询范围，其在$T(a,b)$中的区间划分是不唯一的。在所有的区间划分中，存在一个使区间数目最小的划分。

定义1 在$T(a,b)$中，使范围$[g,h]$获得最小划分区间数目的划分称为规范化划分，得到的每个划分区间称为规范区间，每个规范区间对应的节点称为范围$[g,h]$的分配节点。在任意以分配节点v_i为根的子树中，称该子树叶子节点上存储的所有数据点集$S(v_i)$为规范子集。

定理2 任意一个范围$[g,h]$($a\leqslant g\leqslant h\leqslant b$)可以最多划分为线段树$T(a,b)$中的$2\log_2 L$个规范区间标准子集，并将所有确定范围$[g,h]$的划分的节点称为分配节点。

定理2可以根据范围查询的过程来证明，简叙如下：对于$[g,h]$，假设在$T(a,b)$中得到的最长的划分区间为t，那么在表示$[g,h]$的所有规范区间中，t左边的规范区间应该都为线段树的右孩子，而t右边的规范区间应该都为线段树的左孩子，而在线段树中，最多只有$\log_2 L$个左孩子和$\log_2 L$个右孩子，因此$[g,h]$最多可以用线段树中的$2\log_2 L$个规范区间表示。若查询范围包含的数据子集P'数目为f，还可以证明，线段树的查询时间为$O(n\log n+f)$，存储空间为$O(\log n)$，构造时间为$O(n\log n)$。

2.2 多维范围树

对于包含n个数据点的d维数据集合$P=\{p_1, p_2, \cdots, p_n\}$，其所有数据点第$j$维的属性值为$\{x_j(p_1), x_j(p_2), \cdots, x_j(p_n)\}$($1\leqslant j\leqslant d$)。可以根据数据点的次序，将每一维规范成$[1,n]$中的整数。那么，在每一维中，这$n$个点将确定$n-1$个基本区间$[i,i+1]$($i=1,2,\cdots,n$)。$d$维范围树[11]可以通过如下的递归构造过程得到：

(1)构造数据集合P第1维属性值$\{x_1(p_i)\mid p_i\in p\}$对应的线段树$T_1(1,n)$。$T_1(1,n)$上的每个节点$v$所对应的标准子集由所有第1维属性值位于区间$[B(v), E(v)]$中的数据点组成，用$S_d(v)$表示，则可定义一个$(d-1)$维集合$S_{d-1}(v)\triangleq\{x_2(p_i), \cdots, x_d(p_i)\mid p_i\in S_d(v)\}$。

(2)$T_1(1,n)$的节点v有一个指针指向$S_{d-1}(v)$的范围树$T_{assoc}(v)$。

例如，若要构造二维的范围树，首先构建第1维的线段树$T(1,n)$，然后将$T(1,n)$中的每个节点v的标准子集$S(v)$构造成一棵线段树$T_{assoc}(v)$，并在每个节点v存放一个指向$T_{assoc}(v)$的指针。在图1中，(a)图为二维空间上的一些点，(b)图为这些点的二维范围树。(b)图的基本结构为数据点集第1维所构成的线段树，线段树的每个节点链接到其标准子集形成第2维的线段树。为简便起见，图中用一个三角形表示每个标准子集在第2维上的线段树。

若已知查询区域$R=[g_1,h_1]\times[g_2,h_2]\times\cdots\times[g_d,h_d]$，在范围树中进行范围查询的过程如下：首先在1维线索树T_1中根据范围$[g_1,h_1]$进行查询，确定$[g_1,h_1]$的分配节点，然后在每个分配节点的关联结构范围树$T_{assoc}(v)$中对$[g_2,h_2]$进行查询。依此类推，直到最后完成对$[g_d,h_d]$范围的查询，确定出满足要求的数据点。

对于数据点集合数目为n，数据点的维数为d的范围查询，可以证明基于范围树的方法耗费的查询时间为$O((\log n)^d)$，存储为$O(n(\log n)^{d-1})$，预处理时间为$O(n(\log n)^{d-1})$。

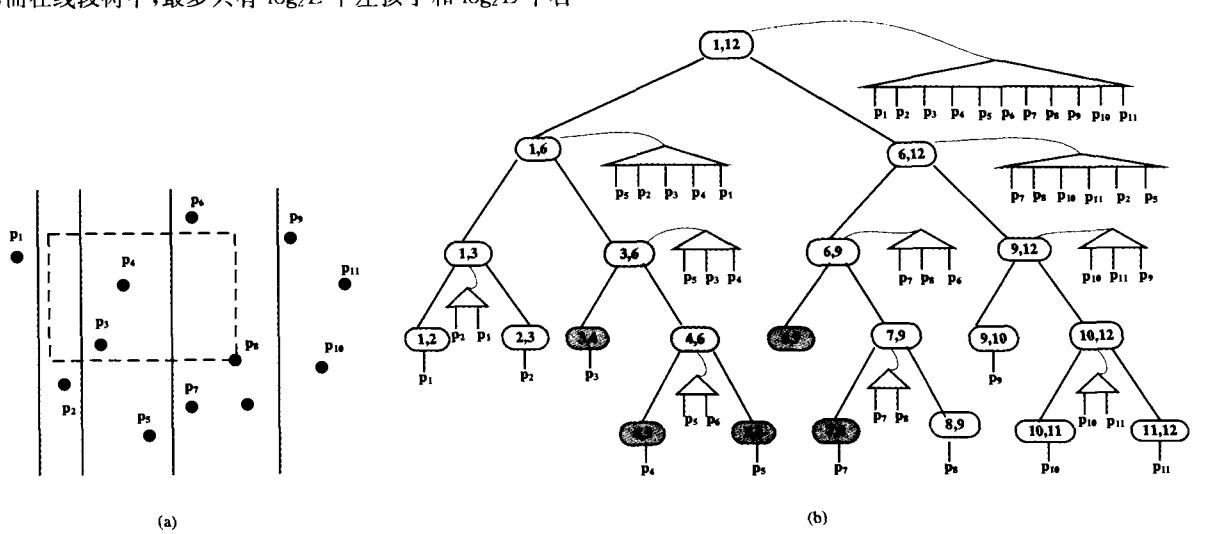

图1 二维范围树的例子

3 基于分布式范围树的多维范围查询

可以利用底层 DHT 的组织方式，方便地将范围树上的区间节点分配到 DHT 网络节点上，建立范围树与 DHT 之间的联系。范围树在 DHT 上的分配方式如下：首先利用散列函数对范围树的节点区间$([g_1,h_1],[g_2,h_2],\cdots,[g_d,h_d])$进行散列，得到键值$key'=Hash([g_1,h_1],[g_2,h_2],\cdots,[g_d,h_d])$，然后区间节点$([g_1,h_1],[g_2,h_2],\cdots,[g_d,h_d])$分配给与$key'$对应的 DHT 网络节点。这样，范围树中的每个区间节点与 DHT 中的网络节点对应起来，而有关范围树的任何信息都可以通过 DHT 的查找操作高效地完成。分布式范围树上的操作包括数据对象键值 key 的插入、删除以及范围查询操作。

3.1 数据索引的插入与删除

在 DHT 中，对于给定的数据对象 p，需要将其键值$key=Hash(p)$放置到与其属性值对应的范围树节点上。同样，当删除一个数据对象时，也需要将其在范围树上 key 的值删除掉。设数据点 p 的属性值范围为$[i_1,i_1+1],[i_2,i_2+1],\cdots,[i_d,i_d+1]$，由图 1 可知，当在范围树中添加数据对象的 key 时，需要将 key 插入到特定的叶节点以及该叶节点的所有祖先节点上。也就是说需要将 key 插入到范围树中所有包含$[i_1,i_1+1],[i_2,i_2+1],\cdots,[i_d,i_d+1]$区间的关联结构上。以 2 维数据为例，数据对象的 key 插入算法可描述如下：

算法 1 2 维数据 Key 的插入算法

$2DKeyInsert([i_1,i_1+1],[i_2,i_2+1],T,key)$

Input：属性值$[i_1,i_1+1],[i_2,i_2+1]$范围树 T，数据对象键 key

Output：插入 key 值的范围树

Step1：在范围树 T 的第 1 维线段树中 T_1 进行二叉查找，找到区间为$[i_1,i_1+1]$的叶子节点。记该叶子节点及其祖先节点的集合为 S'_1；

Step2：对与 S'_1 中每个节点 v 相关联的二维线索树 $T_2(v)$进行二叉搜索，找到 $T_2(v)$上区间为$[i_2,i_2+1]$的叶子节点，记该叶子节点及其祖先节点为 $S'_2(v)$。

Step2.1：对于 $S'_2(v)$包含的每个节点 u，令$key'=Hash([B(v),E(v)],[B(u),E(u)])$。

Step2.2：利用 DHT 的底层逻辑，将 key 插入到key'所对应的 DHT 网络节点上。

因为范围树的每一维是完全的二叉树，当数据集合数目 n 给定时，每个对等节点都可以在本地构造出范围树，所以 key 可以同步地插入到叶节点和它所有的祖先节点中。利用并行机制，插入操作可以达到 $O(1)$的时间复杂度，而删除一个 key 的过程与插入过程非常相似，需要将 key 对应的叶节点以及该叶节点的所有祖先节点上存放的索引全部删除掉。

由于在插入 key 值的过程中，顶层的节点会被插入更多的 key，事实上这些节点上的 key 是冗余的，只是为了提高查询效率。为避免有的区间节点包含太多 key，可以为区间节点设置一个阈值 τ，当超过该阈值 τ 时，就不再在该节点插入 key，而只将 key 插入到底层节点中。这样，既可以保留一部分数据冗余，提高查询的效率，又可以提高系统的负载均衡能力。

3.2 数据对象的范围查询

当在一个客户端进行范围查询时，若给定的查询范围为$[g_1,h_1]\times[g_2,h_2]\times\cdots\times[g_d,h_d]$，则客户端首先将查询范围$[g_1,h_1]\times[g_2,h_2]\times\cdots\times[g_d,h_d]$分割成范围树中多个规范区间的并集，然后向 DHT 网络中对应每个规范区间的分配节点发送查询请求，所有这些节点维护的 key 就是查询范围中的数据对象的索引。以 2 维情况为例，范围查询的步骤可以描述如下：

算法 2 2 维范围查询算法

$2DRangeQuery([g_1,h_1]\times[g_2,h_2],T)$

Input：查询范围$[g_1,h_1]\times[g_2,h_2]$，二维范围树 T

Output：查询范围中所有的数据点子集 P'

Step1：利用本地节点构造的范围树 T，基于二叉搜索将$[g_1,h_1]\times[g_2,h_2]$分裂为 k 个规范区间的并集，即$[g_1,h_1]\times[g_2,h_2]=\bigcup_{k\le 2\log n}[g_{1,k},h_{1,k}]\times[g_{2,k},h_{2,k}]$。

Step2：对每个规范区间进行散列，即$key'_k=Hash([g_{1,k},h_{1,k}],[g_{2,k},h_{2,k}])$。

Step3：利用 DHT 的底层查找机制，将所有的 key'_k 路由到对应的网络节点上，获得这些节点维护的所有 key。

Step4：再利用获得的 key 和 DHT 的查找机制，在 DHT 网络中获得数据对象。

Step5：将所有的数据对象返回到查询节点，则得到整个查询范围的结果。

在执行分布式查询的过程中，由于可以利用本地的范围树结构预先确定所有的分配节点，因此同样可以执行并行查询操作，因此，可以达到 $O(1)$的时间复杂度。

4 实验与结果分析

为验证分布式范围树(DRT)在结构化 DHT 网络上的有效性，利用 Java 在 OpenDHT[12]服务之上实现了一个原型系统，其通过 RPC-XML 调用 OpenDHT 服务。OpenDHT 是一个运行在 PlanetLab 之上的免费、公用的 DHT 服务。为了与已有的方法进行比较，我们还实现了前缀哈希树(PHT)，对两种方法的查询性能进行比较。由于 PHT 方法只考虑了一维数据的情况，因此我们的实验也暂时只考虑 DRT 和 PHT 在一维数据中的查询性能。实验中，将 2^{16} 的 key 预加到分布式范围树和前缀哈希树上，它们均匀分布到 2^{20} 的 key 空间上，并且限定 DRT 中，每个非叶子节点维护的 key 值最多不能超过 50 个，即 $\tau=50$。实验中，每次分别从 $2^7,2^8,2^9,2^{10},2^{11},2^{12},2^{13},2^{14},2^{15},2^{16}$ 等 10 个不同的范围中随机产生 150 个查询进行实验，测试每次查询所耗费的时间，然后取其平均值作为该查询范围的查询时间。

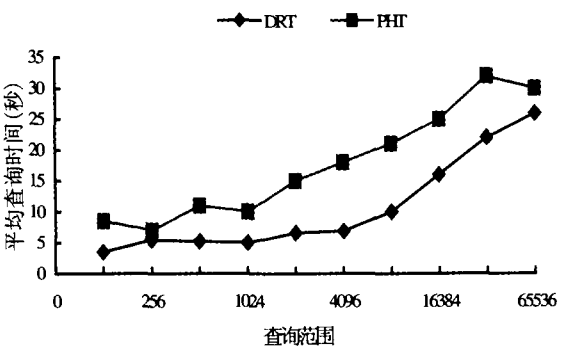

图 2 DRT 和 PHT 在一维数据集上的查询性能比较

图 2 为 DRT 和 PHT 在不同查询范围上的平均查询时间。从实验结果可以看出，在所有的查询范围中，DRT 的查

(下转第 119 页)

变量：$s_0, s_1, p_{61}, t_{20}, t_{28}, t_{29}, t_{31}, t_{32}, t_{33}$.

$DA60.\ p_{62}=s_0+s_1$；$DA61.\ p_{63}=s_0-s_1$；

$DA62.\ u_{t3}=2s_1$；$DA63.\ g_4=u_{12}+s_1$；

$DA64.\ t_{51}=u_{t3}-g_4$；

变量：$s_0, s_1, u_{t3}, p_{61}, p_{62}, p_{63}, t_{20}, t_{28}, t_{29}, t_{31}, t_{32}, t_{33}, t_{51}$.

第十轮

$DM41.\ w=t_{32}p_{61}$；$DM42.\ \omega_i=t_{31}t_{33}$；

$DM43.\ t_{41}=t_{28}p_{62}$ $DM44.\ t_{42}=t_{29}p_{63}$；

$DM45.\ t_{43}=u_{12}s_1$；$DM46.\ p_{70}=s_1^2$；$DM47.\ p_{94}=t_{51}u_{t3}$.

变量：$\omega_i, s_0, s_1, w, u_{t3}, p_{70}, p_{94}, t_{20}, t_{41}, t_{42}, t_{43}, t_{51}$.

$DA65.\ p_{73}=t_{20}-\omega_i$；$DA66.\ p_{76}=s_1-u_{12}$；

$DA67.\ p_{71}=2s_0$； $DA68.\ u_{t2}=p_{70}+p_{71}$；

$DA69.\ p_{69}=u_{11}+s_0$；$DA70.\ g_3=t_{43}+p_{69}$；

$DA71.\ p_{95}=p_{94}+g_3$；$DA72.\ p_{96}=p_{95}-u_{t2}$；

$DA73.\ p_{66}=t_{41}+t_{42}$；$DA74.\ p_{64}=t_{41}-t_{42}$；

$DA75.\ p_{65}=p_{64}/2$；$DA76.\ g_1=p_{65}-t_{43}$.

变量：$\omega_i, w, s_0, g_1, u_{t2}, u_{t3}, p_{66}, p_{73}, p_{76}, p_{96}, t_{51}$.

第十一轮

$DM48.\ g_0=u_{10}s_0$；$DM49.\ p_{72}=u_{t3}s_0$；

$DM50.\ p_{74}=\omega_i p_{73}$；$DM51.\ p_{77}=p_{76}v_{12}$；

$DM52.\ p_{78}=\omega_i u_{12}$；$DM53.\ p_{90}=t_{51}u_{t2}$；$DM54.\ v_{t3}=wp_{96}$.

变量：$\omega_i, s_0, w, g_0, g_1, u_{t2}, u_{t3}, v_{t3}, p_{66}, p_{72}, p_{74}, p_{77}, p_{78}, p_{90}, t_{51}$.

$DA77.\ p_{79}=p_{77}+v_{11}$；$DA78.\ p_{80}=p_{78}+p_{79}$；

$DA79.\ u_{t1}=p_{72}+p_{74}$；$DA80.\ p_{67}=p_{66}/2$；

$DA81.\ p_{68}=p_{67}+u_{10}$；$DA82.\ g_2=p_{68}-g_0$；

$DA83.\ p_{91}=p_{90}+g_2$ $DA84.\ p_{92}=p_{91}-u_{t1}$.

变量：$g_0, g_1, \omega_i, w, s_0, u_{t1}, u_{t2}, u_{t3}, v_{t3}, p_{80}, p_{92}, t_{51}$.

第十二轮

$DM55.\ p_{75}=s_0^2$；$DM56.\ p_{81}=\omega_i p_{80}$；$DM57.\ p_{86}=t_{51}u_{t1}$；

$DM58.\ p_{93}=wp_{92}$；$DM59.\ p_{97}=v_{t3}^2$.

变量：$g_0, g_1, w, u_{t1}, u_{t2}, u_{t3}, v_{t3}, p_{75}, p_{81}, p_{86}, p_{93}, p_{97}, t_{51}$.

$DA85.\ p_{82}=2p_{81}$；$DA86.\ u_{t0}=p_{75}+p_{82}$；

$DA87.\ p_{87}=g_1-u_{10}$ $DA88.\ p_{88}=p_{86}+p_{87}$；

$DA89.\ p_{98}=u_{t3}+p_{97}$；$DA90.\ u_{22}=-p_{98}$；

$DA91.\ t_{61}=2v_{t3}$；$DA92.\ v_{t2}=p_{93}+v_{12}$.

变量：$g_0, w, u_{t0}, u_{t1}, u_{t2}, u_{t3}, v_{t2}, v_{t3}, p_{88}, t_{51}, t_{61}, u_{22}$.

第十三轮

$DM60.\ p_{89}=wp_{88}$；$DM61.\ p_{99}=u_{22}u_{t3}$；

$DM62.\ p_{100}=t_{61}v_{t2}$.

变量：$g_0, w, u_{t0}, u_{t1}, u_{t2}, u_{t3}, v_{t2}, v_{t3}, p_{89}, p_{99}, p_{100}, t_{51}, t_{61}, u_{22}$.

$DA93.\ p_{101}=u_{12}+p_{99}$；$DA94.\ p_{102}=p_{101}+p_{100}$；

$DA95.\ u_{21}=f_5-p_{102}$；$DA96.\ v_{t1}=p_{89}+v_{11}$.

变量：$g_0, w, u_{t0}, u_{t1}, u_{t2}, u_{t3}, v_{t1}, v_{t2}, v_{t3}t_{51}, t_{61}, u_{21}, u_{22}$.

第十四轮

$DM63.\ p_{106}=t_{61}v_{t1}$；$DM64.\ p_{105}=u_{21}, u_{t3}$；

$DM65.\ p_{83}=t_{51}u_{t0}$；$DM66.\ p_{103}=v_{t2}^2$；$DM67.\ p_{104}=u_{22}u_{t2}$.

变量：$g_0, w, u_{t1}, v_{t1}, v_{t2}, v_{t3}, p_{83}, p_{83}, p_{103}, p_{104}, p_{105}, p_{106}, u_{21}, u_{22}$.

$DA97.\ p_{107}=p_{103}+u_{t1}$；$DA98.\ p_{108}=p_{108}=p_{104}+p_{107}$；

$DA99.\ p_{109}=p_{108}+p_{105}$；

$DA100.\ p_{110}=p_{109}+p_{106}$；$DA101.\ u_{20}=f_4-p_{110}$；

$DA102.\ p_{84}=p_{83}+g_0$.

变量：$w, v_{t1}, v_{t2}, v_{t3}, p_{84}, u_{20}, u_{21}, u_{22}$.

第十五轮

$DM68.\ p_{113}=u_{20}v_{t3}$；$DM69.\ p_{112}=u_{21}v_{t3}$；

$DM70.\ p_{85}=wp_{84}$；$DM71.\ p_{111}=u_{22}v_{t3}$.

变量：$v_{t1}, v_{t2}, p_{85}, p_{111}, p_{112}, p_{113}, u_{20}, u_{21}, u_{22}$.

$DA103.\ v_{22}=v_{t2}-p_{111}$；$DA104.\ v_{21}=v_{t1}-p_{112}$；

$DA105.\ v_{t0}=p_{85}+v_{10}$；$DA106.\ v_{20}=v_{t0}-p_{113}$.

变量：$u_{20}, u_{21}, u_{22}, v_{20}, v_{21}, v_{22}$.

（上接第 71 页）

询时间都比 PHT 的查询时间要短。这是由于在 DRT 方法中，可以在非叶节点中存放 key 值，只需找到分配节点之后就可以执行并行查询，而在 PHT 方法中，必须在二叉平衡树中顺序执行，找到所有的叶子节点，然后执行查询，因此，DRT 的耗费的查询时间比 PHT 要少，而且，由于在非叶节点上设置了存放 key 数目的阈值，不会很大地增加存储空间。

结论 在已有的 DHT 结构化 P2P 网络之上支持范围查询依然比较困难。本文提出一种基于分布式范围树的结构化 P2P 范围查询方法。该方法将多维索引的分布式范围树分发到已有的结构化 DHT 覆盖网络中，充分利用 DHT 系统提供的简单数据查找接口，可以有效地实现数据对象的范围查询。一维数据集上不同的查询范围的实验结构表明，DRT 方法所需的查询时间比 PHT 更少。更高维数据集对 DRT 方法的性能影响是我们后续的研究中需要考虑的问题。

参 考 文 献

1 Stoica I, Morris R, Karger D, et al. Chord: A scalable peer-to-peer lookup service for internet applications. In: Proceeding of the ACM SIGCOMM Conference, San Diego, CA, 2001. 149～160

2 Ratnasamy S, Francis P, Handley M, et al. A scalable content-addressable network. In: Proceeding of the ACM SIGCOMM, San Diego, CA, 2001. 161～172

3 Rowstron A, Druschel P. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. LNCS, 2001, 2218: 329～350

4 Zhao B Y, Kubiatowicz J D, Joseph A D. Tapestry: An infrastructure for fault-tolerant wide-area location and routing. Berkeley Computer Science Division, University of California, CA: [Technical Report CSD-01-1141]. 2001

5 Harren M, Hellerstein J M, Huebsch R. Complex Queries in DHT-based Peer-to-Peer Networks. In: Proceeding of IPTPS02, Cambridge, USA, 242～259

6 Triantafillou P, Pitoura T. Towards a Unifying Framework for Complex Query Processing over Structured Peer-to-Peer Data Networks[C]. In: Proceedings of the First International Workshop on Databases, Information Systems and Peer-to-Peer Computing (DBISP2P), Berlin, Germany, 2003. 169～183

7 Chawathe Y, Ramabhadran S, Ratnasamy S, et al. A case study in building layered DHT applications. In: Proceedings of the ACM SIGCOMM, Philadelphia, Pennsylvania, USA, 2005. 97～108

8 Bharambe A R, Agrawal M, Seshan S. Mercury: Supporting scalable multi-attribute range queries. In: Proceedings of the ACM SIGCOMM, Portland, USA, 2004. 353～366

9 Harvey N, Jones M, Saroiu S, et al. SkipNet: A Scalable Overlay Network with Practical Locality Properties. In: Proceedings of the Fourth USENIX Symposium on Internet Technologies and Systems, Seattle, WA, March 2003

10 周培德. 计算几何----算法分析与设计. 清华大学出版社, 2000

11 de Berg M, van Kreveld M, Overmars M, Schwarzkopf O. Computational Geometry: Algorithm and Application, second, revised edition. Springer-Verlag, Berlin, 2000

12 Rhea S, Godfrey B, Karp B, et al. OpenDHT: a public DHT service and its uses. In: Proceedings of the ACM SIGCOMM, 2005