文章编号：1000-8608(2011)03-0433-07

基于区位码和距离的 Chord 网络高维数据范围检索

孟宪福*[1]， 孟泓汐[2]， 张振强[1]

（1. 大连理工大学 计算机科学与技术学院，辽宁 大连 116024；
2. 大连海事大学 信息科学技术学院，辽宁 大连 116026）

摘要： 结构化对等网络对数据的范围检索等相似查询缺乏有效的支持. 由于利用 iDistance 索引进行范围查询时会引入很多“误中点”，提出了一种利用区位码和距离来建立高维数据索引的方法. 该方法首先利用 Code-Distance 技术来建立高维数据的一维索引，并采用位置保持哈希函数给予每个索引值一个唯一标识，该标识将被保存在 Chord 环节点上，从而生成 BM-Chord 系统. 同时，详细叙述了数据过滤技术和范围查询策略. 模拟实验结果表明，BM-Chord 系统在减小中间结果集大小和提高查全率等方面是有效的.

关键词： P2P；范围检索；iDistance；位置保持哈希函数；区位码
中图分类号： TP311 **文献标志码：** A

0 引 言

随着 P2P 技术的发展，P2P 网络已经在多个领域得到实际应用，比如资源共享等，而在 P2P 环境下进行范围检索已成为用户的迫切需要.

Chord 是由环形组成的结构化 P2P 网络，它所具有的良好的数据检索效率、容错性和自适应性等，使其已成为具有高可扩展性和高效率的实用结构化 P2P 模型[1]. 但由于 Chord 是基于 DHT(distributed hash table)的，并不能有效地支持范围检索等相似性检索.

在已有的研究中，对高维数据进行范围查询所采用的基本方法是降维策略，文献[2]提出了被称为 M-Chord 的网络结构，其中采用的索引技术是 iDistance[3]，即通过将高维数据降至一维，然后采用位置保持哈希技术[4]将其映射到 Chord 网络的对应节点上来实现高维数据检索. 然而，由于在查询时这种结构只对索引值进行一次过滤，将产生大量“误中点”[5]，增加了对冗余信息进行精简的时间.

文献[6]采用了基于代表点空间划分策略的一种 Chord 网络结构，该结构利用事先确定的代表点将整个数据空间划分为数个子空间，每个代表点仅对应一个子空间，在此基础上，将代表点映射到一维区间上，并将代表点的一维索引标识作为 Chord 网络中节点的哈希值. 由于是基于准随机序列的均匀分布特性来选取代表点的，对于那些不均匀的数据集，在进行范围检索时会产生大量的“误中点”，影响了检索效率.

文献[7]利用 Hilbert 曲线将高维数据映射到 Chord 环上，文献[8]则给出了以空间填充曲线为基础的范围划分方法，这几种方法都能在 Chord 网络上实现高维数据的范围检索. 但是，这些方法对低维度数据可行，对高维度数据会出现“维度灾难”问题，不能很好地适应数据维度的增长.

VA-file 近似向量索引方法[9]将数据空间划分为 2^b 个长方形单元格，对每个单元格利用长度为 b 的字符串序列进行标识，对全部的高维点数据利用所属单元格标识符进行近似表示，并通过顺序扫描近似向量来实现范围检索. 显然，VA-file 方法的搜索效率在很大程度上受数据分布的

收稿日期： 2009-09-13； **修回日期：** 2011-04-09.
作者简介： 孟宪福*(1960-)，男，副教授，E-mail：xfmeng@dlut.edu.cn.

影响. 与 VA-file 方法不同,文献[10]所提出的区位码近似向量压缩技术是利用与参考点的相对位置关系将多维向量数据近似表示为位码字符串,在检索时通过比较查询点与某个点之间的不相同位码的位数,来判断该点是否为候选数据点.

基于区位码的优化高维索引结构[11]不仅利用 iDistance 距离来表示对象点和参考点之间的远近关系,同时利用区位码来近似表示它们之间的位置关系,以便将多维向量压缩为二维向量.

本文在 Chord 网络协议和已有的高维数据索引算法的基础上,提出 BM-Chord 网络结构,以解决高维数据的范围检索问题. BM-Chord 网络结构采用 iDistance 和区位码向量压缩两种技术将高维数据降为一维,然后利用位置保持哈希函数将一维索引值映射到 Chord 网络环上,环上的每个节点利用 B+树结构来保存数据索引值. 在此基础上,利用本文提出的检索算法和过滤技术,实现基于 Chord 网络的多维数据的范围检索.

1 空间的划分策略

一般来讲,可以事先确定 Chord 网络中可容纳的最大节点个数 M,这样,为了确保每个节点至少拥有一个子空间,就需要将整个数据空间划分为不小于 M 的子空间. 因为 iDistance 索引技术是将数据空间划分为 N 个聚类,而 N 又小于 M,因此不能直接利用;前面介绍的基于区位码的向量压缩技术是将数据空间划分为 $2^{d'}$ 个子分区. 如果将这两种方法结合起来,则可将数据空间划分为 $N2^{d'}$ 个子分区,这样就能满足每个节点至少有一个子分区索引值的要求.

1.1 iDistance 索引技术

iDistance 是采用与参考点之间的距离将高维数据点向量转换为一维数据表示的,为了做到这一点,该技术提供了非常灵活的空间划分策略和参考点选取方法.

首先,该方法将数据空间划分为 N 个聚类,同时确定每个聚类 C_i 的中心点 $c_i(0 \leqslant i < N)$. 对于每一个元素 $x \in D$(这里 D 是 $[0,1]^d$ 空间,d 是空间维度),则依据 x 与聚类中心点的距离为其指定一个一维的距离值,该距离值的计算如下:

$$iDis(x) = d(c_i, x) + i * C \tag{1}$$

其中函数 d 是欧几里德距离;C 是一个足够大的数,能够将属于聚类 C_i 内的所有数据点都映射到区间 $[i*C,(i+1)*C)$ 上,同时保证各个区间之间不相交.

1.2 基于区位码的向量压缩方法

区位码是利用与参考点的相对位置将多维数据空间划分为 $2^{d'}(d' < d)$ 个子分区,每个子分区利用一个位码字符串来表示,这里,每个高维数据点都可以被近似地表示为所属分区的位码字符串,d 是多维数据的维度,d' 是位码字符串的长度.

对于任意高维数据点 $P(p_1, p_2, \cdots, p_d)$,参考点是 $O(o_1, o_2, \cdots, o_d)$,那么,P 的位码就可以表示为 $B_P(b_{p_1}, b_{p_2}, \cdots, b_{p_{d'}})$,这里

$$b_{p_j} = \begin{cases} 1; & p_j \geqslant o_i,\ 1 \leqslant j \leqslant d' \\ 0; & \text{其他} \end{cases} \tag{2}$$

1.3 基于 Code-Distance 的索引结构

如果利用 iDistance 索引方式,将使得不同的高维数据拥有相同的一维索引值,这种现象将随着维度的增高越来越明显,导致在查询过程中产生大量的“误中点”,从而增加了距离计算的次数,导致较低的检索效率. 因此,利用区位码向量压缩技术进行第 2 次过滤,将能有效地改善范围检索的性能. 下面对这两种索引方法进行如下扩展.

首先,假设 $P(p_1, p_2, \cdots, p_d)$ 所属聚类 C_i 的质心为 $C_i(c_{i1}, c_{i2}, \cdots, c_{id})$,这里 $0 \leqslant i < N$. 以点 C_i 为参考点,将聚类 C_i 在 d' 维上划分为 $2^{d'}$ 个子分区,假设将点 P 的位码表示为 $B_P(b_{p_1}, b_{p_2}, \cdots, b_{p_{d'}})$,则利用 Code-Distance 表示的点 P 的索引值为

$$CD(P) = i * C + Dec(B_P) * C' + d(P, C_i) \tag{3}$$

其中 $Dec(B_P) = 2^0 \times b_{p_1} + 2^1 \times b_{p_2} + \cdots + 2^{d'-1} \times b_{p_{d'}}$,$C = 2^{d'} \times C'$,$C' = \sqrt{d}$.

由于 $[0,1]^d$ 空间上最大的 $d(x,y)$ 值($x, y \in D$)为 $\sqrt{d}$,常数 C' 可用于将位于不同分区的距离值区分开;由于 $Dec(B_P)$ 的最大值为 $2^{d'} - 1$,这

样，常数 C 就可以将不同聚类内的 Code-Distance 索引值区分开. 显然，这里的一维索引值既包含了前面提到的区位码的信息，又包含了距离的信息，具有基于近似向量的压缩和高维向一维转化两种方法的优点.

2　BM-Chord 网络

在上面介绍的空间划分策略基础上，本章将详细介绍基于该策略的 BM-Chord 网络的建立及其范围查询过程.

2.1　基本思想

一般来讲，在将高维数据映射到 Chord 节点上时，需要首先将高维数据映射到一维空间上. 下面介绍利用 Code-Distance 技术，通过聚类、iDistance距离和区位码，将高维数据空间映射到一维线性空间上的过程.

由于 Chord 是一种动态网络，本文利用基于网格的聚类生成方法，来生成 N 个代表点，同时将每个代表点作为 Voronoi 图[12]的“核子”，这样，就可以将整个数据空间划分为 N 个子空间(即聚类). 在 Chord 环中每个节点上都保存这 N 个代表点的全局信息，同时将节点上的高维数据对象分配到每个子空间中.

在上面操作的基础上，利用位置保持哈希函数将所得到的 Code-Distance 索引值投影到 $[0, 2^m)$区间上，同时，为每个网络的每个节点分配一段连续的索引值空间，而且每个节点所负责的索引范围不相交，整个节点所负责的索引范围的总和就是整个索引空间.

BM-Chord 网络在逻辑上涵盖了所有可直接寻址的节点，所有数据也存储在这些节点上. 由这些节点所组成的结构能够比较容易地将数据元素插入到网络中，节点间通过消息进行通信，利用范围检索来获取所需的检索数据.

2.2　节点的结构

BM-Chord 网络的逻辑拓扑完全符合 Chord 结构，利用位置保持哈希函数和式(3)，就可以为每个数据点分配一个属于 BM-Chord 范围(即$[0, 2^m)$区间)的标识符. 为了能够在节点间对数据进行划分，网络中的任一个节点 N_i 也需要分配一个属于 BM-Chord 域的值 K_i. 这样，下述内容就组成了节点 N_i 的数据结构：

(1) Chord 网络的路由信息 K_i、节点的前驱和后继节点信息以及指针表；

(2) 利用 B＋树结构存储的$(K_{i-1}, K_i]$ $(\mathrm{mod}\ 2^m)$区间的索引值；

(3) 在节点上保存的全局的索引信息.

2.2.1　*索引信息的构建*　与 Chord 网络相似，在 BM-Chord 网络中，其索引信息也是采用分布式管理的. 为了减少在检索过程中的消息量，在每个节点上也都存储着全局的索引信息，而高维数据的 Code-Distance 索引值将利用 BM-Chord 网络中的每个节点进行分散管理. 同时，采用主成分分析法来确定在哪些维度上需要进行区间划分，以应对数据分布的不均匀性问题.

(1) 全局索引的构建

以相同的参数为每个节点生成相同的代表点，并以这些代表点作为 Voronoi 图的“核子”，每个节点利用这些代表点来构造一棵势点树(vantage point tree，VP-tree)[13]，以此作为全局索引.

在每个节点上设置全局索引主要是考虑如下两种应用：

一是为节点分配索引值. 当一个节点加入网络时，此节点可以根据全局索引查询到其上每个数据对象所在的单元格，也就是其所在的聚类，然后根据 Code-Distance 算法为每个数据对象分配索引值，并将其映射到 Chord 环中的对应节点上.

二是确定在进行相似查询时需要访问的代表点. 在给定一个范围查询的情况下，通过检索全局索引，查询请求节点就可以确定哪些聚类与该相似查询的搜索区域相交. 也就是说，凡是与相似查询的搜索区域相交的聚类都有可能包含所需要的结果，因此，检索请求必须要访问这些聚类，以获取结果. 而以代表点为搜索键，就可以将查询请求路由给这些代表点所属的节点，从而实现范围检索.

(2) 局部索引的构建

将网络中已被分配 BM-Chord 标识符的节点记为 active,还没有被分配标识符的节点记为 inactive.

对于任一个数据 $x \in I$(I 为高维数据的集合),任一个节点 N_{ins} 都可以发出插入请求,其基本过程如下:节点 N_{ins} 利用式(3)计算 $CD(x)$ 的值,如果 N_{ins} 是 inactive 节点,就将请求转发给它所了解的下一个节点;如果 N_{ins} 是 active 节点,就遵照 Chord 网络协议,将请求向前转发给节点 $N_{CD(x)}$,这里 $N_{CD(x)}$ 是管理 $CD(x)$ 值的节点,由此节点根据 $CD(x)$ 值将 x 插入到它所在的 B+树中.

2.2.2 节点的加入及其退出处理过程 第一个加入网络的节点将被分配一个索引值 2^m-1,该索引值涵盖了整个 BM-Chord 域$[0,2^m)$,同时,这第一个入网的节点状态被设置为 active;除了第一个节点,其他的节点在开始时不被分配 BM-Chord 标识符,它们的状态被设置为 inactive.

假如存在一个处于 inactive 状态的节点 N_{new},则任何一个处于 active 的节点都可以对索引值发出分裂的请求.一般来讲,网络中的每个节点都可以确定自己的索引分裂准则,这些分裂准则的定义可以考虑存储容量以及计算负载等因素.假如 N_i 节点发出分裂请求,同时$(K_{i-1},K_i]$为 N_i 节点所负责的索引值区间,则索引的分裂过程如下:

(1) 以 $K_{i-1}<K_{\text{new}}<K_i$ 为前提条件来生成索引值 K_{new},将所生成的 K_{new} 分配给节点 N_{new};

(2) 按照 Chord 网络中的节点加入原则,N_i 节点将位于$(K_{i-1},K_{\text{new}}]$内的数据迁移到 N_{new} 节点上.

显然,如何选择索引值 K_{new} 是一个需要考虑的重要问题,下面详细介绍其选择方法.

如果 $(K_{i-1},K_i]$ 覆盖了 z 个分区,这里 $z>1$,则选择 K_{new} 作为分区的边界值,使 $(K_{i-1},K_{\text{new}}]$ 覆盖$\lfloor z/2 \rfloor$个分区;如果$(K_{i-1},K_i]$只覆盖了一个分区或一个分区的一部分,则选择 $K_{\text{new}}=(K_{i-1}+K_i)/2$,也就是将 $(K_{i-1},K_i]$ 进行平分.

假如有一个处于 active 状态的节点离开了网络,那么它将自动成为 inactive 节点,在此情况下,依据标准 Chord 网络中的节点退出原则,退出节点上所保存的所有标识符范围,将其转移给其后继节点进行管理.

2.3 候选数据的过滤机制以及范围检索算法

在给出过滤机制和检索策略之前,先给出后续使用的相关定义.设 I 是 D 的高维数据索引的有限子集,高维数据的维度是 d.给定高维数据点 $Q \in D$ 以及查询半径 r,范围查询 $Range(Q,r)$ 搜索到的结果集合是 $S=\{x \in I \mid d(Q,x) \leqslant r\}$.

2.3.1 数据过滤机制 对于给定范围检索请求 $Range(Q,r)$,则将与检索范围相交的聚类称为候选数据类.由 *canCluster*[14] 可知,如果对由代表点 $Rep=(C_1,C_2,\cdots,C_N)$ 所构成的 VP-tree 按照下述准则进行查询,则所求得的相交单元格,就是候选数据类集合.

准则 1

(1) 如果 $d(Q,R) \leqslant r$,则返回 R(R 为 VP-tree 最顶层的势点);

(2) 如果 $d(Q,R)+r \geqslant M$,递归查找右子树;

(3) 如果 $d(Q,R)-r \leqslant M$,递归查找左子树.

根据上述准则,就可以得到候选数据类 *canCluster*,根据 *canCluster* 中的每个聚类就能够求出与查询范围相交的连续区间,该区间就组成了需要搜索的 Code-Distance 索引范围的一部分.

仅仅获得候选数据类还是不够的,对于每个候选的 C_j 数据类,还需要求出与检索范围相交的候选分区,该候选分区的获取过程如下.

准则 2 对于所给定的范围查询请求 $Range(Q,r)$,其候选分区的计算过程如下:

假设点 $Q(q_1,q_2,\cdots,q_d)$ 的区位码为 $S(s_1,s_2,\cdots,s_{d'})$,则候选数据类 C_j 内与该查询范围相交的分区 $T(t_1,t_2,\cdots,t_{d'})$ 必满足

$$t_i=\begin{cases} s_i; & |q_i-C_{ji}| \geqslant r,\ 1 \leqslant i \leqslant d' \\ 0 \text{ 或 } 1; & \text{其他} \end{cases} \tag{4}$$

接下来确定候选分区内需要搜索的索引值范围.

对于范围查询 $Range(Q,r)$，若候选聚类 C_j 内分区 $T(t_1,t_2,\cdots,t_{d'})$ 与该查询范围相交，则 T 内需要搜索的 Code-Distance 索引值范围为

$$[j\times C+Dec(T)\times C'+d(Q,C_j)-r,$$
$$j\times C+Dec(T)\times C'+d(Q,C_j)+r]$$

2.3.2　范围检索策略　在上述过滤机制的基础上，下面给出完整的高维数据范围检索策略.

对于由任一节点 A 发起的范围检索请求 $Range(q,r)$，其检索过程如下：

(1) 首先，利用准则 1 来获取候选数据类 $C_i(1\leqslant i\leqslant N)$，并根据准则 2 来获取相交分区 $p_j(0\leqslant j\leqslant(N-1)*\ 2^{d'}-1)$，然后，根据上文 Code-Distance 索引值范围来确定相交分区的检索范围 $I_j=[h_{\text{low}},h_{\text{high}}]$；

(2) $\forall j,0\leqslant j\leqslant(N-1)*2^{d'}-1$，节点 A 发送消息 $SEARCHINTERVAL(I_j)$ 给节点 N_{I_j}，其中 $N_{I_j}=N_{h((h_{\text{low}}+h_{\text{high}})/2)}$，即为管理索引值 $h\left(\frac{h_{\text{low}}+h_{\text{high}}}{2}\right)$ 的节点；

(3) 节点 A 等待所有的回复信息，并依据所回复的 IP 从相关节点获取高维数据对象，通过消冗处理后，生成最终的检索结果集合 S_A，$S_A=\{y\mid d(y,Q)\leqslant r,CD(y)\in S_1\}$.

在节点 N_{I_j} 上执行 $SEARCHINTERVAL(I_j)$ 范围检索请求的处理过程如下：

① 如果在 N_{I_j} 节点上没有保存完整的 I_j 区间，则根据检索范围向其前驱节点或后继节点发送范围检索请求 $SEARCHINTERVAL\ (I_j)$；

② 确认保存在自己节点上的数据信息，并据此生成本地结果集 $S_1=\{x\mid CD(x)\in I_j\}$；

③ 将本地生成的结果集 S_1 返回给节点 A，并通知节点 A 准备接收从其前驱节点或后继节点返回的检索结果.

3　模拟实验与结果分析

本实验是在 Peersim 开源环境下完成的，以确认 BM-Chord 网络对高维数据范围检索的有效性. 所采用的实验数据包括两种：一种是人工生成的模拟数据，另一种是真实数据. 为了验证本文策略的有效性，将本实验与采用代表 iDistance 索引的 M-Chord 网络进行比较.

3.1　中间结果集比较

所谓中间结果集，是指对于给定的范围检索，仅采用索引算法所能得到的查询结果. 显然，在不降低查全率的前提下，中间结果集越小越好.

(1)中间结果集与原始数据集

将网络节点个数设置为 2 000，并分别利用 10 000、20 000、30 000 和 40 000 幅图像的 32 维颜色直方图数据进行实验. 设聚类数为 30，范围检索参数为查询中心点(0,0,…,0)，查询半径 $r=0.25$.

由于 M-Chord 网络是采用 iDistance 索引方法，在查询过程中仅进行一次数据过滤；而 BM-Chord 网络利用两种索引技术的特点，在检索过程中进行两次数据过滤，这样，在不降低查全率的前提下能有效减少中间结果集中的数据个数 n_{int}，其结果如图 1 所示(图中 n_{ori} 为原始数据集数据个数).

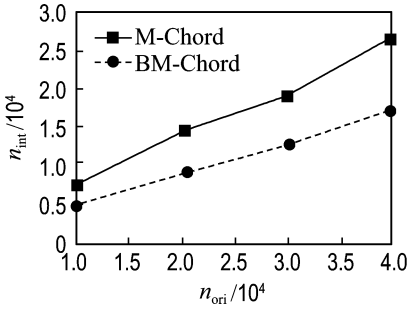

图 1　中间结果集与原始数据集数据个数

Fig. 1　Size of intermediate result set and original data set

(2)中间结果集与维度

将网络中的节点数设置为 2 000，利用 30 000 幅 32 维图像颜色直方图数据和随机生成的 30 000 个 10、15、20 和 25 维数据进行实验，并设聚类个数为 30.

由图 2 可知，由于 M-Chord 网络采用一次数据过滤方法，它比采用两次过滤技术的 BM-Chord 网络所产生的“误中点”数据更多(图中 d 为维度). 这说明本文 BM-Chord 网络在不降低查全率的前提下，能有效减少检索结果中的“误中点”数据.

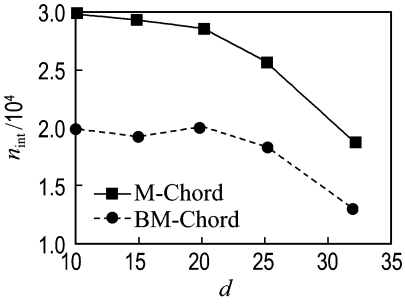

图 2 中间结果集数据个数与维度

Fig. 2 Size of intermediate result set and dimensionality

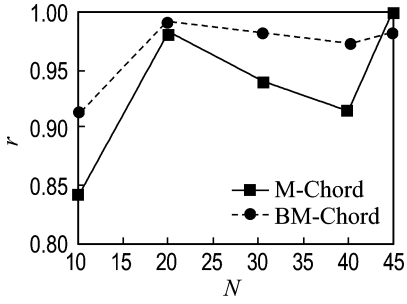

图 4 查全率与聚类数

Fig. 4 Recall ratio and number of clusters

3.2 查全率比较

(1)查全率与维度

网络节点数设置为 2 000,利用 30 000 幅 32 维图像的颜色直方图数据以及随机生成的 30 000 个 10、15、20 和 25 维数据进行实验,设聚类数为 30.

由图 3 知,M-Chord 网络和 BM-Chord 网络的查全率 r 都很高,但 M-Chord 网络对真实数据的查全率明显降低,而 BM-Chord 网络对于不同维度的数据其查全率变化不大.

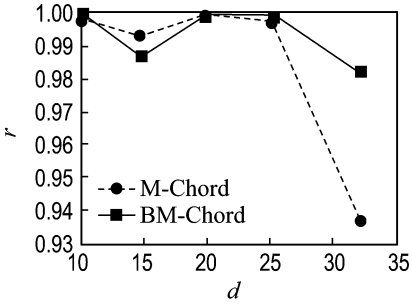

图 3 查全率与维度

Fig. 3 Recall ratio and dimensionality

(2)查全率与聚类数

将网络节点数设置为 2 000,利用 30 000 幅 32 维图像的颜色直方图数据进行实验,并分别将聚类数 N 设置为 10、20、30、40 和 45.

从图 4 可以看出,查全率受聚类数的影响较大,但在索引生成和范围查询过程中如何来选择恰当的聚类数是一个很困难的问题.该问题将在以后的研究过程中进行重点研究.

4 结 语

本文结合一维转换和近似向量两种思想,提出了基于 iDistance 和区位码索引的 BM-Chord 系统,实现了 Chord 环境下的高维数据范围检索.利用两次过滤大大减少了中间结果数据个数,提高了查全率.如何减少消息转发次数以降低网络传输负载以及在保证查全率的前提下如何进一步减少"误中点"的个数将是今后的研究重点.

参考文献:

[1] STOICA I, MORRIS R, KARGER D, *et al*. Chord: A scalable peer-to-peer lookup service for internet applications [C] // **Special Interest Group on Data Communication (ACM SIGCOMM)**. San Diego: ACM, 2001:149-160

[2] NOVAK D, ZEZULA P. M-Chord: A scalable distributed similarity search structure [C] // **First International Conference on Scalable Information Systems (INFOS-CALE 2006)**. Hong Kong: ACM, 2006:1-10

[3] JAGADISH H V, OOI B C, TAN K L, *et al*. iDistance: An adaptive B+ — tree based indexing method for nearest neighbor search [J]. **ACM Transactions on Database Systems**, 2005, **30**(2):364-397

[4] INDYK P, MOTWANI P R, RAGHAVAN P, *et al*. Locality-preserving hashing in multidimensional spaces [C] // **ACM Symposium on Theory of Computing**. El Paso: ACM, 1997:618-625

[5] 梁俊杰，杨新泽，冯玉才. 大规模高维向量空间的快速范围查询[J]. 小型微型计算机系统，2007，**28**(7)：1225-1229

[6] 徐林昊，周傲英. 结构化对等计算系统中的高维相似搜索[J]. 计算机学报，2006，**29**(11)：1982-1994

[7] SCHMIDT C，PARASHAR M. Flexible information discovery in decentralized distributed systems [C] // **The 12th IEEE International Symposium on High Performance Distributed Computing**. Washington：IEEE Computer Society，2003：226-235

[8] GANESAN P，YANG B，GARCIA-MOLINA H. One torus to rule them all：multi-dimensional queries in P2P systems [C] // **The 7th International Workshop on the Web and Databases**. New York：Stanford University，2004：19-24

[9] WEBER R，SCHEK H，BLOTT S. A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces [C] // **The 24th International Conference on Very Large Data Bases**. New York：Morgan Kanfmann，1998：194-205

[10] CUI B，SHEN H，SHEN J，*et al*. Exploring bit-difference for approximate KNN search in high-dimensional databases [C] // **The 16th Australasian Database Conference**. Australia：Australian Computer Society，2005

[11] 梁俊杰，冯玉才. BC-iDistance：基于位码的优化高维索引 [J]. 小型微型计算机系统，2007，**28**(11)：1647-1651

[12] AURENHAMMER F. Voronoi diagrams-A survey of a fundamental geometric data structure [J]. **ACM Computing Survey**，1991，**23**(3)：345-405

[13] YIANILOS P N. Data structures and algorithms for nearest neighbor search in general metric spaces [C] // **The 4th Annual ACM-SIAM Symposium on Discrete Algorithms**（**SODA′1993**）. Philadelphia：Association for Computing Machinery，1993：311-321

[14] 任晓娱. 利用分区和距离实现 Chord 中高维数据范围检索[D]. 大连：大连理工大学，2009

Bit-code and distance-based range query for high-dimensional data in Chord network

MENG Xian-fu[*1]，MENG Hong-xi[2]，ZHANG Zhen-qiang[1]

（1. School of Computer Science and Technology，Dalian University of Technology，Dalian 116024，China；
2. Information Science and Technology College，Dalian Maritime University，Dalian 116026，China）

Abstract：A structured P2P network is not suitable for similarity search，such as range query. Since a lot of false hits may be obtained by using iDistance indexing technique，a high-dimensional data indexing algorithm based on bit-code and distance is proposed. The Code-Distance technique and locality-preserving hashing function are utilized to respectively construct one-dimensional index from data with high-dimension and assign a unique identifier for each index，and those identifiers are kept into nodes to establish BM-Chord network. The data filtering technique and range query strategy are described in detail as well. The simulation results testify the effectiveness of BM-Chord network in terms of the intermediate data set and recall ratio.

Key words：P2P；range query；iDistance；locality-preserving hashing function；bit-code