个对象之间的空间相似度大于等于实际的相似度,则$\forall o\in O, Sim(o,q)\leqslant r$.假设 $o(x,y,z)$映射到三维空间后为$o'(x',y',t')$,计算对象 o 映射后的增量如下:

$$\begin{aligned}\Delta &= SimS(o',q')-Sim(o,q)\\ &=\sqrt{(o'-x'_q)^2+(y'-y'_q)^2+(t'-t'_q)^2}-(\alpha\sqrt{(x-x_q)^2+(y-y_q)^2}+(1-\alpha)|t-t_q|)\\ &=\sqrt{2\alpha^2(x-x_q)^2+2\alpha^2(y-y_q)^2+2(1-\alpha)^2(t-t_q)^2}-(\alpha\sqrt{(x-x_q)^2+(y-y_q)^2}+(1-\alpha)|t-t_q|)\\ &\leqslant(\sqrt{2}-1)(\alpha\sqrt{(x-x_q)^2+(y-y_q)^2}+(1-\alpha)|t-t_q|)\\ &=(\sqrt{2}-1)Sim(o,q)\\ &\leqslant(\sqrt{2}-1)r.\end{aligned}$$

由此,$SimS(o',q')=Sim(o,q)+\Delta\leqslant r+(\sqrt{2}-1)r=\sqrt{2}r$,所以我们得到:当 $R=\sqrt{2}\,r$ 时,以查询点 q 为球心、R 为半径的球内,一定能包含 k 个最优的结果对象.

第 17 行~第 25 行是继续遍历 ST-Rtree,对所有和上面确定的球有交集的节点都要遍历,即,保证遍历到所有有可能包含结果的节点并计算处理.最后,第 26 行~第 33 行是处理上面所有过程中得到的有可能是结果对象的数据对象点,根据公式(3)计算这些点代表的数据对象和查询对象之间的实际相似度,然后按照实际相似度得分排列这些数据对象,从而得到最后的 k 个最优结果. □

5 实验测试

本节针对第 3 节中提出的 ST-Rtree 索引和第 4 节中的查询算法进行实验测试,并设计对比实验,证明该混合索引和查询的高效性和准确性.

5.1 代价分析

对于时空数据的索引的研究由来已久[11,20,22],并取得了很多成果,比如 RT-tree[11],3DR-tree[12],MR-tree[11],HR-tree[13]等.下面通过表 1 来对比说明一下这些索引和本文提出的 ST-Rtree 索引的优缺点.

Table 1

表 1

	索引大小	构建代价	时间变量类型	适合的查询类型
RT-tree	$O\left(\frac{N}{B}\right)$	$O\left(\frac{N}{B}\log\frac{N}{B}\right)$	时间段	对某个时间段的范围查询效率较高,kNN 查询效率低
HR-tree	$O\left(\frac{N^2}{B}\right)$	$O\left(\frac{N^2}{B}\log\frac{N}{B}\right)$	时间点	对某个时间段内的时间点查询效率较高,kNN 查询效率很低
3DR-tree	$O\left(\frac{N}{B}\right)$	$O\left(\frac{N}{B}\log\frac{N}{B}\right)$	时间段	范围查询效率很高,kNN 查询效率低
ST-Rtree	$O\left(\frac{N}{B}\right)$	$O\left(\frac{N}{B}\log\frac{N}{B}\right)$	时间点	范围查询效率很高,kNN 查询效率很高

RT-tree 是先给数据对象的空间信息建立二维的 R-tree 空间索引,然后在每个节点上汇总时间信息,时间信息和空间信息是分开存储的,它更适合移动对象的范围查询.在查询时,先过滤时间信息,再查询空间信息.对于本文提出的对带有地理位置和时间点对象的 kNN 查询,效率不高.MR-tree 和 HR-tree 都是利用重叠的思想,采用两级索引,建立多个不同时间戳的 R-tree,通过时间戳确定对应的 R-tree,然后利用空间 R-tree 索引进行查找.对于像本文这样的精确的 k 近邻查询,需要两重过滤查找,效率很低;且需要建立多个 R-tree,空间代价很大.3DR-tree 是将时间因素作为二维空间之外的第三个维度,虽然给时间单独建立一维索引,但由于时间和空间的度量单位不同,所以不能简单地像三维空间 R-tree 那样通过计算空间距离来查找对象,查询时先确定包含或者相交查询条件的时间区间,再查找在这个时间区间内的空间位置信息,分开过滤时间和空间两个因素,更适合范围查询,而且时间变量大多是一个时间区间,对带有地理位置和时间点对象的精确 kNN 查找效率很低.TPR-tree

索引[16]虽然考虑了时间因素,但主要是用来根据位置函数确定移动对象的位置,本质上还是对空间位置的索引,查询时也只是对移动对象的空间位置查询,时间因素不作为查询条件,不适用于文本的 *k*NN 查询.文献[15]中对移动对象查询时,虽然将时间因素作为一个维度和空间因素共同建立多维 R-tree 索引,但在查询时,先利用查询的时间区间过滤,然后再处理对象的位置信息,相当于两次查找,其索引的效率和 RT-tree 索引相似.但是文章中提出的查询算法主要是针对范围查询和单一维度上的近邻查询,对于综合时间因素和空间因素的 *k*NN 查询,会出现结果不全面和不准确的情况.

所以,综合上面的介绍对比,本文提出的 ST-Rtree 索引在索引大小、构建代价上和已有索引相似,但是在范围查询和 *k*NN 查询时,查询效率都很高.

5.2 对比实验

上一节已经对各种已有索引的代价及其适合的查询类型进行了详细分析,本文采用 RT-tree 的索引思想作为对比,对于 RT-tree 索引,它是先根据数据对象的空间信息建立一个二维的 R-tree 空间索引,然后在每个 R-tree 节点上汇总以这个节点为根的子树的时间信息.对于时间信息的存储,如果在每个节点上汇总所有子节点的全部时间信息,在遍历 RT-tree 的时候,会产生时间信息过滤效率过低,尤其是层数越低的节点,存储的时间信息越多,过滤的时间代价越高.本文采用在每个节点上只存储时间信息的最大值和最小值的 RT-tree 索引,这样做过滤的时间效率提高了,但是可能导致一些无用的遍历,因为仅仅根据存储的时间范围过滤,不能保证以这个节点为根的子树中一定包含最优的结果,可能还要回溯,所以效率也没有本文提出的 ST-Rtree 索引的效率高.

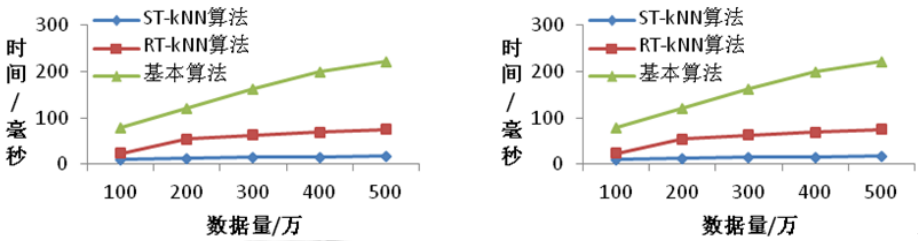
同时,本文还设计了基本的对比实验,即:不给数据建立任何索引,逐条查找并计算相似度,从而找到 *k* 个最近似的结果,将这个基本实验和采用 RT-tree 索引的实验作为对比实验,来和本文提出的查找算法进行对比.

5.3 实验设置

采用两个数据集,一个是真实微博数据集,抽取了 500 万条微博数据,其中,每条微博数据都带有地理位置坐标和时间标签;然后,将地理位置和时间标签进行处理,转化成(x,y,t)这样的三元组.另一个采用合成数据集,包含 1000 万个时空对象,随机抽取国内的地理位置坐标,然后随机分配时间标签,组成三元组.首先给这些三元组对象建立 ST-Rtree 索引,然后进行查询测试.实验时,我们设定调节权重的参数α为 0.6,查询个数 *k* 为 8.实验环境如下:主机采用 Intel(R) Core(TM) i7 CPU,3.07GHz 双核处理器,内存容量为 8G,操作系统为 64 位 Windows7.

5.4 实验结果和分析

图 4 显示了查询时间随着数据量改变的变化趋势,实验时,我们设定参数α为 0.6,*k* 值为 8.



(a) 微博数据集 (b) 合成数据集

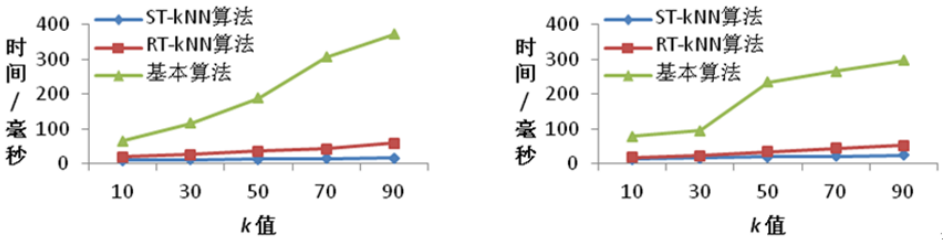
Fig.4 Query time effect of data sizes

图 4 数据量大小对查询时间的影响

我们可以看出:无论是微博数据集还是合成数据集,ST-*k*NN 查询算法、RT-*k*NN 查询算法和基本查询算法总体趋势都是随着数据量的增大,查询时间增加;并且 RT-*k*NN 查询算法和基本算法的查询时间明显大于 ST-*k*NN 算法的查询时间,说明本文提出的 ST-Rtree 索引和 ST-*k*NN 查询算法是很有效的.然而我们可以看到:对于 ST-*k*NN 查询,随着数据量的增大,它的查询时间增加的并不是很明显.这是因为在建立 ST-Rtree 索引的时候,ST-

Rtree 的高度随着数据量的大小呈现对数级增长,所以查询时间增长的慢.假设数据量大小为 N,ST-Rtree 的每个节点的最大容量为 t、最小为 $\left\lceil \frac{t}{2} \right\rceil$,那么 ST-Rtree 的最大高度为 $h=\log_{\left\lceil \frac{t}{2} \right\rceil}\frac{N}{2}+1$,则 ST-*k*NN 查询的最长时间代价为 $\left(\log_{\left\lceil \frac{t}{2} \right\rceil}\frac{N}{2}+1\right)k$.在实验中,我们设定 ST-Rtree 的每个节点的最大容量为 4,最小为 2.图 4(a)中,当数据量从 100 万增长到 500 万时,根据上面的计算我们发现,ST-Rtree 的高度从 12 增长到 14,变化不大,所以我们看到, ST-*k*NN 查询时间增长比较慢.

图 5 显示的是查询时间随着 k 值改变的变化趋势,测试的数据量为 100 万个,参数α设为 0.6,k 值从 10 变化到 90.从图 5(a)、图 5(b)中查询时间的变化趋势可以看出:随着 k 值的增大,无论是微博数据集还是合成数据集的查询时间都增加得很明显,ST-*k*NN 查询算法的查询效率优于其他算法.

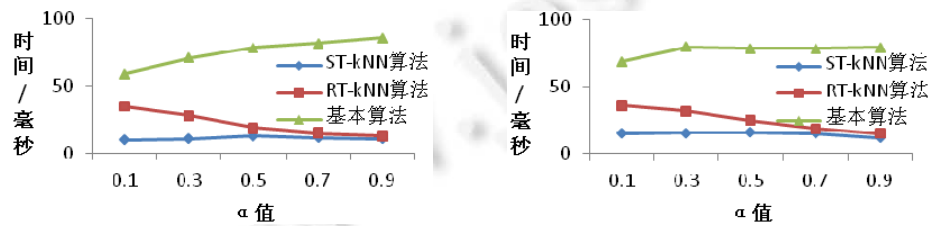


(a) 微博数据集 (b) 合成数据集

Fig.5 Query time effect of k

图 5 k 值对查询时间的影响

图 6 显示的是调节权重的参数α对查询时间的影响.这里,测试的数据量也为 100 万个,查询个数 k 设为 8.在两个数据集的测试结果中我们都可以看到:ST-*k*NN 查询算法和基本查询算法,参数α对它们的查询时间几乎是没有影响的.但是对于 RT-*k*NN 查询算法,从图 6 中可以看出它的查询时间是随着参数α的增大而减小,参数α增大,表示空间因素的权重增加了,说明用空间因素作为相似性查询的主要因素使查询效率提高了;反之,正如第 5.2 节中分析的,因为 RT-tree 索引中的时间信息在每个节点上的存储是用一个时间范围表示的,不够精确,所以查询效率降低了.参数α只是一个用户可以根据自己的偏好随意设定的值,应该尽量保证对查询效率没有影响,因此,RT-*k*NN 查询算法在这方面的性能不好.



(a) 微博数据集 (b) 合成数据集

Fig.6 Query time effect of α

图 6 α值对查询时间的影响

图 7 显示的是用微博数据集构建 ST-Rtree 索引和 RT-tree 索引时需要的时间随着数据量改变的变化趋势.实验时设定的数据量为 100 万~500 万,参数α为 0.6,k 值为 8.随着数据量的增大,构建索引的时间也在增加.同样的,对于构建 ST-Rtree 和 RT-tree 需要的空间进行了测试,从图 8 中可以看到:随着数据量的增大,构建索引需要的空间也在增大.对于 ST-tree 索引和 RT-tree 索引,可以看出,构建它们需要的时间和空间基本是相同的,但是从

上面几个图显示的实验结果来看,ST-tree 索引的效率比 RT-tree 高.

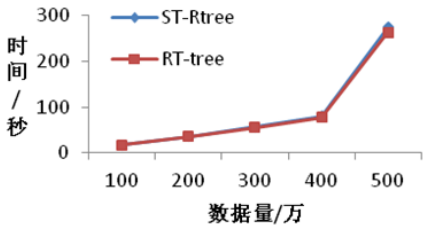


Fig.7 ST-Rtree's contruction time effect of data sizes

图 7 数据量对构建 ST-Rtree 时间的影响

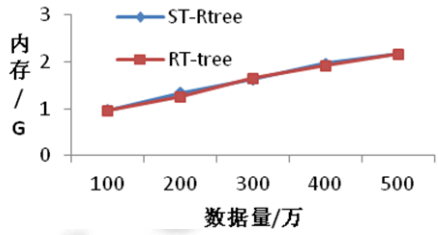


Fig.8 ST-Rtree's contruction memory cost of data sizes

图 8 数据量对构建 ST-Rtree 所需内存的影响

6 结束语

本文通过对带有地理位置标签和时间标签的数据对象进行映射变换,然后给映射变换后的数据对象建立 ST-Rtree 索引,并设计了在 ST-Rtree 索引上的精确查找算法,从而找到了 top-*k* 个离用户给出的查询点在时间和空间上都很近的对象,解决了用户关于带有时空标签信息的查询需求.本文提出的查询处理方法主要是对信息对象的时间和空间变量进行过滤查询,该方法很好地融合了时间和空间两个因素,提出了新的时空数据的混合索引,提高了查询效率.同时,本文提出的索引和查询算法能够帮助用户从网上各种带有时间标签和地理位置标签的信息中快速且准确地查询到想要的信息,为时空文本混合信息的查询研究做了铺垫.互联网上每天都产生大量的各种团购信息、微博信息等等,这些信息种类繁多,但是用户有时候只关心与其联系紧密的某类信息.为了防止用户想要查询的相关结果被大量不相关的信息淹没,下一步,将在本文的研究基础上加入文本信息的过滤功能,并通过研究设计融合时间、空间、文本这 3 个因素的索引,实现对这 3 个因素的过滤查询,以便更符合用户的查询需求.

References:

[1] Chen L, Cong G, Jensen CS, Wu D. Spatial keyword query processing: An experimental evaluation. In: Proc. of the 39th Int'l Conf. on Very Large Data Bases (VLDB 2013). VLDB Endowment, 2013. 217–228. [doi: 10.14778/2535569.2448955]

[2] Cao X, Chen L, Cong G, Jensen CS, Qu Q, Skovsgaard A, Wu D, Yiu ML. Spatial keyword querying. In: Atzeni P, Cheung D, Sudha R, eds. Proc. of the ER 2012. LNCS 7532, Springer-Verlag, 2012. 16–29. [doi: 10.1007/978-3-642-34002-4_2]

[3] Chen YY, Suel T, Markowetz A. Efficient query processing in geographic Web search engines. In: Proc. of the Int'l Conf. on Management of Data (SIGMOD 2006). New York: ACM Press, 2006. 277–288. [doi: 10.1145/1142473.1142505]

[4] Guttman A. R-Trees: A dynamic index structure for spatial searching. In: Proc. of the Int'l Conf. on Management of Data (SIGMOD'84). ACM Press, 1984. 47–57. [doi: 10.1145/602259.602266]

[5] Chen C, Li F, Ooi BC, Wu Sai. TI: An efficient indexing mechanism for real-time search on Tweets. In: Proc. of the Int'l Conf. on Management of Data (SIGMOD 2011). New York: ACM Press, 2011. 649–660. [doi: 10.1145/1989323.1989391]

[6] Cao X, Cong G, Jensen CS. Retrieving top-*k* prestige-based relevant spatial web objects. In: Proc. of the 36th Int'l Conf. on Very Large Data Bases (VLDB 2010). 2010. 3(1-2): 373–384. [doi: 10.14778/1920841.1920891]

[7] Zhang D, Chee YM, Mondal A, Tung AKH, Kitsuregawa M. Keyword search in spatial databases: Towards searching by document. In: Proc. of the 25th Int'l Conf. on Data Engineering (ICDE 2009). Shanghai: IEEE Computer Society, 2009. 688–699. [doi: 10.1109/ICDE.2009.77]

[8] Magdy A, Mokbel MF, Elnikety S, Nath S, He Y. Mercury: A memory-constrained spatio-temporal real-time search on microblogs. In: Proc. of the 30th Int'l Conf. on Data Engineering (ICDE 2014). Chicago: IEEE Computer Society, 2014. 172–183. [doi: 10.1109/ICDE.2014.6816649]

[9] Skovsgaard A, Sidlauskas D, Jensen CS. Scalable top-*k* spatio-temporal term querying. In: Proc. of the 30th Int'l Conf. on Data Engineering (ICDE 2014). Chicago: IEEE Computer Society, 2014. 148–159. [doi: 10.1109/ICDE.2014.6816647]

[10] Chen L, Cong G, Cao X, Tan KL. Temporal spatial-keyword top-*k* publish/subscribe. In: Proc. of the 31st Int'l Conf. on Data Engineering (ICDE 2015). Seoul: IEEE Computer Society, 2015. 255–266. [doi: 10.1109/ICDE.2015.7113289]

[11] Theodoridis Y, Sellis T, Papadopoulos AN, Manolopoulos Y. Specifications for efficient indexing in spatiotemporal databases. In: Proc. of the 10th Int'l Conf. on Scientific and Statistial Database Management. Capri: IEEE Computer Society, 1998. 123−132. [doi: 10.1109/SSDM.1998.688117]

[12] Theodoridis Y, Vazirgiannis M, Sellis T. Spatio-Temporal indexing for large multimedia applications. In: Proc. of the 3rd IEEE Int'l Conf. on Multimedia Computing and Systems. Hiroshima: IEEE Computer Society, 1996. 441−448. [doi: 10.1109/MMCS.1996.535011]

[13] Nascimento MA, Silva JRO. Towards historical R-trees. In: Proc. of the ACM Symp. on Applied Computing (SAC'98). New York: ACM Press, 1998. 235−240. [doi: 10.1145/330560.330692]

[14] Song Z, Roussopoulos N. *K*-Nearest neighbor search for moving query point. In: Proc. of the 7th Int'l Symp. on Advances in Spatial and Temporal Databases (SSTD 2001). London: Springer-Verlag, 2001. 79−96. [doi: 10.1007/3-540-47724-1_5]

[15] Lazaridis I, Porkaew K, Mehrotra S. Dynamic queries over mobile objects. In: Proc. of the 8th Int'l Conf. on Extending Database Technology (EDBT 2002). Springer-Verlag, 2002. 269−286. [doi: 10.1007/3-540-45876-X_18]

[16] Benetis R, Jensen CS, Karciauskas G, Saltenis S. Nearest neighbor and reverse nearest neighbor queries for moving objects. In: Proc. of the Int'l on Very Large Data Bases. New York: Springer-Verlag, 2002. 229−249. [doi: 10.1007/s00778-005-0166-4]

[17] Cong G, Jensen CS, Wu D. Efficient retrieval of the top-*k* most relevant spatial Web objects. In: Proc. of the Int'l Conf. on Very Large Data Bases (VLDB 2009). VLDB Endowment, ACM Press, 2009. 337−348. [doi: 10.14778/1687627.1687666]

[18] Felipe ID, Hristidis V, Rishe N. Keyword search on spatial databases. In: Proc. of the 24th Int'l Conf. on Data Engineering (ICDE2008). Cancun: IEEE Computer Society, 2008. 656−665. [doi: 10.1109/ICDE.2008.4497474]

[19] Lu J, Lu Y, Cong G. Reverse spatial and textual *k* nearest neighbor search. In: Proc. of the 2011 ACM SIGMOD Int'l Conf. on Management of Data. New York: ACM Press, 2011. 349−360. [doi: 10.1145/1989323.1989361]

[20] Mokbel MF, Ghanem TM, Aref WG. Spatio-Temporal access methods. IEEE Computer Society Technical Committee on Data Engineering, 2003,26(2):40−49.

[21] Zhou AY, Yang B, Jin CQ, Ma Q. Location-Based services: Architecture and progress. Chinese Journal of Computers, 2011,34(7):1155−1171 (in Chinese with English abstract).

[22] Zhu SP, Zhao JJ. Research of spatio-temporal access method. Computer Technology and Development, 2008,18(7):56−59 (in Chinese with English abstract).

附中文参考文献:

[21] 周傲英,杨彬,金澈清,马强.基于位置的服务:架构与进展.计算机学报,2011,34(7):1155−1171.

[22] 祝蜀平,赵瑾瑾.时空数据库索引方法研究.计算机技术与发展,2008,18(7):56−59.

李晨(1991－),女,辽宁鞍山人,硕士生,主要研究领域为查询处理.

寇月(1980－),女,博士,副教授,CCF 会员,主要研究领域为实体搜索,数据挖掘.

申德荣(1964－),女,博士,教授,博士生导师,CCF 高级会员,主要研究领域为分布式数据管理,数据集成.

聂铁铮(1980－),男,博士,副教授,CCF 会员,主要研究领域为数据质量,数据集成.

朱命冬(1985－),男,博士,讲师,CCF 会员,主要研究领域为相似性查询,分布式处理,数据分析.

于戈(1962－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为数据库,大数据管理.