一种基于 LSH 面向二元混合类型数据的相似性查询方法

朱命冬[1),2)] 申德荣[1)] 寇 月[1)] 聂铁铮[1)] 于 戈[1)]

[1)](东北大学计算机科学与工程学院 沈阳 110819)

[2)](河南工学院计算机科学与技术系 河南 新乡 453003)

摘 要 局部敏感哈希方法(LSH)已经被广泛用于高维数据和大规模数据集的最近邻查询,然而现有方法大多将LSH方法用于单一类型的数据,文中尝试将LSH方法用于二元混合类型数据,如图像-文本数据,空间-文本数据等.文中提出了一种基于LSH混合索引结构的相似性查询方法,该方法可有效地管理含两种数据类型的数据,并且融合两种数据类型的相似性进行最近邻查询.文中提出的查询方法主要有三个特点:首先,结合LSH方法为混合数据构建混合哈希值,该混合哈希值保留有数据对象之间内容相似性的信息,基于混合哈希值构建哈希索引,进行快速准确的最近邻查询;其次,该方法解决传统LSH方法固定敏感半径的问题,可以有效地处理可变查询范围的相似性查询;最后,该方法在分布式环境中不需要全局索引信息,保证分布式查询的伸缩性.文中通过理论分析证明了查询方法和查询算法的准确性和有效性,进一步通过分布式系统优化及基于真实数据和合成数据的大量实验验证了方法的伸缩性和高效性.

关键词 二元混合数据;相似性查询;局部敏感哈希;分布式查询算法;最近邻查询

中图法分类号 TP391 **DOI 号** 10.11897/SP.J.1016.2018.01827

A LSH-Based Method for Similarity Queries on Binary Hybrid Data

ZHU Ming-Dong[1),2)] SHEN De-Rong[1)] KOU Yue[1)] NIE Tie-Zheng[1)] YU Ge[1)]

[1)](*College of Computer Science and Engineering, Northeastern University, Shenyang* 110819)

[2)](*Department of Computer Science and Technology, Henan Institute of Technology, Xinxiang, Henan* 453003)

Abstract Locality Sensitive Hashing (LSH) has been widely used in high-dimensional data and large-scale datasets for nearest neighbor queries, but LSH is mostly applied for data with a single data type, so this paper attempts to adapt LSH to binary hybrid data, such as data with image and text, data with location and text, etc. On the one hand, an effective index is difficult to be built on binary hybrid data, which are normally high-dimensional data consisting of different data structures. On the other hand, for different data types, distance functions are different. Such as Euclidean distance function for numeric data, Jaccard distance function for character data, etc., soflexible similarity queries are difficult to efficiently processed on binary hybrid data, which should consider similarity of two data types simultaneously. To this end, this paper proposes a generalized LSH based method for similarity queries on binary hybrid data. The method effectively manages data consisting of two different data types, and processes nearest neighbor queries by considering similarity of both data types. The method mainly has three features: Firstly, an LSH based hybrid hash value, which conserves similarity among data objects, is constructed

收稿日期:2016-05-05;在线出版日期:2017-02-06.本课题得到国家"九七三"重点基础研究发展规划项目基金(2012CB316201)和国家自然科学基金(61472070,61672142)和河南省高等学校重点科研项目计划(18B520011)资助.**朱命冬**,男,1985 年生,博士,讲师,中国计算机学会(CCF)会员,主要研究方向为大数据分析和相似性查询处理.E-mail:zhumingdong@hait.edu.cn.**申德荣**,女,1964 年生,教授,博士生导师,主要研究领域为 Web 数据处理和分布式数据库.**寇 月**,女,1980 年生,副教授,主要研究方向为实体识别和 Web 数据管理.**聂铁铮**,男,1980 年生,副教授,主要研究方向为数据质量和数据集成.**于 戈**,男,1962 年生,博士,教授,中国计算机学会(CCF)会士,主要研究领域为分布式和并行数据库管理、OLAP 和数据仓库、数据集成、图数据管理等.

for each data object, and then a hash index is built on hybrid hash values for processing nearest neighbor queries efficiently. Specifically, one single type hash value for each data type is generated by the corresponding LSH function, and then the hybrid hash value is built by concatenating two hash values of single data type. Intuitively similar objects have close hybrid hash values, so that it is with high probability that similar objects are stored in close hash buckets and can be found with few disk I/O in processing nearest neighbor queries. The lengths of the two hash values are judiciously chosen to make nearest neighbor query correct and effective. Secondly, the method overcomes the limit of traditional LSH based methods which have fixed sensitive query ranges, and effectively deals with the variable query ranges of similarity queries. The sensitive radius of LSH is fixed while the range of queries may vary from user to user and query to query. For instance, if the sensitive range of LSH is 300 m, the LSH can only process queries with query ranges as 300 m, however similarity queries of users may differ from one another in terms of both data types. In such a case, our method can process the queries with larger ranges such as 350 m, 400 m or 500 m. Finally, the method does not require global index information, then guarantee scalability in a distributed environment. Our algorithms depend on little global information, so they are easier to be maintained on a share nothing architecture. In addition, similar objects are prone to be in a bucket which is stored in one compute node, the communication cost of queries can be significantly reduced. Theoretical analysis shows the accuracy and effectiveness of the method and algorithms. Furthermore, distributed system optimization and extensive experiments on real data and synthetic data validate scalability and efficiency of the method.

Keywords binary hybrid data; similarity query; locality sensitive hashing; distributed query algorithm; nearest neighbor query

1 引 言

随着计算机技术和移动互联网的发展,互联网数据呈现类别多样化和类型复杂化的趋势.一个数据对象的描述信息会包含有多种数据类型.例如,在线电子商务平台中(淘宝、京东、亚马逊等),一个商品的描述中会包含有商品的图片和商品的文字描述.基于位置的服务(LBS)的应用中(美团、糯米、点评网等),一条推广内容往往包含有商家的地理位置信息和促销的文本信息.相似性计算是数据管理和数据挖掘中的一个基本操作,对这些混合类型的数据的相似性查询处理可以给用户提供更丰富的信息和更好的用户体验.例如,通过对包含有图片和文本的数据(简称为图片-文本数据)进行相似性查询可以返回给用户视觉上相似和文本描述相关的商品;通过对包含有空间位置和文本的数据(简称为空间-文本数据)进行相似性查询处理则可以返回位置相近且满足特定关键词需求的促销信息.

对大规模的混合类型数据进行相似性查询,难点之一是如何构建索引提高查询效率,从而避免在大数据集中进行对对比较.对于单一数据类型可以根据具体数据特征构建索引.例如,对于图片可以抽取图片特征构建特征向量进而基于特征向量构建树型索引或哈希索引等[1-3];对于空间数据可以构建R-tree索引或者Grid索引等[4-5];对于文本数据可以根据文本的特征构建倒排索引或者Trie索引等[6-7].然而,对混合数据类型构建索引则需要同时考虑多种数据特征.对于空间文本数据,很多研究工作尝试将分别针对空间和文本的索引整合在一起.例如,文献[8]提出一种空间文本混合索引IR-tree.IR-tree将空间距离近的数据对象聚集在一起,每个非叶子索引节点汇总子节点的文本信息,每个叶子节点包含一个倒排索引.文献[9]提出了一种基于区域的面向空间文本数据的相似性查询方法SEAL.SEAL采用了基于倒排的"过滤-确认"框架,在过滤阶段,SEAL将空间信息转化为基于网格的特征并采用阈值敏感的过滤算法.对于图片文本数据,文献[10]将带有图片和文档统一映射到维基百科的概念空间中,每个数据对象可以用概念向量表示.当有查询时,通过查询对象的最近邻资源计算出查询的概念向量,基于该概念向量查询出相似结果.M^3R[11]

是一个监督的多模态互主题加强模型(multi-modal mutual topic reinforce modeling),该模型使用一个联合的跨模态概率图模型发现不同模态之间的关联,构建查询模型.

基于以上分析可以看出,面向混合类型数据的相似性查询已经有了一定的研究成果,但现有方法仍然有一定的局限性,首先,在处理高维数据时性能较差.显然图片的特征信息和文本数据都属于高维数据,"维数灾难"使得基于层次结构或树形的索引结构在处理高维数据时性能较差;其次,在大数据背景下,处理大规模数据时为了提高伸缩性,往往需要分布式管理方法.在分布式环境下,现有方法需要维护一定的全局信息,降低了系统的可伸缩性.

为了解决这些问题,本文尝试使用 LSH(局部敏感哈希方法)来进行面向二元混合数据的相似性查询,LSH 方法的优势有:(1)混合类型数据本身一般具有高维特性,基于 LSH 的方法处理高维数据时具有较好的性能.例如,从图片抽取的特征往往包括颜色、纹理、局部内容特征、全局部内容特征等,单图片的 SIFT 特征描述子[12]就可达到 128 维;对于文本,一条微博的文本长度限制为 140,去掉停用词并且分词后也可能达到 100,可见混合类型数据(图像文本、空间文本)一般具有高维特性;(2)基于 LSH 的方法不需要维护全局信息,便于在分布式环境中实现;(3)基于 LSH 的方法在数据的存储位置中保留了数据之间的内容相似性,便于进行相似性查询.

基于 LSH 面向混合类型数据进行相似性查询的关键点是如何有效地构建混合哈希值,从而可以有效地处理查询并且准确地返回结果.具体地说,需要解决的挑战有:(1)不同类型数据的相似度度量方法是截然不同的,所以第一个挑战是如何构建混合哈希值使得对应的哈希桶里的数据对象两种类型数据都是相似的;(2)用户进行相似性查询时查询范围是变化的,但 LSH 方法一般根据固定的敏感范围计算相似的数据对象.比如构建一个空间 LSH 的敏感距离是 300 m,在这个 LSH 索引上就不能进行 400 m 或 500 m 的查询.因此第二个挑战是如何在构建的 LSH 索引上有效地处理可变查询范围的适应性查询;(3)如何有效地将混合索引部署在分布式环境中.

为了解决这些挑战,本文提出了二元混合 LSH 查询方法,该方法根据数据对象中两种数据类型的相似性构建混合哈希值,通过混合哈希值将较相似的数据对象以较大的概率划分到同一个哈希桶中,以便于处理相似性查询.通过分析证明从理论上保证了二元混合 LSH 查询方法的准确性和有效性.之后我们基于混合 LSH 提出了可改变查询范围的适应性查询算法和 kNN 算法,最后我们在分布式环境下实现了算法.本文的主要贡献可总结如下:

(1)综合考虑两种数据类型的相似性,巧妙构建二元混合 LSH 索引和查询方法,通过理论分析证明了该方法的准确性和有效性.

(2)基于二元混合 LSH 查询方法,提出了具有准确性保证的可变查询范围的 NN 查询算法(见定义 2)以及约近的 kNN 查询算法(见定义 3),并给出了算法的性能分析.

(3)基于分布式平台实现二元混合 LSH 查询方法及相关算法,通过大量实验验证方法的可伸缩性和有效性.

本文第 2 节给出本文使用的数据模型和查询定义等基本知识;第 3 节提出二元混合 LSH 索引;第 4 节给出可变查询范围的 NN 查询算法;第 5 节提出约近 kNN 查询算法;第 6 节讨论分布式算法实现的相关问题;第 7 节给出实验结果和实验分析;在第 8 节给出相关工作;在第 9 节对本文进行总结.

2　预备知识

下面给出本文使用的数据模型.

本文的数据模型是面向含有二元混合数据类型的数据对象.注意本文主要专注于基于欧氏距离的数值型数据和基于 Jaccard 距离的字符型数据的二元混合数据类型.为了便于理论分析,本文假设两种数据类型是相互独立的.

设 O 是全体数据集,O 中的数据对象 o 含有两种数据类型,表示为 $o=\{f\text{-}type, s\text{-}type\}$,其中 $o.f\text{-}type$ 是指对象 o 的数值型数据,$o.s\text{-}type$ 是代表对象 o 的字符型数据.以线性加权和的方式计算不同类型相似性被广泛采用[13-15],对于 $\forall o_1, o_2 \in O$,o_1 和 o_2 的相似性计算函数为 DistEJ,如式(1)所示.其中 DistE 是针对 $f\text{-}type$ 的归一化欧氏距离,见式(2).DistJ 是针对 $s\text{-}type$ 的 Jaccard 距离,见式(3),两种数据类型比重的参数为 $\alpha \in (0,1)$.

$$\begin{aligned}\mathrm{DistEJ}(o_1,o_2)=&\\ &\alpha\times\mathrm{DistE}(o_1.f\text{-}type,o_2.f\text{-}type)+\\ &(1-\alpha)\times\mathrm{DistJ}(o_1.s\text{-}type,o_2.s\text{-}type)\end{aligned}\quad(1)$$

$$\mathrm{DistE}(o_1.f\text{-}type,o_2.f\text{-}type)=\frac{\mathrm{EuclideanDist}(o_1.f\text{-}type,o_2.f\text{-}type)}{dmax} \quad (2)$$

$$\mathrm{DistJ}(o_1.s\text{-}type,o_2.s\text{-}type)=1-\frac{(o_1.s\text{-}type\bigcap o_2.s\text{-}type)}{(o_1.s\text{-}type\bigcup o_2.s\text{-}type)} \quad (3)$$

因为 LSH 方法具有对高维数据不敏感、易于维护、对查询结果的准确性有理论保证等优点,已有一些研究工作将 LSH 用于单一数据类型的相似性查询[16-17],本文首次尝试将其扩展到混合数据类型. 下面给出要用到的 LSH 相关的定义.

LSH 是一种概率划分方法,通过哈希函数族将相似的数据对象以较高的概率存放在同一哈希桶中. 形式化描述为:设 O 是数据对象集,$o\in O$,$\mathcal{D}(o_1,o_2)$ 是数据对象 o_1 和 o_2 之间的距离,对于查询数据对象 q 和查询半径 r,$B(r,q)=\{o\in O|\mathcal{D}(q,o)\leq r\}$. 一个 LSH 函数族定义如下[18].

定义 1. LSH. 一个哈希函数族 $\mathcal{H}=\{h:R^d\rightarrow U\}$ 当满足下列条件时,则被称为 (r_1,r_2,p_1,p_2)-sensitive LSH,对于 $\forall o_1,o_2\in O$,

(1) 如果 $\mathcal{D}(o_1,o_2)\leq r_1$,则 $h(o_1)=h(o_2)$ 的概率大于或等于 p_1;

(2) 如果 $\mathcal{D}(o_1,o_2)\geq r_2$,则 $h(o_1)=h(o_2)$ 的概率小于或等于 p_2. 其中 $r_1<r_2$,$p_1>p_2$.

因为计算数据对象的相似度使用了欧氏距离和 Jaccard 距离,下面分别介绍针对这两种距离的 LSH 方法.

适用于欧氏距离的 LSH 函数族是以如下的方式构建的[17],首先将数据对象 $o\in O$ 投影到一条直线上,然后对这个投影进行常量 b 的位移. 然后以宽度为 W 对这条线分割成多个段,最后哈希函数返回位移后 o 的投影所在的段. 正式的定义为 $h_{a,b}(\boldsymbol{o})=\left\lfloor\frac{\boldsymbol{a}\cdot\boldsymbol{o}+b}{W}\right\rfloor$,其中,$\boldsymbol{o}$ 是数据对象 o 的向量形式,$\boldsymbol{a}$ 是随机生成的 d 维向量,每一维根据正则分布 $\mathcal{N}(0,1)$ 相互独立生成的,W 是由用户指定的常数,b 是在 $(0,W)$ 之间随机生成的实数. 对于两个数据对象 o_1 和 o_2,根据文献[18],按照 $h_{a,b}$ 计算哈希值,o_1 和 o_2 具有相同哈希值的概率是 $ps(x)=Pr_{a,b}[h_{a,b}(o_1)=h_{a,b}(o_2)]=\int_0^W\frac{1}{x}f_2\left(\frac{t}{x}\right)\left(1-\frac{t}{W}\right)\mathrm{d}t$,其中 f_2 是标准正则概率密度函数. 因此上述定义的哈希函数族是 (d_1,d_2,p_1,p_2)-sensitive LSH,其中 d_1 和 d_2 是欧氏距离常量,$p_1=ps(r_1)$,$p_2=ps(r_2)$.

适用于 Jaccard 距离的 LSH 是 min-hash[19]. 设 T 是一个词组集 $h(T)=\arg\min_{t_i\in T}g(t_i)$,其中 $t_i\in T$,$g(t_i)$ 是随机数生成函数. H 为 $g(t_i)$ 值域的基. 对于两个词组集 T_1 和 T_2,容易得出 $Pr[h(T_1)=h(T_2)]=\frac{|T_1\bigcap T_2|}{|T_1\bigcup T_2|}$,因此该哈希函数族是 $(\mathrm{r}_1,\mathrm{r}_2,p_1,p_2)$-sensitive LSH,其中 $\mathrm{r}_1,\mathrm{r}_2$ 是 Jaccard 距离常量,$p_1=1-\mathrm{r}_1$,$p_2=1-\mathrm{r}_2$.

给定一个查询对象 q,距离 R 和一个约近系数 c,传统 (R,c)-NN 查询是指[17]:如果存在和查询 q 的距离小于 R 的数据对象,则返回一个距离 q 小于 cR 的数据对象. 但是这是针对单一数据类型,我们将其扩展到二元混合数据类型上.

定义 2. (D,R,c)-NN 查询. 设 O 是二元混合数据对象集,q 为查询数据对象,D 为数值类型的查询距离,R 为字符类型的查询距离,c 为约近因子,令 $B(D,R,q)=\{o\in O|\mathrm{DistE}(o,q)\leq D\&\mathrm{DistJ}(o,q)\leq R\}$,$(D,R,c)$-NN 查询是指

(1) 如果 $B(D,R,q)$ 非空,则返回一个属于 $B(cD,cR,q)$ 的数据对象;

(2) 如果 $B(cD,cR,q)$ 范围内无数据对象,则返回空.

下面进一步给出面向二元混合数据的 kNN 查询的定义.

定义 3. 约近 kNN 查询. 设 O 是二元混合数据对象集,DistEJ 为数据对象之间的距离函数,q 为查询数据对象,c 为约近因子,O 集合中距离 q 最小的 k 个数据对象即为查询 q 的 kNN,定义为 $o_1^*,\cdots,o_k^*$. 查询 q 的 c 倍约近 kNN 是指同查询 q 的距离是真实 kNN 距离 c 倍之内的数据对象,即如果 $\mathrm{DistEJ}(o_i,q)\leq c\times\mathrm{DistEJ}(o_i^*,q)$,其中 $o_i\in O$,$1\leq i\leq k$,则 $o_1,\cdots,o_k$ 是查询 q 的 c 倍约近 kNN.

表 1 汇总了本文常用的变量和函数名称.

表 1 本文中常用的变量和函数

符号	含义
P	全体数据集
n	数据集数据量
q	查询数据对象
k	最近邻查询返回的数据对象数量
D	数值型数据的查询范围
R	字符型数据的查询范围
d	数值型 LSH 的敏感半径
r	字符型 LSH 的敏感半径
C	LSH 查询时检查数据对象数量的系数
c	约近查询的近似系数
$B(r,q)$	对于单类型数据,距离查询 q 为 r 以内的数据对象集
$B(d,r,q)$	对于二元混合数据,距离 q 数值型距离 d 以内且字符型距离 r 以内的数据对象集
DistEJ	数据对象之间的混合距离(见式(1))
DistE	数据对象之间的数值型的距离(见式(2))
DistJ	数据对象之间的字符型的距离(见式(3))

3　二元混合 LSH 索引

一种处理二元混合数据的简单方法是：先用针对一种数据类型的查询方法过滤出候选数据对象，然后用另一种数据类型的方法再进行过滤. 直观地说，单一类型依次过滤的性能依赖于数据类型的选择顺序，同时单一类型的过滤效率要小于多种类型的协同过滤性能. 本节首先介绍二元混合 LSH 索引，然后通过理论分析证明其准确性和有效性.

二元混合 LSH 索引的基本思想是将所含两种类型数据都相似的数据对象以较大的概率哈希映射到同一个桶里，其构建过程主要由三步组成：如图 1 所示，首先，通过一个敏感范围为 D 的面向欧氏距离的 LSH，将 $o.f\text{-}type$ 转化为 k_1 个哈希值. 然后，通过一个敏感范围为 R 的面向 Jaccard 距离的 LSH，将 $o.s\text{-}type$ 转化为 k_2 个哈希值. 最后，将生成的 k_1 个哈希值和 k_2 个哈希值连接起来，就是 o 的混合哈希值.

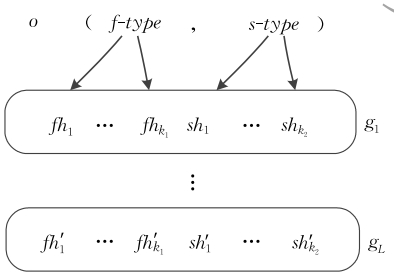


图 1　二元混合 LSH 索引示意图

具体地说，对于 $\forall o\in O, o=\{f\text{-}type, s\text{-}type\}$，假设已有如下 3 个参数：欧氏距离的敏感范围 d，Jaccard 距离的敏感范围 r，以及约近因子 $c>1$，二元混合 LSH 的构建过程如下：首先对于 $f\text{-}type$ 部分，基于欧氏距离构建一个 (d, cd, fp_1, fp_2)-sensitive LSH 族(参考第 2 节)，记为 fH. 对 $s\text{-}type$ 部分，基于 Jaccard 距离构建一个 (r, cr, sp_1, sp_2)-sensitive LSH 族，记为 sH. 然后合并两个 LSH 族，分别从 fH 和 sH 中随机选取 k_1 和 k_2 个哈希函数，定义一个新的 LSH 族：$G=g: S\rightarrow U^{(k_1+k_2)}$，即 $g(o)=\{fh_1(o),\cdots, fh_{k_1}(o), sh_1(o),\cdots, sh_{k_2}(o)\}$，其中 $fh_i\in fH, sh_i\in sH$. 定义一个整数 L，独立且随机地从 G 中选取 L 个函数，即 $g_1,\cdots, g_L$. 这样每个对象就会有 L 个哈希值，对应到 L 个哈希桶中. 为节省存储空间，哈希桶中只存储对象的指针.

显然，如图 1 所示，二元混合 LSH 哈希值是由 k_1 个来自 (d, cd, fp_1, fp_2)-sensitive LSH 哈希值和 k_2 个来自 (r, cr, sp_1, sp_2)-sensitive LSH 哈希值组合而成，因此我们称这样的混合哈希值为 $(d, r, c, fp_1, sp_1, fp_2, sp_2)$-sensitive 混合 LSH. 又因为 fp_1, sp_1, fp_2, sp_2 这些参数是由 d, r, c 决定的，因此称为 $\{d, r, c\}$-sensitive 混合 LSH，当上下文没有歧义时可以进一步简称为 (d, r)-sensitive 混合 LSH.

基于 (d, r, c)-sensitive 混合 LSH，我们可以进行 (d, r, c)-NN 查询，即对于一个查询对象 q，我们需要找到数据集中 $f\text{-}type$ 距离小于 d 且 $s\text{-}type$ 距离小于 r 的 c 倍约近的最近邻. 对于这样的查询我们首先生成 L 个哈希值 $g_1(q),\cdots, g_L(q)$，将查询发送给这 L 个哈希值对应的 L 个哈桶中，在桶内最多检查 $C\times L$ 个数据对象，其中 C 是一个常量. 设 $o_1,\cdots, o_{C\times L}$ 是被检查的数据对象. 对于任意的 o_i，如果 $\mathrm{DistE}(o_i, q)<cD$ 且 $\mathrm{DistJ}(o_i, q)<cR$，那么返回 o_i 作为最后的查询结果；否则就返回空(NULL).

为了保证算法的正确性，k_1，k_2 和 L 的选取必须以常数概率满足如下两个性质.

性质 1.　如果存在一个数据对象 o 使得 $\mathrm{DistE}(o.ftype, q.ftype)<D$ 且 $\mathrm{DistJ}(o.stype, q.stype)<R$，那么存在 $i\in[1, L]$，使得 $g_i(o)=g_i(q)$.

性质 2.　和查询对象 q 具有同样的哈希值，分在同一个哈希桶里但 $\mathrm{DistE}(o, q)>cD$ 或者 $\mathrm{DistJ}(o, q)>cR$ 的数据对象的数量小于 $C\times L$，C 是一个常数.

性质 1 保证了满足查询条件的数据对象和查询对象至少同在一个哈希桶中. 如果有查询结果性质 2 保证在检查 $C\times L$ 个数据对象后可以找到查询结果.

定理 1.　当 $k_1=\log_{fp_2}\left(1-\sqrt{1-\frac{1}{n}}\right)$，$k_2=\log_{sp_2}\left(1-\sqrt{1-\frac{1}{n}}\right)$，$L=\left(1-\sqrt{1-\frac{1}{n}}\right)^{-(\log_{fp_2}fp_1+\log_{sp_2}sp_1)}$ 时，性质 1 和性质 2 以常数概率成立.

证明.　已知 $k_1=\log_{fp_2}\left(1-\sqrt{1-\frac{1}{n}}\right)$，$k_2=\log_{sp_2}\left(1-\sqrt{1-\frac{1}{n}}\right)$. 不失一般性，假设存在某个数据对象 o^* 满足 $\mathrm{DistS}(o^*.f\text{-}type, q.f\text{-}type)<D$ 且 $\mathrm{DistT}(o^*.s\text{-}type, q.s\text{-}type)<R$.

用 $P(A)$ 表示事件 A 发生的概率. 对于 $\forall o'\in O$，令 $Pa=P(g(o')=g(q)\,\&\,o'\in(O-B(q, cD))\cap B(q, cR)))$，且令 $Pb=P(g(o')=g(q)\,\&\,o'\in(O-B(q, cD))\,|\,g(o')=g(q)\,\&\,o'\in(O-B(q, cD)))$，可

知 $Pa \leq Pb$. 又因为 $Pb=1-(1-fp_2^{k_1})(1-sp_2^{k_2})=\frac{1}{n}$，所以 $Pa \leq \frac{1}{n}$. 所以和 q 具有同一个哈希值 g_i 但又不符合查询条件的数据对象个数的期望值 $\left(\frac{1}{n}\times n=1\right)$ 小于 1.5. 对于 L 次 g_i 的哈希值则期望值小于 1.5L. 那么根据马尔科夫不等式，所有和 q 具有相同哈希值但不符合查询条件的数据对象个数大于 3L 的概率小于 $\frac{1}{2}$. 因此性质 2 以常数概率 $\left(>\frac{1}{2}\right)$ 成立.

因为

$$P(g(o^*)=g(p))=fp_1^{k_1}\times sp_1^{k_2}=\left(1-\sqrt{1-\frac{1}{n}}\right)^{(\log_{fp_2} fp_1+\log_{sp_2} sp_1)},$$

令 $L=\left(1-\sqrt{1-\frac{1}{n}}\right)^{-(\log_{fp_2} fp_1+\log_{sp_2} sp_1)}$，则 $P(g(o^*)\neq g(p))<\frac{1}{e}$. 所以性质 1 以 $1-\frac{1}{e}>\frac{1}{2}$ 的概率成立. 证毕.

根据定理 1 证明过程，本文可设 $C=3$. 从定理 1 的证明可知性质 1 成立的概率大于 $\frac{1}{2}$，性质 2 成立的概率大于 $\frac{1}{2}$，则性质 1 和性质 2 同时成立的概率大于 $\frac{1}{2}\times\frac{1}{2}=\frac{1}{4}$. 因此定理 1 中的常数概率是指大于 $\frac{1}{4}$ 小于 1 的常数. 常数概率的意义在于通过多次执行可以使概率逐渐趋近于 1.

4 适应性(D,R,c)-NN 查询处理

本节提出了如何将大范围的查询分解为小范围的查询，从而可以在具有固定敏感距离的二元混合 LSH 上进行变化查询范围的 NN 查询处理. 回顾上一节，比如存在一个(d,r,c)-sensitive 的混合 LSH，可以很容易地基于该混合 LSH 进行(d,r,c)-NN 查询. 但当有(D,R,c)-NN 查询时，其中 $D>d$，$R>r$，需要将该查询转化为多个为(d,r,c)-NN 的子查询之后，才能基于该混合 LSH 进行查询处理.

4.1 适应性(D,R,c)-NN 查询

基于已构建的(d,r,c)-sensitive 混合 LSH 处理任意的(D,R,c)-NN 查询，注意，d 和 r 足够小，使得 $d \leq D$ 且 $r \leq R$. 适应性查询的基本思想是，将原始查询转化为可以在已构建的混合 LSH 上直接处理的多个子查询. 下面我们给出正式的定义.

定义 4. LSH 查询转换. LSH 查询转换的概念是基于单数据类型的(r_1,r_2,p_1,p_2)-sensitive LSH. 已有(r_1,r_2,p_1,p_2)-sensitive LSH，对于任意(R,c)-NN 查询 q，其中 $R>r_1$，$cR>r_2$，如果该查询可以转化为一个基于该 LSH 的含有多个子查询的查询集 S，同时 S 满足下面所列的两个条件，则 S 是查询 q 基于(r_1,r_2,p_1,p_2)-sensitive LSH 的一个查询转换：

T_1：查询 q 覆盖的空间或者内容要被 S 集中 r_1 覆盖的空间或内容包含；

T_2：S 集中 r_2 覆盖的空间或者内容要被查询 q 中 cR 的面积或内容包含.

下面通过引理 1 给出 LSH 查询转换的功能特性.

引理 1. 如果(R,c)-NN 查询 q 有查询结果，那么 q 的一个基于(r_1,r_2,p_1,p_2)-sensitive LSH 的查询转换可以以常数概率(大于 $\frac{1}{2}$ 且小于 1)返回 c 倍约近的查询结果.

证明. 设 QT 为(R,c)-NN 查询 q 的一个查询转换，$QT=\{qt_i \mid 0<i<\|QT\|\}$. 如果存在一个对象 $o^* \in B(q,R)$，那么根据定义 4 中的条件 T_1，则 o^* 至少在 QT 集中一个子查询的查询范围中，假设为 qt_i. 又因为在 $B(q,cR)$ 之外的对象也都在 $B(qt_i,r_2)$ 之外，根据 LSH 查询[18]的性质 2，和 qt_i 具有相同哈希值且在 $B(qt_i,r_2)$ 之外的数据对象个数小于 C×L，因此在同一哈希桶里当检验 C×L 个对象之后可以以常数概率(大于 $\frac{1}{2}$ 且小于 1)返回一个 c 倍约近的查询结果. 所以引理 1 得证. 证毕.

结合定理 1 和引理 1，使用 LSH 查询转换可以处理具有可变查询范围的(R,c)-NN 查询，但是对基于欧氏距离的数值型数据和基于 Jaccard 距离的字符型数据构建 LSH 查询转换并不容易. 下面我们就分别介绍欧氏距离和 Jaccard 距离构建 LSH 查询转换的方法.

(1) 基于欧氏距离数值型数据的 LSH 查询转换

不失一般性，以二维的欧式空间为例. 对于欧式空间数值型数据，构建一个 LSH 查询转换类似于圆覆盖问题. 目前圆覆盖问题的最优解只在若干特定情况下才有. 我们给出了一种通用方法并证明该

方法是 2 倍约近最优方法.

假设已有一个(d,cd,p_1,p_2)-sensitive LSH 和一个(D,c)-NN 查询 q,其中 $d<D$,查询 q 的中心点是 $p=(p_x,p_y)$. 如图 2 所示,粗体的实线圆为查询 q 的查询范围,即(p_x,p_y)为圆心以 D 为半径的圆,用边长为$\sqrt{2}d$ 的正方形来划分圆,则需要的正方形数量是$\left\lceil\frac{2D^2}{d^2}\right\rceil$. 然后以每一个正方形的中心点为查询中心进行$(d,c)$-NN 查询,这些查询组成一个集合 $ST=\{st_i\}$,则 ST 即为 q 的一个 LSH 查询转换. 因为 $d<D$,所以 ST 很容易满足定义 4 中的 T_2,即 ST 中以 cd 为半径的圆所覆盖的面积被 q 中以 cD 为半径的圆所包含. 根据面积公式,用为半径为 d 的圆覆盖查询 q 半径为 D 的圆,使用圆的数量最少是$\left\lceil\frac{D^2}{d^2}\right\rceil$,因为$\left\lceil\frac{2D^2}{d^2}\right\rceil\leq 2\times\left\lceil\frac{D^2}{d^2}\right\rceil$,所以 ST 是 2 倍约近最优的 LSH 查询转换. 因此很容易得出以下推论.

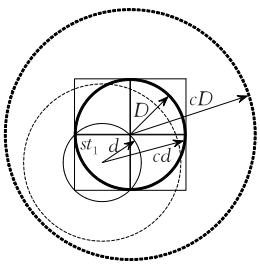


图 2　基于欧氏距离的查询转换

推论 1.　存在常数 c 使得 ST 是查询 q 的 LSH 查询转换,并且是 2 倍近似最优的.

(2)基于 Jaccard 距离字符型数据的 LSH 查询转换

对于基于 Jaccard 距离的字符型数据,假设已有(r,cr,p_1,p_2)-sensitive LSH 和一个(R,c)-NN 查询 q,其中 $r<R$,且 q 中有 TL 个词组. 为了通过该 LSH 找到与查询 q 相似的数据对象,有两种情况需要考虑,一种是含的词组数量比 q 少的数据对象,另一种是含的词组数量比 q 多的数据对象. 对于比 q 含的词组数量少的数据对象,我们需要去除掉 q 中的一些词组使得 q 被哈希到与这些数据对象一样的哈希桶里,从而可以得到查询结果. 指定一个整数 m,生成 q 中 TL-m 个词组的所有组合,这样就生成一个查询集 DS. 对于比 q 含的词组数量多的数据对象,我们需要在 q 中增加一些通配符使得 q 和这些数据对象匹配. 同样根据指定的整数 w,在 q 中增加 w 个通配词组,当随机抽取到 q 中的通配词组来生成哈希值时,通配词组会被哈希成所有可能的哈希值. 例如其中的一个词组会被哈希成 0 或者 1 时,而通配词组会被哈希成 0 和 1 两种情况. 用这种方法,有生成了一个查询集 AS. 通过合并 DS 和 AS 就得到了查询集 TT.

设 IN 和 UN 分别为两个词组集的交集的元素数和并集的元素数. 当参数 m 和 w 满足式(4)、(5)和式(6),可以保证和查询 q 相似的数据对象都被 TT 查询集中相似的数据对象包含,即满足定义 4 中的条件 T_1. 当参数 m 和 w 满足式(7)、(8)和式(9)时,可以保证 TT 查询集中 c 倍约近相似的数据对象被查询 q 中的 c 倍约近相似的对象包含,即满足定义 4 中的条件 T_2. 当这些公式都被满足时,则定义 4 中的条件 T_1 和条件 T_2 都满足,则 TT 为查询 q 的一个 LSH 查询转换. 通过计算最小满足条件的参数 m 和 w,可以得出推论 2.

$$\begin{cases}\frac{IN}{UN}\geq 1-R & (4)\\ \frac{IN}{UN-m}\geq 1-r & (5)\\ \frac{IN+w}{UN+w}\geq 1-r & (6)\end{cases}$$

$$\begin{cases}\frac{IN}{UN}\leq 1-cR & (7)\\ \frac{IN}{UN-m}\leq 1-cr & (8)\\ \frac{IN+w}{UN+w}\leq 1-cr & (9)\end{cases}$$

推论 2.　当 $m=\left\lceil\frac{R-r}{(1-R)(1-r)}TL\right\rceil$,$w=\left\lceil\frac{R-r}{(1-R)(1-r)}TL\right\rceil$时,其中 TL 为查询 q 中的词组数,查询集 TT 是查询 q 的一个 LSH 查询转换.

证明.　首先考虑式(4)、(5)和式(6),根据式(4)可以得出

$$UN\leq\frac{IN}{1-R} \tag{10}$$

根据式(5),m 需要满足

$$m\geq\frac{UN\times(1-r)-IN}{1-r} \tag{11}$$

因为式(10),当 $m\geq\frac{\frac{IN}{1-R}(1-r)-IN}{1-r}=\frac{(R-r)IN}{(1-r)(1-R)}$,进一步当 $m\geq\frac{(R-r)TL}{(1-R)(1-r)}$时,

m 可以满足式(11). 根据式(6),w 应当满足

$$w \geq \frac{(1-r)UN-IN}{r} \tag{12}$$

因为式(10),当 $w \geq \dfrac{(1-r)\dfrac{IN}{1-R}-IN}{r} = \dfrac{(R-r)IN}{r(1-R)}$,进一步当 $w \geq \dfrac{(R-r)TL}{r(1-R)}$ 时,w 可以满足式(12). 所以当 $m \geq \dfrac{(R-r)TL}{(1-R)(1-r)}$ 且 $w \geq \dfrac{(R-r)TL}{r(1-R)}$ 时,式(4)、(5)和式(6)成立.

类似地,对于式(7)、(8)和式(9),当 $m \leq \dfrac{(cR-cr)TL}{(1-cR)(1-cr)}$ 且 $w \leq \dfrac{(R-r)TL}{r(1-cR)}$ 时,式(7)、(8)和式(9)成立.

因为当 $c \geq 1$ 且 $1-cR>0$ 时,$\dfrac{(cR-cr)TL}{(1-cR)(1-cr)} > \dfrac{(R-r)TL}{(1-R)(1-r)}$,$\dfrac{(R-r)TL}{r(1-R)} > \dfrac{(R-r)TL}{r(1-cR)}$,所以 m 和 w 的取值存在,且当 $m = \left\lceil \dfrac{R-r}{(1-R)(1-r)}TL \right\rceil$,$w = \left\lceil \dfrac{R-r}{(1-R)(1-r)}TL \right\rceil$ 时,式(4)~(8)和式(9)成立. 即推论 2 成立. 证毕.

基于 LSH 查询转换,可以构建适应性(D,R,c)-NN 查询算法. 适应性(D,R,c)-NN 查询算法的基本思想是分解混合类型的查询成单类型的查询,之后对单类型的查询构建 LSH 查询转换,然后连接两种类型的查询,最后基于已有的混合 LSH 处理这些连接过的查询.

具体地,如图 3 所示,假设已有(d,r,c)-sensitive 混合 LSH,如第 3 节所述,基于此可以很容易处理(d,r,c)-NN 查询,当有(D,R,c)-NN 查询时,首先将(D,R,c)-NN 的查询结果等价于(D,c)-NN 和(R,c)-NN 的查询结果的连接,构建(D,c)-NN 查询和(R,c)-NN 分别基于(d,cd,fp_1,fp_2)-sensitive LSH 和(r,cr,sp_1,sp_2)-sensitive LSH 的 LSH 查询转换 DT 和 RT,连接 DT 和 RT 生成 DRT. 最后在(d,r,c)-sensitive 混合 LSH 上处理 DRT 查询集并返回结果. 具体过程见算法 1.

算法 1. 适应性(D,R,c)-NN 查询算法.

输入:(d,r,c)-sensitive 混合 LSH,(D,R,c)-NN 查询 q

输出:q 的 c 倍约近 NN

1. 计算出(D,c)-NN 基于(d,cd,fp_1,fp_2)-sensitive LSH 的 LSH 查询转换 DT.
2. 计算出(R,c)-NN 基于(r,cr,sp_1,sp_2)-sensitive LSH 的 LSH 查询转换 RT.
3. $DRT \leftarrow DT \times RT$.
4. 基于(d,r,c)-sensitive 混合 LSH 处理查询 DRT.

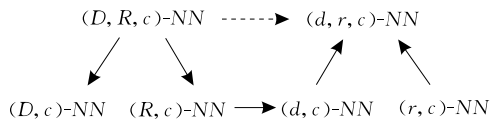


图 3 适应性查询示意图

定理 2. 如果(D,R,c)-NN 查询存在查询结果,那么通过适应性查询算法可以以常数概率(大于 $\frac{1}{2}$ 且小于 1)返回该查询的 c 倍约近结果.

证明. 已有(D,R,c)-NN 查询 q,设 DRT 是连接后的(d,r,c)-NN 查询集,$DRT=\{drt_i \mid 0<i<|DRT|\}$. 假设有一个数据对象 $o^* \in B(q,D) \cap B(q,R)$,则存在某一整数 i,使得 $o^* \in B(drt_i,d) \cap B(drt_i,r)$. 令 $O'=O-B(q,D) \cap B(q,R)$,则 $O' \subseteq O-B(drt_i,d) \cap B(drt_i,r)$. 根据定理 1 中的性质 2,与 drt_i 在同一个哈希桶里的 O' 集中数据对象的个数小于 $3L$. 因此检验过 $3L$ 个的数据对象后,drt_i 查询可以以常数概率(大于 $\frac{1}{2}$ 且小于 1)返回 c 倍约近的 q 的查询结果,从而定理 2 得证. 证毕.

4.2 多混合 LSH 适应性查询方法

根据定理 1,每个 LSH 查询转换集合中的每个查询最多需要检验 $3L$ 个数据对象,所以哈希桶的 I/O 次数和子查询集中的查询个数成正比,即 LSH 查询集中子查询的个数是查询代价的重要指标. 结合推论 1 和推论 2,适应性查询中被连接后的 LSH 查询转换集中的子查询个数(QN)是:

$$QN \leq \left(\frac{2D^2}{d^2}+1\right)(C_{TL}^{m}+H^{w}) \tag{13}$$

其中,C_{TL}^{m} 从 TL 个不同词组中取出 m 个词组的组合数. 如式(13)所示,当查询范围比较大时,查询代价非常高,所以我们提出了多混合 LSH 适应性查询方法.

多混合 LSH 适应性查询方法的意图是构建多个混合 LSH,使得查询与其最近的混合 LSH 的距离尽可能的小,从而使子查询的个数减少. 下面给出两种多混合 LSH 适应性查询的方法.

(1)等比等差法

如式(13)所示,QN 与 $\dfrac{D}{d}$(数值信息部分)和

$(R-r)$(字符信息部分)成正比. 对于(D,R,c)-NN 查询,其中 $D\in[MinD,MaxD]$,$R\in[MinR,MaxR]$. 我们构建多个混合 LSH 分别为(d_i,r_j,c)-sensitive 混合 LSH,其中 $i\in\left(0,\log_b\left(\frac{MaxD}{MinD}\right)\right)$,$j\in\left(0,\frac{MaxR-MinR}{t}\right)$,$d_1=MinD$,$\frac{d_{i+1}}{d_i}=b$,$r_1=MinR$,$r_{j+1}-r_j=t$,$b$ 和 t 分别是等比数列的等比和等差数列的公差.

用等比等差法构造出来的多混合 LSH 适应性查询具有如下性质.

推论 3.　等比等差法用

$$\log_b\left(\frac{MaxD}{MinD}\right)\times\frac{MaxR-MinR}{t}$$

个混合 LSH 可以处理任意的(D,R,c)-NN 查询,其中 $D\in[MinD,MaxD]$,$R\in[MinR,MaxR]$,只需要最多$(2b^2+1)(C_{TL}^m+H^w)$个子查询的 LSH 查询转换,其中 $m=\left\lceil\frac{t}{(1-MaxR)^2}TL\right\rceil$,$w=\left\lceil\frac{t}{(1-MinR)(MinR-t)}TL\right\rceil$.

证明.　设多混合 LSH $MAH=\left\{mah_{ij}\mid mah_{ij}\right.$是$(d_i,r_j,c)$-sensitive 混合 LSH,$i\in\left(0,\log_b\frac{MaxD}{MinD}\right)$,$\left.j\in\left(0,\frac{MaxR-MinR}{t}\right)\right\}$,已知一个$(D,R,c)$-NN 查询 q,距离查询 q 最近的混合 LSH 是 mah_{ks},其中 $k=\underset{(0<i<\log_b\frac{MaxD}{MinD},D>d_i)}{\arg\min}\frac{D}{d_i}$,$s=\underset{0<j<\frac{MaxR-MinR}{t},R>r_j}{\arg\min}(R-r_j)$. 对于数值型部分,$\frac{D}{d_k}\le b$,则 $2\frac{D^2}{d^2}+1\le 2b^2+1$. 对于字符型部分,因为 $R-r_s\le t$,$R>r_s$,$\frac{R-r_s}{(1-R)(1-r_s)}$是 r_s 的单调增函数,所以$\frac{R-r_s}{(1-R)(1-r_s)}<\frac{t}{(1-MaxR)^2}$,因为$\frac{R-r_s}{(1-R)r_s}$是 r_s 的单调减函数,所以$\frac{R-r_s}{(1-R)r_s}<\frac{t}{(1-MinR)(MinR-t)}$. 所以 $QN\le(2b^2+1)(C_{TL}^m+H^w)$,其中 $m=\left\lceil\frac{t}{(1-MaxR)^2}TL\right\rceil$,$w=\left\lceil\frac{t}{(1-MinR)(MinR-t)}TL\right\rceil$.　证毕.

用等比等差法构造出来的多混合 LSH 适应性查询保证了最差情况下计算复杂度的下限. 例如,对于任意(D,R,c)-NN 查询,$1<D<27$,$0.1<R<0.14$,设置 $b=3$,$t=0.02$,等比等差法使用 $\log_3 27\times\frac{0.14-0.1}{0.02}=6$ 个混合 LSH 使得当 $H=3$,$TL=6$ 时 LSH 查询转换的子查询个数小于 285. 注意这是最差情况下的子查询数.

(2)贪心方法

虽然等比等差法可以保证性能下界,但该方法试图平均地分配多个混合 LSH 的范围,而实际查询的范围一般情况下并不是均匀分布的. 基于这样的事实我们提出了贪心方法. 贪心方法的思想是混合 LSH 尽可能地与最频繁查询的查询范围接近从而减少总体的查询转换代价.

假设由于存储空间的限制,最多可以构建 S^2 个混合 LSH. 首先,根据查询样本集计算出样本数据空间部分查询范围的概率分布函数 F. 然后,将 F 函数的 y 轴分解为 S 份,则每份为$\frac{1}{S}$. 再计算每份的查询范围值 $d_i=F^{-1}\left(i\times\frac{1}{S}\right)$,$i\in[1,S]$. 用同样的方式计算出样本数据文本部分的查询范围值 $r_j=F^{-1}\left(j\times\frac{1}{S}\right)$,$j\in[1,S]$. 最后构建$(d_i,r_j,c)$-sensitive 混合 LSH.

等比等差法和贪心方法可以结合使用,算法开始运行时可以用等比等差法,当运行过一段时间,查询的样本被抽取出来,则可根据查询的频度使用贪心方法. 当查询的分布是均匀分布时,贪心方法和等比等差法本质上是一样的.

5　约近 kNN 查询方法

本节提出了基于混合 LSH 的约近 kNN 查询算法.

二元混合相似度计算公式如式(1)所示. 设 us 和 ut 分别是数值类型部分和字符类型部分的最小查询范围,根据式(1),令 $udist=\alpha\times us+(1-\alpha)\times ut$,$KL$ 为 k 个候选最近邻的列表. 已知一个 kNN 查询 q,首先使用$\left(\frac{udist}{\alpha},c\frac{udist}{1-\alpha}\right)$-NN 查询检验 $C\times L+k$ 个数据对象,然后将距离最近的 k 个数据对象放到 KL 中. 如果在对应的哈希桶中没有 k 个数据对象或者 KL 中有数据对象不属于

$B\left(q,c\frac{udist}{\alpha},c\frac{udist}{1-\alpha}\right)$，则扩大查询范围 c 倍，即进行 $\left(c\frac{udist}{\alpha},c\frac{udist}{1-\alpha}\right)$-NN 查询. 这个过程循环执行直到足够数量满足条件的数据对象被找到. 最后返回 KL 中的 k 个数据对象. 算法详细过程见算法 2.

算法 2. 约近 kNN 查询算法.

输入：(d,r,c)-sensitive 混合 LSH，kNN 查询 q，权重 α

输出：kNN 查询结果

1. 初始化 $dist\leftarrow udist$
2. $KL\leftarrow\varnothing$
3. $o_k\leftarrow$NULL
4. For $|KL|<k$ or $o_k\notin B\left(q,c\frac{udist}{\alpha},c\frac{udist}{1-\alpha}\right)$ do
5. $TempleList\leftarrow$通过 $\left(\frac{udist}{\alpha},\frac{udist}{1-\alpha}\right)$-NN 查询获取 $3L+k$ 个数据对象
6. $KL\leftarrow$从 $TempleList$ 选择出候选 k 个最近邻
7. $o_k\leftarrow KL$ 中第 k 个数据对象
8. $dist\leftarrow c\times dist$
9. End For
10. Return KL

定理 3. 算法 2 返回的查询 q 的 kth-NN 是 q 的真实 kth-NN 的 $2c^2$ 倍的约近.

证明. 设 o^* 是查询 q 的真实的 kth-NN，o_k 是通过算法 2 返回的约近 kth-NN.

如果查询结果是在第一个 NN 查询后返回的，因为定理 1 的性质 2 使得检查到的数据对象不属于 $B\left(q,c\frac{udist}{\alpha},c\frac{udist}{1-\alpha}\right)$ 数据对象小于 $3L$，则返回的 k 个数据对象与查询 q 的距离都小于 $2c\times udist$，即 $\mathrm{DistEJ}(o_k,q)<2c\times udist$. 又因为 $\mathrm{DistEJ}(o^*,q)>udist$，所以 o_k 是 o^* 的 $2c^2$ 约近.

如果 o_k 是在第 i 个循环返回的，其中 $i>1$，那么根据定理 1 的性质 1，$o^*.f\text{-}type>c^{i-1}\frac{udist}{\alpha}$ 或者 $o^*.s\text{-}type>c^{i-1}\frac{udist}{1-\alpha}$. 根据式(1)则 $\mathrm{DistEJ}(q,o^*)>c^{i-1}udist$，即 $2c^2\,\mathrm{DistEJ}(q,o^*)>2c^{i+1}udist$. 因为 $o_k.f\text{-}type<c^{i+1}\frac{udist}{\alpha}$ 且 $o_k.s\text{-}type<c^{i+1}\frac{udist}{1-\alpha}$，所以 $\mathrm{DistEJ}(q,o_k)<2c^{i+1}udist$. 那么 $\mathrm{DistEJ}(q,o_k)<2c^2\mathrm{DistEJ}(q,o^*)$，即 o_k 是 o^* 的 $2c^2$ 约近. 证毕.

6 分布式处理算法和优化

为了验证算法在处理大规模数据时的易维护性和可伸缩性，我们将混合 LSH 整合到分布式环境中. 本文提出的算法不依赖于全局信息，所以相比其他混合索引它们更容易扩展到在各个计算节点相互独立的 P2P 的分布式环境中. 又因为相似的对象容易在同一个哈希桶中被查找到，因此查询通信代价也较低. 在本节中，我们将讨论算法在分布式环境中实现的问题.

6.1 查询过程

关于数据存储，数据对象的哈希值生成后，它会被发送到具有相同 ID 的哈希桶中. 哈希桶是在分布式环境下计算节点以一致性哈希的方式管理维护.

分布式 (D,R,c)-NN 查询算法见算法 4. 具体地说，对于查询，每个计算节点负责一定范围的键值，查询是以 DHT 的方式路由和转发. 具体地说，当一个计算节点接收一个 (D,R,c)-NN 查询 q，该计算节点会充当协调者并负责转发查询，收集中间结果，并返回最终的结果给用户. 首先，协调者计算 q 的 L 个哈希值，L 个哈希值对应的 L 个哈希桶，q 会被转发给维护这 L 个哈希桶的计算节点上，接收到转发查询的计算节点计算对应哈希桶中的 3 个数据对象，如果有满足查询 q 的查询条件的数据对象，则返回该数据对象给协调者，否则返回 FALSE. 各个计算节点在返回结果的同时会返回哈希桶中数据对象的个数，以便于协调者统计并进行下轮转发. 因为有的哈希桶中数据对象有可能小于 3，所以当协调者计算的数据对象总数小于 $3L$ 时，协调者进行第二轮转发，将查询转发给哈希桶中有足够数量数据对象的计算节点. 最终，协调者根据收集到的数据对象计算出 NN，并将其返回给查询请求者. 因此协调者最多需要二轮通信就可以得出结果. 注意每个哈希桶需要维护一个指针，记录上一次哈希桶计算的数据对象的位置，从而可以以轮询的方式(round-robin)计算哈希桶中的数据对象.

对于适应性混合 LSH 算法，接收到查询的计算节点首先将 (D,R,c)-NN 转化为多个 (d,r,c)-NN 子查询，接下来的计算过程和上述 (D,R,c)-NN 的过程类似. 但是为了更快的得到结果，距离中心点较近的点会优先处理. 对于 kNN 查询算法，如算法 2 所示，算法主要由多次的 (D,R,c)-NN 组成，因此其分布式实现过程也类似.

算法 3. 分布式 (D,R,c)-NN 查询算法.

输入：(D,R,c)-NN 查询 q

输出：NN 查询结果

1. 计算节点作为协调者接收 (D,R,c)-NN 查询 q.

2. 协调者计算 q 的 L 个哈希值.
3. 协调者将 q 分别转发 L 个哈希桶所在的计算节点.
4. 接收到转发查询的计算节点以轮询的方式计算对应哈希桶中的 3 个数据对象，并返回满足条件的数据对象给协调者.
5. 协调者根据返回数据对象数量进行第二轮转发.
6. 协调者根据收集到 $3L$ 个数据对象计算出 NN.
7. 协调者返回 NN 给查询请求者.

6.2 负载均衡

负载均衡是影响分布式系统性能的重要指标之一. 已有工作[20]通过优化数据存储的方式来维持负载平衡. 但查询的分布是动态变化的，因此本节提出查询敏感的负载平衡方法.

本文采用的具体负载均衡方法见算法 4. 起初各个数据节点负责相同大小的数据范围，比如有 5 个计算节点且总范围是[1,100]，则 5 个计算节点分别负责[1,20]，[21,40]，[41,60]，[61,80]和[81,100]. 当一个计算节点会周期地和其它计算节点比较负载，如果它的负载超过平均负载的一个阈值，比如 150%，则该计算节点保持具有平均负载的范围然后将剩下的范围的数据发给相邻的计算节点并更新 DHT. 之后如果其相邻的节点的负载超过平均负载的阈值则重复以上过程直到实现系统的负载平衡.

算法 4.　查询敏感负载均衡算法.

输入：数据范围$[dr_1, dr_2]$，负载阈值 θ，节点 i 动态查询负载

输出：周期调整完成信号 TRUE

1. 均匀划分各个计算节点负载$[dr_i, dr_{i+1}]$，计算节点 $i \in [0, cn]$
2. For 每一个时间周期 do
3. For 计算节点 i，$i \in [0, cn]$
4. If $ld_i > \theta$
5. 计算节点 i 保留平均负载，将剩余的数据范围发送给计算节点 $i+1$
6. 更新 DHT 路由表
7. i++
8. End If
9. 返回 TRUE
10. End For
11. End For
12. 超时返回 FALSE

7 实验测试与分析

7.1 实验设置

本文实验使用了真实数据集和合成数据集来评价算法的准确性和效率，实现了二元混合 LSH 查询方法(BHL)，基于等比等差法的多二元混合 LSH 适应性查询方法(MBHL-E)和基于贪心方法的多二元混合 LSH 适应性查询方法(MBHL-G).

本文的查询方法是面向含数值型和字符型的二元混合数据. 为了验证查询方法的效率和准确性，实验采用了两个数据库：MicroBlog 和 SynSet. MircoBlog 是抽取自新浪微博的包含有图片和文本的微博数据集，含有的记录数是 100 万. 对于图片部分，通过 SIFT 算法[12]将图片数据转化为特征向量，图片之间的相似性是通过特征向量之间的欧氏距离衡量的. 对于文本部分，微博有 140 个字符的限制，然后去掉停止词(stop words)后平均约 32 个字符. SynSet 是含有位置信息和文本信息的合成数据集，位置信息是随机生成于边长为 100 km 的正方形区域，文本信息随机抽取自词典集. 通过调整 SynSet 数据集的平均词组数来验证算法处理高维数据的性能，通过调整 SynSet 数据集的大小，来验证算法的可伸缩性. 作为比较，根据不同数据类型的特征，本文实现了相关的方法 LSHDSS[20]和 SEAL[9]. 对于 MicroBlog 数据集，本文方法和 LSHDSS 进行了比较，LSHDSS 是一种基于 LSH 针对高维数据的分布式相似性查询方法，通过二级映射数据划分方法兼顾查询效率和负载均衡. LSHDSS 是一种主要针对单一数据类型的分布式相似性查询方法，本文通过二次哈希过滤扩展方法以便处理二元混合数据. 对于 SynSet 数据集，本文和 SEAL 进行比较，SEAL 是一种针对空间-文本类型的一种相似性查询方法，通过前缀过滤策略提高查询效率. 为了准确地验证分布式处理的性能，算法实现中不使用缓存技术.

实验运行在 50 台商务 PC 上，这些 PC 运行的操作系统是 Red Hat Linux system 6.1 操作系统、IntelCore i3 2100 3.1 GHz CPU 和 8 GB 内存.

对于单数据类型的 LSH，实验使用已有的基于欧氏距离的 LSH[18]和基于文本的 LSH[19]，且当 $W=5$，$H=7$ 时，实验效果最佳，因此实验中使用这样的设定，具体的参数取值见表 2. 为了准确验证算法的性能，当检验一个参数对算法性能的影响时，只有该参数的参数值发生变化而别的参数值保持默认值. 实验中的查询对象是从数据集中随机抽取出的数据对象，实验结果是 20 次运行结果的平均值.

表 2 实验参数

参数	取值范围	默认值
C	2,3	3
数据集大小(记录数)	1M～5M	1M
D	0.02～0.1	0.04
R	0.02～0.1	0.04
k	10～90	30
词组数量(token)	20～100	32
计算节点(cn)	10～50	10

实验中使用了 4 个性能指标:消息数、查询时间、查询准确度和 Gini 系数.

(1) 消息数

消息数是指一个查询从发起到收到结果总共使用的消息数量.消息数主要分为两部分:一部分是使用一致性哈希过程中找到负责某个哈希桶的计算节点,计算复杂度是 $O(\log(cn))$,其中 cn 是计算节点的数量.另一部分是转发查询和收集结果的消息数.分布式环境下消息数决定着通信代价、直接影响查询的可伸缩性和处理时间,是算法性能的重要指标之一.

(2) 查询时间

查询时间是指一个查询从发起到返回查询结果所使用的时间.相较于消息数来说,查询时间在一定程度上依赖于实验条件,但在调整某一参数时查询时间的变化可以较直观地反映出查询性能的变化趋势和参数的影响.

(3) 查询准确度

查询准确度用来衡量查询结果的质量.具体地,对于 kNN 查询,设 $o_1^*,\cdots,o_k^*$ 是真实的结果,$o_1,\cdots,o_k$ 是实际返回的结果,查询准确度 $accuracy=\frac{1}{k}\sum_{i=1}^{k}\frac{\mathrm{DistST}(q,o_i)}{\mathrm{DistST}(q,o_i^*)}$.对于 (D,R,c)-NN 查询,设 $k=1$,即可用相同的公式计算查询准确度.根据准确度的计算公式,查询准确度是大于等于 1 的,且越接近 1 表明查询结果越好.

(4) Gini 系数

Gini 系数是检验数据分布是否均衡的指标,$G=1-2\int_0^1 L(x)\mathrm{d}x$,其中 $L(x)$ 是分布的洛伦茨曲线,本文中用于检验负载均衡.

7.2 参数 c 的影响

首先检验参数约近系数 c 对空间代价和查询准确度的影响.表 3 是当 $c=2$ 或 $c=3$ 时二元混合 LSH 的性能.

表 3 参数 c 的影响

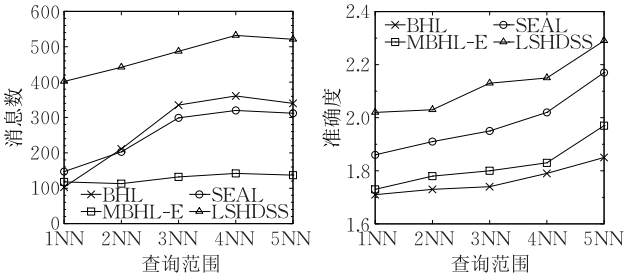
	BHL					
	$c=2$			$c=3$		
	消息数	准确度	索引大小/GB	消息数	准确度	索引大小/MB
MicroBlog	608	1.52	1.18	194	1.64	229
SynSet	714	1.63	2.10	216	1.72	575

因为 L 和数据集大小成正比,且索引大小和 L 成正比,所以 MicroBlog 的索引大小比 SynSet 的索引大小小一些.在分布式环境下,二元混合 LSH 的索引是均匀分布在各个计算节点上的,各个计算节点需要维护的索引大小是总大小的 $\frac{1}{cn}$.如对于 SynSet 数据集中 $c=3$ 时,平均单个计算节点需要维护的索引大小是 57.5 MB.如表 3 所示,当 $c=3$ 时的查询准确度略差于当 $c=2$ 时的准确度,但仍在可接受的范围.当 $c=3$ 时的通信代价和存储代价分别约为 $c=2$ 时的 $\frac{1}{3}$ 和 $\frac{1}{5}$.因此以小量的准确度换取大量的通信代价和存储代价是值得的.

7.3 (D,R,c)-NN 查询的性能

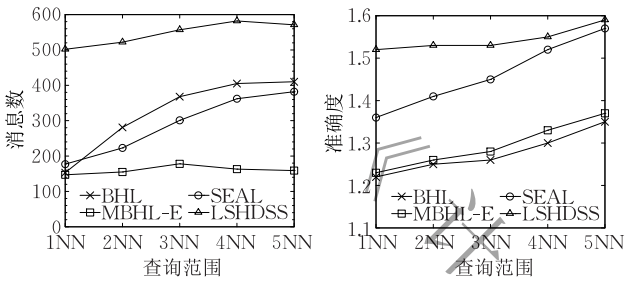
图 4 和图 5 显示了在 MicroBlog 和 SynSet 上对于(0.02,0.02)-NN,(0.04,0.04)-NN,(0.06,0.06)-NN,(0.08,0.08)-NN 和(0.1,0.1)-NN 的算法的消息数和准确度.受空间限制,图中将横坐标(0.02,0.02)-NN、(0.04,0.04)-NN、(0.06,0.06)-NN、(0.08,0.08)-NN 和(0.1,0.1)-NN 分别表示为 1NN、2NN、3NN、4NN 和 5NN.随着查询范围的扩大,BHL 和 SEAL 方法的消息数起初显著增加然后逐渐减少.对于 BHL 方法,随着查询范围的扩大,查询过程中会生成 LSH 查询转换,查询转换中的子查询导致消息数的增加.SEAL 方法随着查询范围的扩大,过滤的效率会降低,因此需要检查更多的倒排表和候选对象,最终导致消息数的增加.但从(0.08,0.08)开始满足查询条件的对象开始增加,算法更容易提前结束,因此消息数开始减少.相比较而言 MBHL-E 方法的通信代价比较稳定,MBHL-E 的消息数小于 SEAL 方法和 LSHDSS 方法,且 SEAL 的消息数小于 LSHDSS.根据图 4(b)和图 5(b),HLSH 的准确度是最好的,并且 MHLSH-E 的准确度要好于 SEAL 和 LSHDSS.图 6 显示了变化查询范围时查询的时间代价,可以看出随着查询范围的扩大,查询时间逐渐增加.因为在分布式环境中数据获取的时间主要是由网络通讯时间主导的,因此查询的时间代价具有与消息数相

似的变化趋势. MHLSH-E 的准确度保持较快的查询时间.



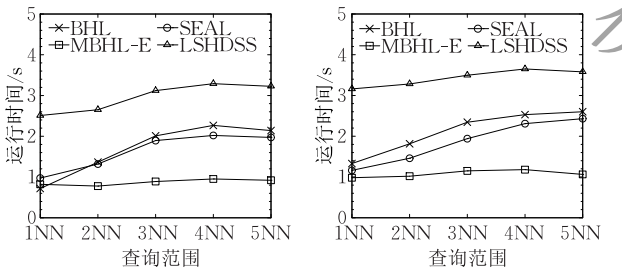
(a) 消息数变化情况　　(b) 准确度变化情况

图 4　在 MicroBlog 上变化查询范围时(D,R,c)-NN 查询性能



(a) 消息数变化情况　　(b) 准确度变化情况

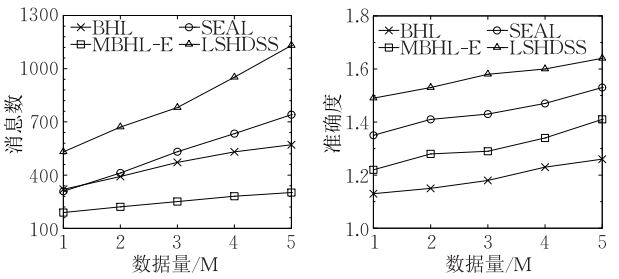
图 5　在 SynSet 上变化查询范围时(D,R,c)-NN 查询性能



(a) MicroBlog 上的运行时间　　(b) SynSet 上的运行时间

图 6　变化查询范围时(D,R,c)-NN 查询的时间代价

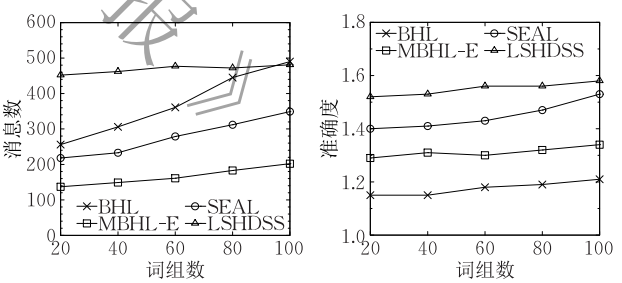
受限于真实数据集固定的数据量和词组数，实验中尝试变化人工数据集 Synset 的大小和词组数来验证数据集大小和数据维度对算法的影响. 图 7 是在 SynSet 上当数据集大小变化时查询消息数和查询准确度的变化趋势. 随着数据集大小的增加，SEAL 方法和 LSHDSS 方法的消息数迅速增多. 因为对于 SEAL 方法，倒排表中的数据对象平均数量和数据集大小成正比，对于 LSHDSS，其两阶段处理方法对数据集大小比较敏感. 因为定理 1 的性质 2，随着数据集大小的变化，BHL 和 MBHL-E 的通信代价相对稳定，且 MBHL-E 的消息数是最小的. 图 7(b)显示了 BHL 和 MBHL-E 的准确度要好于 SEAL 和 LSHDSS 的准确度. BHL 方法的准确度略好于 MBHL-E 方法，但这是以大量的消息数为代价的.



(a) 消息数变化情况　　(b) 准确度变化情况

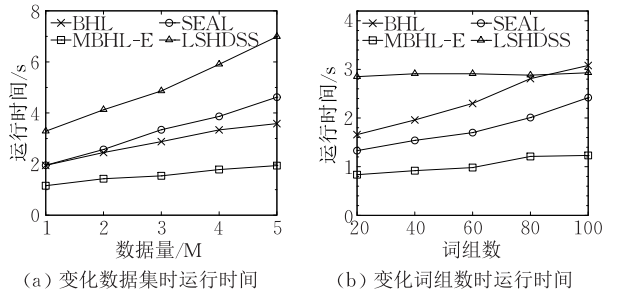
图 7　在 SynSet 上变化数据集大小时(D,R,c)-NN 的查询性能

图 8 是在 SynSet 上当平均词组数变化时(D,R,c)-NN 查询性能的变化. 很明显词组数类似于关系数据库中的维度数，由于 LSH 方法对维度数是不敏感的，因此如图所示随着词组数的增加 MBH-E 和 LSHDSS 方法的性能比较稳定；而对于 BHL 方法，因为随着词组数的增加 LSH 查询转换中的子查询数量明显增加，所以 BHL 的信息数会跟着增长. 对于固定的查询范围，SEAL 的复杂度是和词组数成正比的，因此随着词组数的增加，SEAL 方法的消息数线性增加. 如图 8(b)所示，BHL 的查询准确度是最好的，且 MBHL-E 的准确度要好于 SEAL 和 LSHDSS. 图 9 显示了变化数据集大小和词组数量时(D,R,c)-NN 查询时间代价的变化趋势. 从图 9(a)可以看出随着数据集的增大查询时间基本程线性增长的趋势，根据定理 1，BHL 和 MBHL-E 需要检查的数据对象更多以保证准确度，



(a) 消息数变化情况　　(b) 准确度变化情况

图 8　在 SynSet 上变化词组数量时(D,R,c)-NN 查询性能

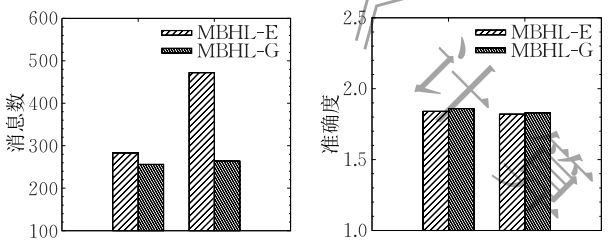


(a) 变化数据集时运行时间　　(b) 变化词组数时运行时间

图 9　在 SynSet 上变化数据集大小和词组数量时(D,R,c)-NN 查询的时间代价

然而,因为增多检查的数据对象一般可在同一个哈希桶中完成,因此这两个算法时间代价变化不大.从图 9(b)可以看出基于 LSH 的方法更能适应高维数据,LSHDSS 和 MBHL-E 的时间代价对词组数的增加不敏感,且 MBHL-E 方法保持最低的时间代价.

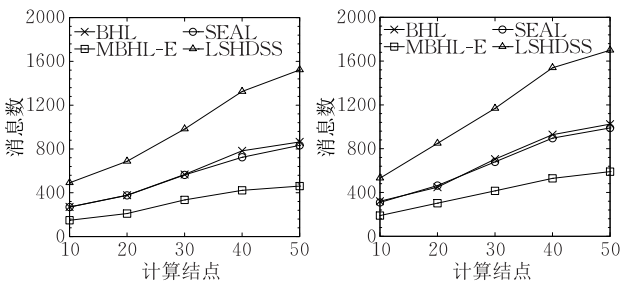
图 10 是在 MicroBlog 上 MBHL-E 和 MBHL-G 的查询性能的比较. MBHL-G 是查询敏感的方法,在一个区间范围内,查询越频繁则混合 LSH 就越密集,且如果查询是均匀分布的话 MBHL-E 和 MBHL-G 是等价的.如图 10 所示,一类查询的查询范围服从自然分布(normal),另一类查询的查询范围的分布偏移度是 0.8 (skewed),对于偏移分布的查询,MBHL-G 的消息数约为 MBHL-E 的$\frac{1}{2}$,而它们的准确度是比较接近的.



(a)消息数 (b) 准确度

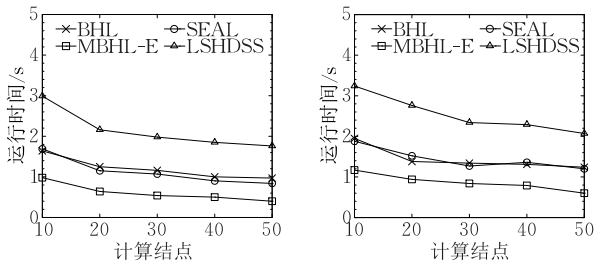
图 10 在 MicroBlog 上 MBHL-E 和 MBHL-G 的性能比较

图 11 和图 12 是随着计算节点的增加,(D,R,c)-NN 查询通信代价和时间代价的变化.从图 11 可以看出,随着计算节点的增加,各个算法的通信代价都有所增加.其中 LSHDSS 需要分别两次查询,然后对查询结果进行连接,通信代价增加得比较多.其它 3 种方法基本保持次线性速度增加.从图 12 可以看出,随着计算节点的增多,查询时间逐渐减少,MBHL-E 方法表现出了较好的可伸缩性.时间代价减少最明显是在从 10 个节点到 20 个计算节点时,可见根据不同的数据量和数据分布都有相应的最高性价比的计算节点数量.



(a) MircroBlog上的消息数变化 (b) SynSet上的消息数变化

图 11 变化计算节点数量时(D,R,c)-NN 查询的通信代价

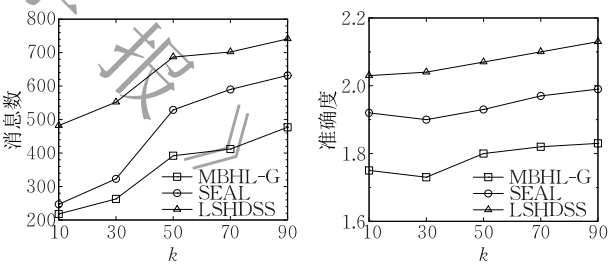


(a) MircroBlog上的运行时间 (b) SynSet上的运行时间

图 12 变化计算节点数量时(D,R,c)-NN 查询的时间代价

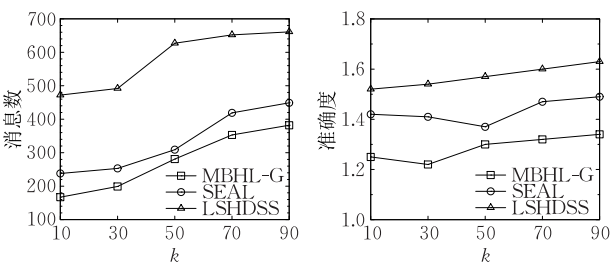
7.4 kNN 查询性能

如算法 2 所述,kNN 查询主要是由多个(D,R,c)-NN 查询组成,因此 kNN 查询与(D,R,c)-NN 查询有类似的特性.图 13 和图 14 是在 MicroBlog 和 SynSet 上变化 k 的情况下 MBHL-G,SEAL 和 LSHDSS 算法的消息数和准确度.如图 13(a)和图 14(a)所示,随着 k 的增加,消息数也逐渐增加.当算法需要扩大查询范围来获取更多的候选对象时,通信代价自然迅速增长.如图 13(b)和图 14(b)所示,在扩大查询范围之前,随着 k 的增加,查询准确度有提高的趋势,但扩大查询范围之后查询准确度有一定程度的降低.图 15 显示了 kNN 查询的时间代价,随着 k 的增加,查询时间逐渐增加,在常用的 k 值范围内,MBHL-G 方法保持最小的时间代价.



(a) 消息数 (b) 准确度

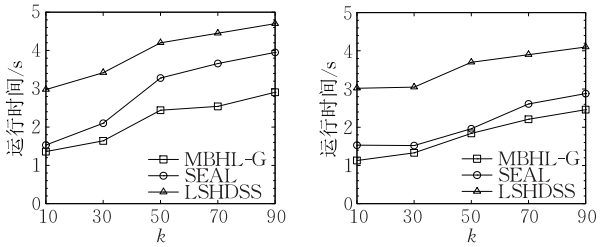
图 13 在 MicroBlog 上 kNN 查询的性能



(a) 消息数 (b) 准确度

图 14 在 SynSet 上 kNN 查询的性能

因为 LSHDSS 方法包含有负载均衡策略,因此,如图 16 所示实验中比较了 MBHL-G 算法和 LSHDSS 的负载均衡的情况.因为查询敏感的负载



(a) MicroBlog上的运行时间　　(b) SynSet上的运行时间

图 15　kNN 查询的时间代价

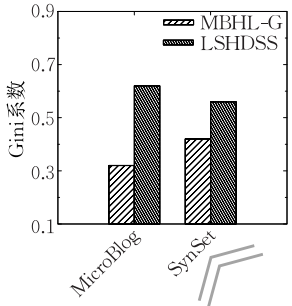


图 16　负载均衡的比较

均衡策略使得 MBHL-G 的 Gini 系数远小于 LSHDSS 的，且维持在 0.5 以下，可以说明算法的负载均衡方法的有效性.

8　相关工作

本文的查询方法主要面向基于欧氏距离的数值型数据和基于 Jaccard 距离的字符型数据的二元混合数据类型. 当前针对这类数据的研究热点有面向图像-文本数据类型的相似性查询[10,21]和面向空间-文本数据类型的相似性查询[7,13,22-23]. 对于图像-本文数据，文献[10]提出了针对跨领域多模态数据的相似性查询框架，该框架将带有标签的图片和文档映射到维基百科的概念空间中，每个数据对象可以用概念向量表示. 当有查询时，通过查询对象的最近邻资源计算出查询的概念向量，基于该概念向量查询出相似结果. 为了方便检索和查询 Web 图片，文献[21]提出了图片文本自动对齐方法，该方法首先从网页中获取图片文本对，然后根据图片的视觉特征对图片进行聚类并对同类中的文本进行相关性排序，最后基于随机漫步方法进一步优化文本的相关性顺序. 针对于空间文本数据. SEAL[22]是一种面向空间文本数据的相似性查询框架，SEAL 采用“过滤-确认”结构，在过滤阶段 SEAL 将空间信息转化为基于网格的特征，并采用阈值敏感的过滤算法提高查询效率，进一步将空间信息和文本内容整合在一起生成混合特征提高过滤效率. IQ-tree[7]是一种面向空间文本流数据针对布尔范围连续查询(BRC)的混合索引，IQ-tree 集成了适用于空间数据的 Quad-tree 结构和适用于文本数据的倒排索引结构，IQ-tree 的基本思想是将查询存储在索引结构中，当有流数据到达时则通过索引结构查询是否有被满足的查询. 文献[13]针对空间文本数据的逆向 kNN 查询(RkNN)提出了解决方案，构建了 IUR-tree 混合索引. IUR-tree 将文本信息整合到 R-tree 索引，通过计算最小边界和最大边界从而有效过滤不相关的数据对象并最终得到结果.

为了适应数据规模的增大，提高查询可伸缩性，分布式相似性查询方法[20,24-26]成为了近年来的研究热点. 文献[20]提出来在结构化拓扑网络上处理基于 LSH 面向高维数据的相似性查询方法 LSHDSS. LSHDSS 设计两种位置敏感数据映射方法使得相似的数据存储在同一个计算节点上或者相邻的计算节点上从而保持负载均衡的同时减少查询过程的通信代价. SimPeer[24]使用 iDistance 的基本原理在层次化的 P2P 网络拓扑结构中处理基于度量空间的相似性查询. MCAN[25]利用枢纽点将高维度量空间的数据映射到低维的向量空间中，然后使用 CAN 拓扑结构存储和路由查询处理. 枢纽点是以集中式方法基于样本数据，然后分发到各个计算节点上. I2RS 方法[26]提出一种指定空间文本相关性的情况下查询相似图片的分布式推荐系统，该系统使用一种支持度量距离的索引 SPB-tree[27]来提高查询效率，并且在 BS 模型中的 Server 端采用分布式环境提高查询的可伸缩性.

综上所述，现有的面向大规模的混合类型数据相似查询方法，为了提高查询效率多采用基于树型索引、网格索引和哈希索引等方法. 本文提出了基于 LSH 的面向混合类型数据的相似性查询方法. 该方法保证查询的准确性和有效性的同时，避免了树型结构和层次结构在分布式环境中的较高的维护代价. 因此，与本文较相关的方法是 LSHDSS 和 SEAL 方法.

9　结束语

本文提出了基于 LSH 面向二元混合类型数据的相似性查询方法，给出了处理可变查询范围的 (D,R,c)-NN 查询方法和约近 kNN 查询算法. 通过理论分析，证明了查询方法和查询算法的准确性

和有效性.为了验证算法的可伸缩性,利用 LSH 结构不依赖于全局信息的特性,将查询方法和查询算法部署在 P2P 的分布式平台中.最后,通过基于真实数据和合成数据的大量实验,进一步验证了算法高效性.为了提高查询方法的通用性,下一步工作将考虑两方面的相似性查询的扩展:从内容上将考虑更多的数据类型,如时间数据和链接数据等;从形式上将考虑流数据和不确定数据等.

致 谢 在此,我们向对本文的工作给予支持和建议的同行,尤其是向东北大学计算机科学与工程学院软件研究所 407 教研室的老师和同学表示感谢!

参 考 文 献

[1] Arya S, Mount D, Netanyahu N, et al. An optimal algorithm for approximate nearest neighbor searching fixed dimensions. Journal of the ACM, 1998, 45(6): 891-923

[2] Shakhnarovich G, Darrell T, Indyk P. Nearest-neighbors methods in learning and vision: Theory and practice. Pattern Analysis and Applications, 2008, 11(2): 221-222

[3] Matei B, Shan Y, Sawhney H, et al. Rapid object indexing using locality sensitive hashing and joint 3D-signature space estimation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2006, 28(7): 1111-1126

[4] Chen Lisi, Cui Yan, Cong Gao, Cao Xin. SOPS: A system for efficient processing of spatial-keyword publish/subscribe. Proceedings of the VLDB Endowment, 2014, 7(13): 1601-1604

[5] Zhong Ruicheng, Li Guoliang, Tan Kian-Lee, et al. G-Tree: An efficient and scalable index for spatial search on road networks. IEEE Transactions on Knowledge and Data Engineering, 2015, 27(8): 2175-2189

[6] Feng Jianhua, Wang Jiannan, Li Guoliang. Trie-join: A Trie-based method for efficient string similarity joins. The VLDB Journal, 2012, 21(4): 437-461

[7] Chen Lisi, Cong Gao, Cao Xin. An efficient query indexing mechanism for filtering geo-textual data//Proceedings of the ACM SIGMOD Conference. New York, USA, 2013: 749-760

[8] Li Z, Lee K, Zheng B, et al. IR-Tree: An efficient index for geographic document search. IEEE Transactions on Knowledge and Data Engineering, 2011, 23(4): 585-599

[9] Fan Ju, Li Guoliang, Zhou Lizhu, et al. SEAL: Spatio-textual similarity search. Proceedings of the VLDB Endowment, 2012, 5(9): 824-835

[10] Chen Liu, Wu Sai, Jiang Shouxu, Tung A. Cross domain search by exploiting Wikipedia//Proceedings of the IEEE 28th International Conference on Data Engineering. Washington, USA, 2012: 546-557

[11] Wang Yanfei, Wu Fei, Song Jun, et al. Multi-modal mutual topic reinforce modeling for cross-media retrieval//Proceedings of the ACM International Conference on Multimedia. Orlando, USA, 2014: 307-316

[12] Lowe D. Distinctive image features from scale-invariant key points. International Journal of Computer Vision, 2004, 1(6): 91-110

[13] Lu Jiaheng, Lu Ying, Cong Gao. Reverse spatial and textual k nearest neighbor search//Proceedings of the ACM SIGMOD Conference. Athens, Greece, 2011: 349-360

[14] Chen L, Cong Gao, Cao Xin, Tan K. Temporal spatial-keyword Top-k publish/subscribe//Proceedings of the International Conference on Data Engineering. Seoul, Korea, 2015: 255-266

[15] Li Yuchen, Bao Zhifeng, Li Guoliang, Tan K. Real time personalized search on social networks//Proceedings of the International Conference on Data Engineering. Seoul, Korea, 2015: 639-650

[16] Gao J, Jagadish H, Lu W, Ooi B. DSH: Data sensitive hashing for high-dimensional k-NN search//Proceedings of the ACM SIGMOD Conference. Utah, USA, 2014: 1127-1138

[17] Gan J, Feng J, Fang Q, Ng W. Locality-sensitive hashing scheme based on dynamic collision counting//Proceedings of the ACM SIGMOD Conference. Scottsdale, USA, 2012: 541-552

[18] Datar M, Immorlica N, Indyk P, Mirrokni V. Locality-sensitive hashing scheme based on p-stable distributions//Proceedings of the Symposium on Computational Geometry. New York, USA, 2004: 253-262

[19] Broder A, Glassman S, Manasse M, Zweig G. Syntactic clustering of the Web. Computer Networks, 1997, 29(8): 1157-1166

[20] Haghani P, Michel S, Aberer K. Distributed similarity search in high dimensions using locality sensitive hashing//Proceedings of the 12th International Conference on Extending Database Technology. Saint Petersburg, Russia, 2009: 744-755

[21] Zhou Ning, Fan Jianping. Automatic image—text alignment for large-scale Web image indexing and retrieval. Pattern Recognition, 2015, 48(1): 205-219

[22] Fan Ju, Li Guoliang, Zhou Lizhu, et al. SEAL: Spatio-textual similarity search. Proceedings of the VLDB Endowment, 2012, 5(9): 824-835

[23] Hu Jun, Fan Ju, Li Guo-Liang, Chen Shan-Shan. Top-k fuzzy spatial keyword search. Chinese Journal of Computers, 2012, 35(11): 2237-2246(in Chinese)
(胡骏, 范举, 李国良, 陈姗姗. 空间数据上 Top-k 关键词模糊查询算法. 计算机学报, 2012, 35(11): 2237-2246

[24] Doulkeridis C, Vlachou A, Kotidis Y, Vazirgiannis M. Peer-to-peer similarity search in metric spaces//Proceedings of the International Conference on Very Large Data Bases. Vienna, Austria, 2007: 986-997

[25] Falchi F, Gennaro C, Zezula P. A content addressable network for similarity search in metric spaces//Proceedings of the Databases, Information Systems and Peer-to-Peer Computing. Seoul, Korea, 2005: 98-110

[26] Chen Lu, Gao Yunjun, Xing Zhihao, et al. I2RS: A distributed geo-textual image retrieval and recommendation system. Proceedings of the VLDB Endowment, 2015, 8(12): 1884-1895

[27] Chen Lu, Gao Yunjun, Li Xinhan, et al. Efficient metric indexing for similarity search//Proceedings of the International Conference on Data Engineering. Seoul, Korea, 2015: 591-602

ZHU Ming-Dong, born in 1985, Ph. D., lecturer. His research interests include big data analysis and similarity query.

SHEN De-Rong, born in 1964, professor, Ph.D. supervisor. Her research interests include Web data processing and distributed database.

KOU Yue, born in 1980, associate professor. Her research interests include entity resolution and Web data management.

NIE Tie-Zheng, born in 1980, associate professor. His research interests include data quality and data integration.

YU Ge, born in 1962, Ph. D., professor. His research interests include distributed and parallel database, OLAP and data warehouse, data integration, graph data management, etc.

Background

Because of a wide range of applications, nearest neighbor (NN) queries have been studied in many different fields, and with the popularity of location based service, sensor network, face recognition, and so on, hybrid data prevail. This has led to a lot of research on similarity query algorithms on binary hybrid data. What's more, with the development of cloud computing and the demand of processing large data, distributed algorithms become a trend. For tree structure and hierarchical structure methods, maintenance cost is very high. Hence, there is an increasing demand to propose novel methods for similarity queries on large scale hybrid data.

This work is supported by the National Basic Research 973 Program of China (Grant No. 2012CB316201) and the National Natural Science Foundation of China (Grant Nos. 61472070, 61672142) and the University Key Scientific Research Program of He′nan Province (Grant No. 18B520011).