计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.190800143

不确定时态数据 Top-k 查询

韦建华　　许建秋
南京航空航天大学计算机科学与技术学院　南京 210016
(jianhua@nuaa.edu.cn)

摘　要　时态数据在医疗、经济和电子商务等领域有着广泛的应用。由于时间的测量技术不精确等因素,时态数据具有不确定性。文中针对该数据进行研究,处理 Top-k 查询,即返回与查询点相交的 k 个权值最大的数据,该权值是根据数据权值和相交概率按一定规则组合计算所得。为有效解决该查询问题,提出了一个基于关系模型和辅助结构的 2D R-tree 结构,其中关系模型用于管理所有区间数据的 R-tree,辅助结构用于管理 R-tree 中每个节点内部数据权值的大小关系。基于该结构,提出了按权值的降序访问数据的查询算法。从根节点开始遍历 R-tree,对于与查询点相交的节点,根据辅助结构中存储的信息找到数据权值最大的项,将它确定为下一个访问对象。实验使用数据规模在 30 万到 1000 万的合成数据集,以及包括大约 320 万条的航班信息的真实数据集。在可扩展数据库 SECONDO 系统下,将所提方法与无索引方法、R-tree 和区间树方法在性能上进行比较,并以平均 I/O 访问次数和 CPU 时间作为性能的评判指标。实验结果表明,在 1000 万条的数据规模下,所提方法优于对比方法 2～3 个数量级。通过将实验返回的 k 个结果的概率与权值和实际相交数据的概率和权值作比较可以发现,实验返回的 k 个结果的概率与权值均靠近实际相交数据的概率和权值的最大值,因此所提算法可行且有效。

关键词:时态数据;不确定性;top-k
中图法分类号　TP391

Efficient Top-k Query Processing on Uncertain Temporal Data

WEI Jian-hua and XU Jian-qiu
Department of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China

Abstract　Temporal data is widely used in many applications such as medical, economic and e-commerce. The uncertainty is mainly caused by factors such as inaccurate measurement techniques. This paper studies top-k queries over uncertain temporal data. Such a query returns Top-k intervals with the largest scores which are calculated by a function combining the original weight of the data and the probability of intersection with the query data. To answer the query efficiently, this paper proposes a 2D R-tree based on the relational model and auxiliary structures. The relational model is used to manage all intervals, and the auxiliary structure is used to manage the order of the weights of each node in the R-tree. Based on the proposed index structure, a query algorithm for accessing data in descending order by weights is proposed. It traverses the R-tree from the root node. For each node that intersects with the query point, the item with the largest weight in it can be found according to the information stored in the auxiliary structure, and it is determined as the next accessed object. This paper uses synthetic datasets with data sizes ranging from 300000 to 10 million, and a real dataset of a flight information with size of 3.2 million. In the extensible database system SECONDO, the proposed method is compared with the unindexed method, R-tree, and interval tree, and the average I/O access times and CPU time are used as the indicators of the experimental results. The experimental results show that the proposed approach outperforms baseline methods by 2 to 3 orders of magnitude using 10 million intervals. Comparing the probabilities and weights of the k results with the results of all intersecting data, it is found that the probabilities and weights of the k results are close to the maximum value of the actual intersecting data, so the proposed algorithm is feasible and effective.

Keywords　Interval data, Uncertainty, Top-k

到稿日期:2019-08-29　返修日期:2019-10-27　　本文已加入开放科学计划(OSID),请扫描上方二维码获取补充信息。
基金项目:国家自然科学基金(61972198)
This work was supported by the Natural Science Fundation of China(61972198).
通信作者:许建秋(jianqiu@nuaa.edu.cn)

1 引言

随着时空数据库与信息技术的深入发展，时态数据处理技术已经成为许多新一代数据库与信息系统的关键技术。快速检索并提取出时态数据中有用的信息，对数据的分析和总结工作有极大的帮助。实际应用中，不确定时态信息产生的原因有多个，如记录数据时间的粒度与事件发生的确切时间粒度不一致或时间测定技术的不精确等。这些因素会导致数据存在不确定性，且这种不确定性会传播到查询中。

本文研究不确定时态数据的 Top-k 查询，即给定查询时间且系统报告满足条件的 k 元组：1）k 元组中与查询时间相交的数据；2）k 元组中具有最大权值的数据。为了更好地理解研究的问题，给出以下示例。

例 1 分析某公司一年内签订的工程合同。数据集由许多记录组成，每一条记录包括 4 项内容：该工程预计的起始和终止时间，完成该工程所需的确定工时，以及工程款。需要求解的问题是：查找在 $t=50$ 时刻，工程存在的可能性大并且工程款较大的前 2 个工程项目。具体示例如图 1 所示。

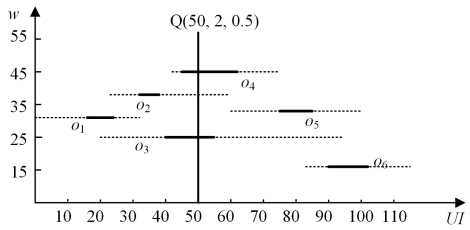

图 1 不确定时态数据 Top-k 查询示例

Fig. 1 Example of Top-k query

以 o_1 为例，[0，32]表示该工程从开始到结束的时间范围，10 是完成它所需的时间长度。该工程可能在这一范围的任意时间段完成，如[5，15]，[15，25]。

时态数据已广泛存在于许多应用领域中，如时间和多版本数据库中的交易时间和有效时间[1,2]。目前，与确定时态数据相关的研究有相交[3]、stabbing[4]和连接[5]等操作。本文研究的是长度和范围确定，但起始和终止点不确定的时态数据。对于这样的数据，它与查询数据相交的结果是一个带有概率性的结果。

本文的主要贡献如下：

（1）定义不确定性时态数据的相关概念，给出不确定性的计算方法，并提出不确定性时态数据的 Top-k 查询；

（2）基于 2D R-tree 提出了一种新的结构，即按 R-tree 中节点的权值大小来管理不确定时态数据；

（3）设计基于该索引的深度优先遍历算法，按权值大小来优先处理不确定时态数据，以提高查询效率；

（4）使用真实数据集和合成数据集，在效率和可扩展性方面进行对比实验，结果表明了本文提出的算法优于其他方法。

2 相关工作

2.1 时态数据查询

目前，很多工作中的时态数据都是基于间隔数据。例如，Layer 等[6]研究了两组基因组间隔之间的交叉计数。通过为每个间隔定义权重，stabbing-max 查询返回包含查询点的最大权重间隔[7]。Sfakianaki 等在云密钥值存储中研究了区间查询[8]。为丰富数据表示并支持不同类型数据间隔的应用程序，Xu 等[9]研究了带类型间隔的查询，考虑到关键字的相似性，其返回的每个对象的类型与查询完全匹配并具有最大权重。Li 等[10]研究了线性域中间隔上的分散点的问题，即给定一条线上排序的 n 个成对不相交的间隔，在每个间隔中找到一个点，使得这些点的最小成对距离最大化。

以上研究的数据都是起始点和终止点确定的间隔，而现实应用中有些数据的起始点和终止点并不能确定，因此需要对不确定间隔数据进行研究。Cowley 等提出了新的不确定时间戳表示方法，并扩展了 Allen 的 13 种区间关系来表示不确定的时间区间，还提供了相应的运算符来操作这些确定的和不确定的时间区间[11]。从不确定性产生的角度考虑，Plesniewicz 等[12]提出了不完全信息下区间值模糊软集的数据分析方法，其中涉及在缺失条目的百分比高于阈值时忽略不完整数据，并且在缺失条目的百分比低于阈值时忽略不完整信息的填充方法。为解决模糊信息环境中严重的信息丢失问题，Wu 等[13]研究了一种新的多属性群决策（Multi-Attribute Group Decision Making，MAGDM）模型，提出了一种概率区间值犹豫模糊信息聚合方法，该方法结合了阿基米德范数的思想。关于不确定时态数据，Zhang 提出了基于间断区间的时态知识表示[14]；Lin 等给出了不确定时态信息的表示方法[15]；Katerina 等提出了一个时序概率数据模型，引入了谱系感知时间窗口，提出了一种将区间与谱系表达式绑定的机制，能够直接过滤不相关的间隔并最终确定输出谱系表达式[16]。

2.2 时态数据索引

当待查询数据集过大时，为提高时态数据的查询效率，需引入合适的索引结构。常见的时态数据索引包括区间树、线段树和优先级搜索树等[17-19]。通过组合辅助结构，区间树有一个变体结构——interval B^+-tree[20]，其内部节点的结构类似于 B^+-tree 的内部节点。Mei 等[21]提出了一种新的时域数据分割混合索引 SHB^+-tree，首先根据时间顺序将所存储的时间表中的时间数据分离成片段，然后在每个段中构造混合索引，其是时间索引和对象索引的组合，并且时间数据由它们共享。为有效报告与给定查询间隔相交的所有间隔数据，Kriegel 等[22]提出了一种有效支持交叉查询的新方法——Relation Interval tree。

常见的时态索引是在传统数据库索引技术 B-tree 的基础上发展而来的 R-tree，以及经过不断改进的 R*-tree 和 G-tree 等。Bliujute 等提出的 4D R-tree 索引方法[23]是将带变元的时态数据变换为不带变元的数据，从而消除了时间变元，再利用 R-tree 实现时态数据索引，进而实现了带时间变元的时态数据的索引。但是 4D R-tree 也有不足之处，即查询成本代价较高、查询时需要大量的磁盘 I/O 操作等。因此，Zeng 等[24]又提出了一种改进的 4D R-tree，根据有效时间截止值和事务时间截止值是否为时间元，将时态数据分为 4 种类型，

对 4 种类型的双时态数据进行数据变换,消除其中的时间变元,使其对应的时态区域为固定矩。这个方法弥补了 4D R-tree 的不足,提高了数据查询的效率,减少了磁盘 I/O 操作次数,也避免了不符合实际情况的搜索结果。

综上所述,现有的不确定性时态数据查询方法对不确定性表示与计算的研究较深,研究者们已经提出了许多较为成熟的解决方案。然而,对于不确定性与权值影响下的 Top-k 查询,目前的研究尚不能满足需求。因此,本文研究了查询概率和权值相结合的时态数据。本文以 R-tree 的变体为主要方法,将不确定间隔表示为 x 轴,将其对应的权值作为 y 轴,而不是以间隔的起始点和终止点范围作为二维 R-tree 的 x 轴和 y 轴,用它们之间的面积表示不确定间隔,这样既管理了权值属性,也避免了无效区间的产生。

3 问题描述

不确定性时态数据 Top-k 查询是返回与给定查询相交可能性大且权值最大的 k 个数据。不确定时态数据以间隔形式表示,为了明确所解决的问题,下面给出相关定义。

定义 1(不确定时态数据) 间隔的起始点在一定的范围内,长度是确定的数据被称为不确定时态数据。

$$I=\{(s,e)\mid s,e\in \mathrm{R}^+,s<e\}$$

$$O=\{(i,l,w)\mid i\in I,l\in \mathrm{R}^+,w\in \mathrm{R}^+\}$$

其中,I 是不确定时态数据的范围,l 是它的长度,w 是它的权值。

定义 2(Top-k 查询) $Q=(x,k,r)$,其中 x 是查询点,k 是返回结果的目标个数,r 是用户自定义系数,r 的取值范围为[0,1]。用户可以通过改变 r 的大小来选择返回偏向于权值大的数据或偏向于相交概率大的数据。

当查询 Q 与不确定时态数据相交时,还需要计算它与不确定时态数据相交的概率。由此给出定义 3。

定义 3(不确定时态数据相交概率) 对于不确定时态数据集 O 中任意一个长为 l、范围为 L 的不确定时态数据 o,它与查询 Q 相交是一个概率事件,将其概率记为 $p(Q,o)$。

$$p(Q,o)=\begin{cases}(Q.x-o.i.s)/(o.i.e-o.i.s), & o.i.s<Q.x\leqslant o.i.s+o.l\\ o.l/(o.i.e-o.i.s), & o.i.s+o.l\leqslant Q.x\leqslant o.i.e-o.l\\ (o.i.e-Q.x)/(o.i.e-o.i.s), & o.i.e-o.l\leqslant Q.x<o.i.e\\ 1, & o.i.e-o.l<Q.x<o.i.s+o.l\end{cases}$$

点与不确定间隔数据相交分为以下 4 种情况,其示意图如图 2 所示。

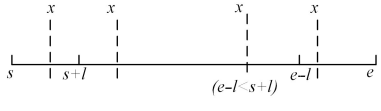

图 2 点与不确定间隔数据相交的 4 种情况

Fig. 2 Four cases of intersection of point with interval

结合例 1,可得与查询数据相交的对象是 o_2,o_3,o_4,且 $p(Q,o_2)=0.25$,$p(Q,o_3)=0.57$,$p(Q,o_4)=0.4$。

定义 4(不确定时态数据 Top-k 查询函数)

$$f(Q,o)=(1-Q.r)*o.w_n+Q.r*p(Q,o)$$

其中,$o.w_n$ 是权值归一化处理后的值,归一化公式为:

$$o.w_n=(o.w-w\_min)/(w\_max-w\_min)$$

其中,w_min 和 w_max 分别表示数据集中权值的最大值和最小值。$Q.r$ 的大小会影响最终的 k 个结果。当 $Q.r=0.1$ 时,$f(Q,o_2)=35.125$,$f(Q,o_3)=35.78$,$f(Q,o_4)=40.54$,例 1 返回的结果是 o_3,o_4。当 $Q.r=0.9$ 时,$f(Q,o_2)=4.125$,$f(Q,o_3)=3.013$,$f(Q,o_4)=4.95$,例 1 返回的结果是 o_2,o_4。

定义 5(不确定时态数据 Top-k 查询) 给定一个查询 Q,返回含 k 个元组的数据集 $O'\subset O$,使得 $\forall o'\in O'$:

(1) $Q.x\in o'.i$;

(2) $\nexists o\in O\backslash O'$:$Q.x\in o.i\wedge f(Q,o)>f(Q,o^-)$。

例 2 对于例 1 中的数据,给定查询 $Q=(50,2,0.5)$,可根据定义 4 计算出 $f(Q,o_2)=19.625$,$f(Q,o_3)=12.785$,$f(Q,o_4)=22.7$,最终返回的结果是 o_2,o_4。

4 不确定时态数据索引

本文采用 2D 数据表示不确定时态数据和权值,并建立 2D R-tree 索引结构,从而可同时对间隔数据和权值进行管理。每个维度的最小值和最大值分别设定为[s,e]和[$w-\varepsilon$,$w+\varepsilon$],建立 2 维矩形框。

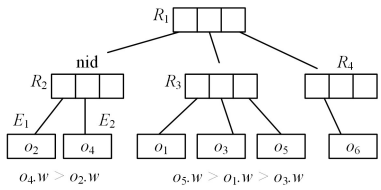

图 3 2D R-tree

Fig. 3 2D R-tree

查询目的是返回与给定数据相交的可能性最大并且权值较大的数据,所以对于一个时态数据,它的权值越大就越可能是查询结果。因此,文中提出添加一个辅助结构来存储 R-tree 节点的权值关系,以实现按权重的降序对 R-tree 中节点的条目进行遍历来提高查询效率,即改变以往遍历 R-tree 节点中项的顺序,不再按顺序访问而是按权重的降序遍历 entries。以图 3 中的数据为例,对于 R_2 中的数据,标准的 R-tree 的访问顺序是 E_1,E_2,但是若按权值的大小排序,访问顺序则是 E_2,E_1。将这个结构称为 Relation R-tree(RR-tree),此处的 Relation 指存储关系的一个结构。下文将用 B-tree 和 Array 方法来管理这个关系,相应的结构记为 BR-tree 和 AR-tree。

4.1 BR-tree

为了能够按权重的降序对 R-tree 中节点的条目进行遍历,通过遍历 R-tree 来维护一个关系表,该关系表管理的是节点的 NodeId 与其对应的条目的索引号。它的模式是 EntryRel(TupleId:Int,NodeId:Int,EntryOrder:List),其中 TupleId 是对应节点被访问的顺序;NodeId 是对应节点的 id 号;

EntryOrder 维护的是一个有序对，每一个有序对由 NodeId 和权值组成，这些有序对按权值从大到小的顺序被存储。利用 B-tree 来管理这个关系表时，记 B-tree 中每一个节点为 $node(NodeId, L)$，$L=\langle(w_1, index_1), \cdots, (w_k, index_k)\rangle$。当遍历 R-tree 时，针对当前访问的节点，可以根据节点的

id 找到其在 B-tree 中的位置，根据 B-tree 中存储的有序链表按权值降序选择下一个访问的项。

例 3　当遍历图 3 中的 R-tree，可以根据每个节点的访问顺序获得一个 relation，再根据 relation 中的 NodeId 构建 B-tree，如图 4 所示。当遍历到 NodeId 为 nid 的节点时，找到 B-tree 中关键字为 nid 的节点，获取对应的 EntryOrder 以选择下一个访问目标，此处访问顺序为 E_2，E_1。

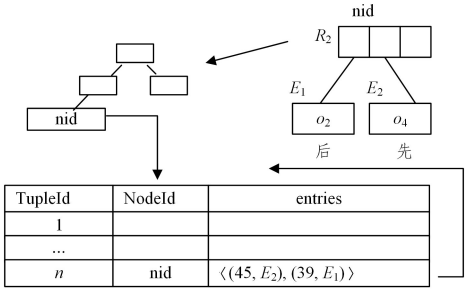

图 4　BR-tree

Fig. 4　BR-tree

4.2　AR-tree

结合 B-tree 管理权值的方法，每当访问到 R-tree 的一个节点时，就需要遍历一次 B-tree 才能找到 B-tree 中与当前节点对应的节点，其时间复杂度为 O(log N)。为了能够在遍历 R-tree 时在关系中用常数时间找到与当前节点对应的权值大小的信息，本文提出利用数组来存储关系表，将该数组称为关系数组，用 AR-tree 表示。创建一个大小与 R-tree 节点数目相同的数组，以节点的 NodeId 为数组下标，关系中的 TupleId 为数组内容进行映射。映射的函数为 $f\_array: NodeId \rightarrow TupleId$，即将遍历 R-tree 时所得节点的 NodeId 与其访问顺序相对应。这样当访问到某个节点时，可以根据其 NodeId 直接定位到它在数组中的位置，从而获取它在 relation 中的位置。

例 4　构建如图 5 所示的数组。假设当前 NodeId 为 nid 的节点是第 n 个被访问的，可直接定位到 AR-tree [nid]并且获取到它的内容是 n，然后找到 relation 中 TupleId 为 n 的元组以获取对应的 EntryOrder，选择下一个访问节点 E_2。

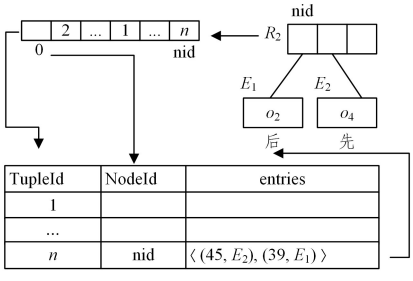

图 5　AR-tree

Fig. 5　AR-tree

5　不确定时态数据 Top-k 查询算法

5.1　查询框架

不确定时态数据 Top-k 查询是返回在给定数据库 O 内，与查询数据 $Q.x$ 相交概率大，且权值大的前 k 个数据。本文利用 RR-tree 处理该查询，采用深度优先方法由根节点开始遍历 RR-tree。设置一个大小为 k 的小顶堆 H，存储与查询数据相交的 k 个候选结果，堆中元素在查询过程中保持更新。在访问 RR-tree 中的每个节点时，以与查询数据相交的对象的 MBR 计算出的 score 值作为 RR-tree 的剪枝条件。在遍历 RR-tree 后，保存在小顶堆 H 中的 k 个间隔数据就是与查询数据相交且 score 值最大的 k 条数据，即为 Top-k 查询的结果。在查询过程中充分利用已知限制条件遍历 RR-tree，并从候选节点中裁剪掉不可能的部分，以提高查询效率，具体步骤如图 6 所示。

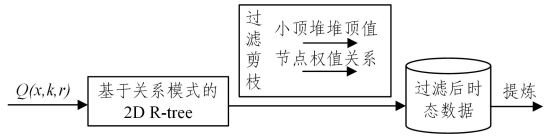

图 6　Top-k 查询过程

Fig. 6　Process of Top-k query

5.2　过滤过程

Top-k 查询如算法 1 所示，主要步骤如下。

(1)设定大小为 k 的小顶堆保存当前查询结果，队列 Q 保存查询节点，将 RR-tree 根节点入队列。

(2)对于每个出队列的节点 N，首先根据 relation 中存储的信息按权值降序访问 N 中的项，并且判断该项是否与查询 $Q.x$ 相交。小顶堆 H 的堆顶元素存储当前堆中与查询点相交且 $score$ 值最小的间隔。若当前堆的大小为 k，则将 $score$ 值设为 min，进一步计算项的 MBR 与查询点相交的 score 值，将 $score$ 值小于 min 的项过滤。若当前堆的大小小于 k 则转入步骤(3)。

(3)对于步骤(2)中满足条件的项，如算法 2 所示，若当前节点为非叶节点，则将满足条件的项以 relation 中存储的按权值从大到小的顺序入队列；若节点为叶子节点，判断项指向的间隔数据是否与 $Q.x$ 相交，若相交则进一步计算相交的 $score$ 值。若 $score$ 值大于堆顶值 min，则将其插入小顶堆，并更新 min 作为之后的判断条件。

(4)在遍历 RR-tree 后，H 中的所有间隔为与查询点相交概率大且权值大的 k 个间隔。

当查询数据与当前节点的 MBR 相交时，需要根据查询函数计算 $score$ 值。首先需要计算它们相交的概率，这里分为叶子节点和非叶子节点两种情况，对于非叶子节点，只要查询数据与当前矩形框相交，则其概率就为 1；对于叶子节点，其概率根据定义 3 分情况计算。

算法 1　Top-k 查询

输入：查询数据 Q.x，用户自定义系数 r，查询结果个数 k，不确定时态数据集 O，RR-tree

```
输出：小顶堆 H
1. H←∅；Q←∅
2. Q. push(RR-tree. RootNode)；
3. WHILE(Q 不为空)
4.   NodeQ. front()；Q. pop()；
5.   AccessRelation(Node，RR-tree)；
6.   FOR EACH entry ∈ Node
7.     EMBR←entry. MBR()；
8.     IF(Q. x∈EMBR)
9.       IF(|H| = Q. k)
10.        min = f(Q，H. front())；
11.          IF(f(Q，EMBR) < min)
12.            Update(Q，H，Q. x，entry)；
13.          END IF
14.        ELSE
15.          Update(Q，H，Q. x，entry)；
16.      END IF
17.    END IF
18.  END FOR
19. END WHILE
20. RETURN H
```

算法 2 Update

```
输入：队列 Q，查询数据 Q. x，小顶堆 H，项 entry
输出：更新 entry 后的 H，Q
1. IF (Node 为内部结点)
2.   Q. push(entry. ptr)；
3. ELSE
4.   IF(Q. x∈entry. i , f(Q，entry. i) > min)
5.     H. push_back(entry. tid)；
6.     更新 H 中最小 score 值 min；
7.   END IF
8. END IF
```

由 Top-k 时态查询过程得到两个重要引理。

引理 1 给定查询 $Q.x$ 与节点项的 MBR，计算出的 $f(MBR)$比当前矩形框内部中的任意矩形框 MBR_in 的 $f(MBR_in)$都大。

证明：假设节点 N 和其内部项 E 都是非叶子节点，则 $p(Q,N)=p(Q,E)=1$，根据定义 4 可得，此时查询函数 f 的大小只与权值 w 有关。对于非叶子节点，权值取矩形框纵坐标最大值，E 作为 N 的子节点，其权值必定小于或等于 N 的权值，因此 $f(Q,N)\geqslant f(Q,E)$。假设 N 是非叶子节点，它的内部节点 E 是叶子节点，那么 $p(Q,N)=1$，$p(Q,E)\leqslant 1$，E 在 N 内部，因此 $N.w>E.w$，所以 $f(Q,N)>f(Q,E)$。

引理 2 给定节点项与查询数据 $Q.x$，以及小顶堆的堆顶值 min，若 $f(Q,MBR)\leqslant min$，则将项指向的节点从树中裁剪掉。

证明：因为小顶堆的堆顶值 min 是堆中最小的元素，若当前节点 N 与查询点 $Q.x$ 相交且 $f(Q,N)\leqslant min$，由引理 1 可得，节点 N 中的任一子节点与查询数据相交的查询函数值都将小于或等于 min，所以此节点可直接剪掉。

5.3 提炼过程

对于筛选后要进行精细计算的每条数据，在进行 Top-k 查询前，先计算不确定时态数据的相交概率以及查询函数值 $score$。将所得的 $score$ 值与小顶堆中堆顶元素进行比较筛选，最终小顶堆 H 中保存的数据即为查询结果。

6 实验

为验证本文提出的基于关系模式的 2D R-tree 处理 Top-k 查询算法的有效性，采用 C++语言在 Linux 环境下，向可扩展数据库 SECONDO[25] 中添加操作，以实现不确定时态数据的 Top-k 查询。实验环境为：Intel(R) Core(TM) I7-4770 CPU @3.40GHz，8GB 内存，Ubuntu14.04 64bit 操作系统。

6.1 实验数据

实验中使用真实数据集和合成数据集，其具体信息如表 1 所列。合成数据集在[1，100000]内随机选择间隔的起始点，在[1，1000]内随机选取长度，在[1，50000]内随机选取权值，产生不同规模的数据集，记为{S1，S2，S3，S4，S5，S6}。真实数据集 Flight 是某时间段中国航空公司航班的路线信息，这些数据包含约 3.2×10^6 条航班记录。在这个间隔数据中，起始点表示起飞时间段的起点，终止点表示起飞时间段终点加上飞行时间。间隔长度为飞行时间，权重为飞行距离。

表 1 数据集信息

Table 1 Information of data set

name	number	range	weight
S1	3×10^5	[1，100000]	[1，50000]
S2	9×10^5		
S3	1.5×10^6		
S4	5×10^6		
S5	8×10^6		
S6	1×10^7		
Flight	3.2×10^6	[1，2045]	[1，5000000]

6.2 对比方法

本实验对比了 4 种方法：1)无索引的基础方法；2)标准区间树；3)标准 R-tree；4)在标准 R-tree 的基础上，使用 B-tree 作为一个辅助结构来存储权值信息以提高查询效率。将这 4 种方法分别记为 Baseline，I-Tree，R-tree 和 BR-tree。本文提出的方法记为 AR-tree，其是在标准 R-tree 的基础上，利用数组作为一个辅助结构来存储权值信息。

6.3 查询性能实验

实验测试基于关系 R-tree 的索引的查询性能，包括数据量、返回结果 k 和自定义系数 r。实验对不同的参数进行测试，以比较不确定时态数据 Top-k 查询的平均 I/O 次数及 CPU 时间，实验参数设置如表 2 所列。

表 2 实验参数

Table 2 Experimental parameters

k	10	50	100	500	1000
r	0.1	0.3	0.5	0.7	0.9

6.3.1 数据量对算法性能影响

本节实验以不同合成数据集为实验数据，在相同查询数

据和返回结果下，比较基础方法和 4 种索引方法在不同数量级的数据集上的性能，结果如图 7 所示。可以看出，随数据量增长，相较于基础方法、区间树、R-tree 和 BR-tree，AR-tree 的 I/O 代价和 CPU 时间最小且性能较稳定。

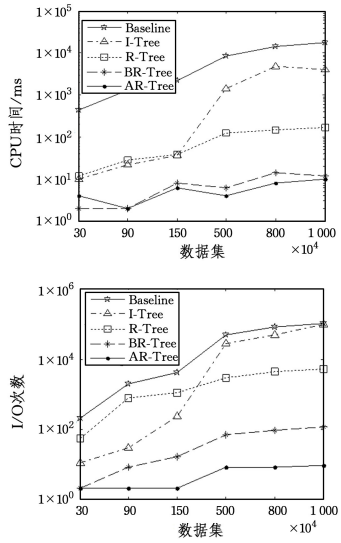

图 7 数据集对查询性能影响

Fig. 7 Effect of data sets on query performance

6.3.2 k 对算法性能影响

本节实验以 Flight 数据为实验数据，通过设置不同 k 值来比较各方法的性能，实验结果如图 8 所示。BR-tree 和 AR-tree 都包含通过优先队列的阈值剪枝的方法，k 值会影响空间剪枝能力，因此当 k 值较小时，能尽早利用优先队列中的 min 对小于该值的节点进行剪枝，当 k 值增加时，空间剪枝能力降低，I/O 次数及 CPU 代价增加，但相比其他方法，其仍能以最低成本完成查询。

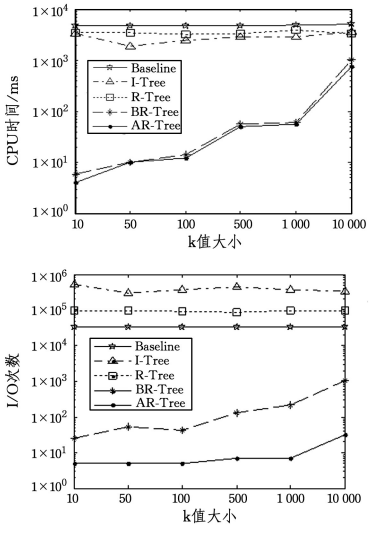

图 8 k 值对查询性能的影响

Fig. 8 Effect of k on query performance

表 3 是对查询数据 $Q(1125, k, 0.5)$ 的 k 个实验结果的 p 和 $score$ 的分析，k 是实验变量。所有数据中，与查询数据相交的有 1447811 条，相交概率 p 的范围为 $[0.01, 1]$，$score$ 的范围为 $[0.143, 1]$。本算法最终返回的 k 个结果是相交概率都为 1，最终的 $score$ 值都大于 0.999，且所有相交数据中相交概率和权值都较大的 k 条数据。

表 3 k 个结果的概率和权值

Table 3 p and $score$ of k results

k	10	50	100	500	1000
p	=1	=1	=1	=1	=1
$score$	>0.999	>0.999	>0.999	>0.999	>0.999

6.3.3 r 对算法性能的影响

本节实验以 S1 为实验数据，设置不同的系数 r，实验结果如图 9 所示。可见系数 r 对实验性能影响甚小，它影响的是查询结果。r 偏小时，查询结果偏向于权值大的数据；r 偏大时，查询结果偏向于相交概率大的数据，如表 4 所列。

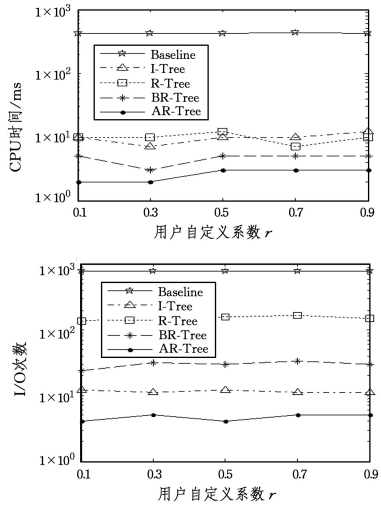

图 9 r 对查询性能的影响

Fig. 9 Effect of r on query performance

表 4 不同参数 r 下 p 和 w 的范围

Table 4 Scope of p and w under different value of r

r	0.1	0.3	0.5	0.7	0.9
p	[0.11,1]	[0.76,1]	1	1	[0.98,1]
w	[45623, 49946]	[43669, 48950]	[43075, 48253]	[43125, 48523]	[43075, 47339]

结束语 本文设计了一种基于关系模式的 2D R-tree 的不确定时态数据 Top-k 查询算法，在 2D R-tree 的基础上添加一个附加结构来管理 R-tree 中每个节点的 NodeId 和其对应的条目的索引号，使得查询过程可以按权值从大到小索引数据以提高查询效率。文中对空间和时间成本进行了全面的理论分析，通过实验证明了所提算法的高效性和可扩展性。本文算法考虑的是不确定时态数据的 Top-k 查询，在接下来的工作中可以考虑不确定时态数据的连续 Top-k 查询。

参 考 文 献

[1] LI R, ZHANG X, ZHOU X, et al. INK: A Cloud-Based System for Efficient Top-k Interval Keyword Search[C]// Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management. New York, 2014: 2003-2005.

[2] SNODGRASS R T, AHN I. A Taxonomy of Time in Databases.

[J]. ACM,1985,14(4):236-246.
[3] UYSAL M. Efficiently Processing Queries on Interval-and-Value Tuples in Relational Databases[C]// International Conference on Very Large Data Bases. DBLP,2005:385-396.
[4] ARGE L,VITTER J S. Optimal external memory interval management [J]. SIAMJ. 2003,32(6):1488-1508.
[5] HAMPEL M. Joining Interval Data in Relational Databases [C]// ACM SIGMOD International Conference on Management of Data. DBLP,2004:683-694.
[6] LAYER R M,SKADRON K,ROBINS G,et al. Binary Interval Search:a scalable algorithm for counting interval intersections [J]. Bioinformatics,2013,29(1):1-7.
[7] AGARWAL P K,ARGE L,YI K. An optimal dynamic interval stabbing-max data structure[C]// Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms. Vancouver,SODA,2005:803-812.
[8] SFAKIANAKIS G,PATLAKAS I,NTARMOS N,et al. Interval indexing and querying on key-value cloud stores[C]// Proceedings of IEEE 29th International Conference on Data Engineering. Brisbane:IEEE Press,2013:805-816.
[9] XU J Q,LU H. Efficiently answer top-k queries on typed intervals[J]. Information Systems,2017,71:164-181.
[10] LI S,WANG H. Dispersing points on intervals [J]. Discrete Applied Mathematics,2018,239:106-118.
[11] COWLEY W,PLEXOUSAKIS D. An Interval Algebra for Indeterminate Time[C]// Seventeenth National Conference on Artificial Intelligence. AAAI,2000:470-475.
[12] PLESNIEWICZ G S,VU NGUEN T M,DMITRY M. Query answering over fact bases for fuzzy interval ontologies[C]// Proceedings of the 2nd International Workshop on Soft Computing Applications and Knowledge Discovery. Moscow,2016:93-100.
[13] WU W Y,LI Y,NI Z W,et al. Probabilistic interval-valued hesitant fuzzy information aggregation operators and their application to multi-attribute decision making[J]. Algorithms,2018,11(8):120-128.
[14] ZHANG S C. Interval-Gap-Based Representation of Temporal Knowledge[J]. Journal of Software,1994,5(6):49-54.
[15] LIN J Y,PENG H,XIE J M. Uncertain temporal information representation and the extensions of temporal operation [J]. Computer Science,2005,32(8):161-163.
[16] KATERINA P,MARTIN T,MICHAEL B. Supporting Set Operations in Temporal-Probabilistic Databases[C]// 34th IEEE International Conference on Data Engineering (ICDE). 2018:1180-1191.
[17] BLANKENAGEL G. External segment trees[J]. Algorithmica,1994,12(6):498-532.
[18] IMAI H,ASANO T. Dynamic orthogonal segment intersection search[J]. Journal of Algorithms,1987,8(1):1-18.
[19] MCCREIGHT E M. Priority search trees [J]. SIAM J,1985,14(2):257-276.
[20] GUY D T MOL R D,BRONSELAER A. Indexing Possibilistic Numerical Data:The Interval B^+-tree Approach[M]// Information Processing and Management of Uncertainty in Knowledge-Based Systems. Springer International Publishing,2016:305-316.
[21] MEI W,MENG X,PENG S,et al. A hybrid index for temporal big data[J]. China Modern Medicine,2017,72:264-272.
[22] KRIEGEL H P,PÖTKE,MARCO,et al. Managing Intervals Efficiently in Object-Relational Databases[C]// International Conference on Very Large Data Bases. Morgan Kaufmann Publishers Inc,2000:407-418.
[23] BLIUJUTE R,JENSEN C S,SALTENIS S,et al. R-tree Based Indexing of Now-Relative Bitemporal Data[C]// International Conference on Very Large Data Bases. Morgan Kaufmann Publishers Inc.,1998:345-356.
[24] ZENG X J. Temporal index trees for uncertain temporal information [J]. Microcomputer Information,2011(5):236-237.
[25] GÜTING R H,BEHR T,DÜNTGEN C. SECONDO:A platform for moving objects database rsearch and for publishing and integrating research implementations[J]. IEEE Data Eng,2010,33(2):56-63.

WEI Jian-hua, born in 1994, postgraduate. Her main research interests include temporal data and uncertainty.

XU Jian-qiu, born in 1982, Ph.D, professor, is a member of China Computer Federation. His main research interests include mobile objects, and spatio-temporal data.