WUJNS
Wuhan University Journal of Natural Sciences

Vol. 11 No. 3 2006 543-546

Article ID: 1007-1202(2006)03-0543-04

Processing Constrained *K* Closest Pairs Query in Spatial Databases

☐ LIU Xiaofeng, LIU Yunsheng†, XIAO Yingyuan

College of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, Hubei, China

Abstract: In this paper, constrained *K* closest pairs query is introduced, which retrieves the *K* closest pairs satisfying the given spatial constraint from two datasets. For data sets indexed by R-trees in spatial databases, three algorithms are presented for answering this kind of query. Among of them, two-phase Range+Join and Join+Range algorithms adopt the strategy that changes the execution order of range and closest pairs queries, and constrained heap-based algorithm utilizes extended distance functions to prune search space and minimize the pruning distance. Experimental results show that constrained heap-base algorithm has better applicability and performance than two-phase algorithms.

Key words: spatial databases; query processing; R-tree; closest pairs query; constrained closest pairs query

CLC number: TP 311.131

Received date: 2005-10-21

Foundation item: Supported by National Natural Science Foundation of China (60073045)

Biography: LIU Xiaofeng (1974-), male, Ph. D. candidate, research direction: spatiotemporal database, spatial database, real-time database, etc. E-mail: kiddenliu@sina.com

† To whom correspondence should be addressed. E-mail: ysliu@hust.edu.cn

0 Introduction

K closest pairs query (K-CPQ) finds the *K* closest pairs between two data sets, which is used frequently in various applications. In spatial databases, it is very important for evaluating K-CPQ efficiently. During the past several years, a few algorithms and techniques have been devised for it[1-5]. However, all of these techniques for K-CPQ search results in entire data space. Sometimes users might want to know the closest pairs in some area. For instance, someone may be interesting in the closest marketplace and residential area in some zone of a city, or the closest resort and city within some province. This kind of K-CPQ with spatial constraint is called Constrained *K* Closest Pairs Query (CCPQ), which discovers *K* closest pairs within given spatial range. The spatial ranges of above examples are some zone of city and some province.

There exists some work related to CCPQ. Jing Shan[6] introduced the notion of self range closest pair query (SRCP) and devised a SRCP tree for evaluating this kind of query. However, SRCP finds the closest pairs in one data set and the maintenances of SRCP tree need time and space. Ferhatosmanoglu[7] addressed constrained nearest neighbor query, whereas their techniques cannot be applied to CCPQ.

In this paper, we will address CCPQ in the context of spatial databases and Euclidean space, assuming that the two spatial data sets are stored in structures belonging in the family of R-trees[8, 9], due to their popularity.

1 Constrained Closest Pair Query

Definition 1 Let *P* and *Q* be two finite data sets stored in

a spatial database, C be the constrained spatial range. Then, the result of constrained K closest pairs query CCPQ(P,Q,C,K) is a set of ordered sequences of K ($1\leqslant K\leqslant|P|\cdot|Q|$) different pairs of objects of $P\times Q$:

① CCPQ(P,Q,C,K)=$\{(p_1, q_1), (p_2, q_2),\cdots, (p_K, q_K)\}\subseteq P\times Q$;

② $(p_i, q_i)\neq(p_j, q_j)$, $i\neq j$, $1\leqslant i, j\leqslant K$;

③ $\forall(p_i, q_i)\in$CCPQ(P,Q,C,K), contains(C,p_i) $\wedge$ contains(C,q_i), $1\leqslant i\leqslant K$;

④ $\forall p\in P\wedge$contains(C,p), $\forall q\in Q\wedge$contains(C,q), $(p,q)\notin$CCPQ(P,Q,C,K): dist(p,q)$\geqslant$dist(p_K,q_K)$\geqslant$dist(p_{K-1},q_{K-1})$\geqslant\cdots\geqslant$dist(p_1,q_1).

Where dist is Euclidean distance function between two objects, contains is a predicate checking if a spatial range contains an object.

Without spatial constraint, let M_P and M_Q be two Minimum Bound Rectangle (MBRs), then, the minimum distance between M_P and M_Q is defined as MINMINDIST, and MINMAXDIST expresses an upper bound of distance for at least one pair of objects[1]. Due to the given spatial constraint, these distance functions need to be extended for CCPQ.

Definition 2 Let M_P and M_Q be two MBRs, C be the constrained spatial range. Then, the minimum distance between M_P and M_Q with constraint C is defined as

$$\text{C-MINMINDIST}(M_P,M_Q,C)=\begin{cases}\text{Mindist}(M_P\cap C, M_Q\cap C),\\ \quad\text{if } M_P\cap C\neq\varnothing\wedge M_Q\cap C\neq\varnothing;\\ \infty, \text{ otherwise.}\end{cases}$$

Where Mindist returns the minimum distance between two spatial ranges, $M_P\cap C$ and $M_Q\cap C$ are spatial intersection of corresponding MBRs and C.

Theorem 1 Let M_P and M_Q be two MBRs, C be the constrained spatial range. M_P and M_Q enclose two sets of MBRs $\{M_{P_1}, M_{P_2},\cdots, M_{P_m}\}$ and $\{M_{Q_1}, M_{Q_2},\cdots, M_{Q_m}\}$ respectively. Then

① C-MINMINDIST(M_{P_i}, M_{Q_j}, C) $\geqslant$ C-MINMINDIST(M_P,M_Q, C), $1\leqslant i\leqslant m, 1\leqslant j\leqslant n$;

② C-MINMINDIST(M_P, M_{Q_j}, C) $\geqslant$ C-MINMINDIST(M_P,M_Q, C), $1\leqslant j\leqslant n$;

③ C-MINMINDIST(M_{P_i}, M_Q, C) $\geqslant$ C-MINMINDIST(M_P,M_Q, C), $1\leqslant i\leqslant m$.

Theorem 1 guarantees that if C-MINMINDIST of two MBRs is greater than some value T, the C-MINMINDISTs of their child MBRs are certainly greater than T.

Theorem 2 Let M_P and M_Q be two MBRs, C be the constrained spatial range. The nonempty data sets contained in $M_P\cap C$ and $M_Q\cap C$ are O_1 and O_2, then $\forall(o_1,o_2)\in O_1\times O_2$, C-MINMINDIST($M_P, M_Q, C$)$\leqslant$ dist(o_1,o_2).

Definition 3 Let M_P and M_Q be two MBRs, C be the constrained spatial range. Then, C-MINMAXDIST (M_P,M_Q,C) is defined as the distance which there exists at least one pair of objects (contained in M_P and M_Q) with distance smaller than or equal to. Let F_P and F_Q be the sets of faces of M_P and M_Q, which are fully contained in C respectively. Then:

① If $F_P\neq\varnothing\wedge F_Q\neq\varnothing$, C-MINMAXDIST($M_P$, M_Q,C)= min{Maxdist(f_i, f_j): $f_i\in F_P, f_j\in F_Q$};

② Otherwise, C-MINMAXDIST(M_P,M_Q,C)=∞.

Where Maxdist returns the maximum distance between two spatial ranges.

Theorem 3 Let M_P and M_Q be two MBRs, C be the constrained spatial range. The nonempty data sets contained in $M_P\cap C$ and $M_Q\cap C$ are O_1 and O_2, then $\exists(o_1,o_2)\in O_1\times O_2$, dist($o_1,o_2$) $\leqslant$C-MINMAXDIST (M_P,M_Q,C).

2 Algorithms for CCPQ

2.1 Naive Algorithms

Constrained K Closest Pairs Query naturally involves both range and closest pairs queries. A simple and straightforward approach for CCPQ is to execute these two queries sequentially. In terms of execution orders of range and closest pairs queries, two na ve algorithms can be devised.

The first method computes the general non-constrained closest pairs by an incremental distance join algorithm and checks if the objects of pairs are within C while outputting the closest pairs. This Join+Range method is referred to as JR algorithm. However, when the distances of the closest pairs in C are larger, JR might search more invalid data space before finding the satisfactory closest pairs. To avoid this, we might need to know the possible maximum distance between objects of the constrained closet pairs.

Theorem 4 Let Maxdist(C) be the maximum distance between two points in constraint C of CCPQ(P,Q, C,K). Then, $\forall(p,q)\in$CCPQ(P,Q,C,K): dist(p,q) $\leqslant$Maxdist(C).

Based on Theorem 4, while outputting the closet pairs (p,q) incrementally, JR can stop when dist(p,q)

$>$Maxdist(C). The following is JR algorithm.

Step 1 Get the next closest pair (p,q) between P and Q through incremental distance join. If dist$(p,q) >$ Maxdist(C), then stop. If p and q are within C, then output (p,q).

Step 2 If the number of outputted pairs is equal to K, then stop, else repeat algorithm from Step 1.

The second method first performs a range query retrieving the objects in C from P and Q, and then tests for the closest pairs. This Range+Join method is called RJ algorithm. The following is RJ algorithm.

Step 1 From P and Q, retrieve all objects falling within C to sets P_C and Q_C respectively.

Step 2 Join data sets P_C and Q_C to produce the K closest pairs.

2.2 Constrained Heap-Based Algorithm

Let T be the distance of the K-th closest pairs found so far, according to Theorem 2, the following Pruning Heuristic can be deduced.

Pruning Heuristic Let M_P and M_Q be two MBRs, C be the constrained spatial range. If C-MINMINDIST $(M_P,M_Q,C)>T$, the paths corresponding to (M_P,M_Q) will be pruned.

When $K=1$, we can get the following Updating Heuristic based on Theorem 3.

Updating Heuristic Let M_P and M_Q be two MBRs enclosing two sets of MBRs $\{M_{P_1}, M_{P_2},\cdots,M_{P_m}\}$ and $\{M_{Q_1},M_{Q_2},\cdots,M_{Q_n}\}$ respectively, C be the constrained spatial range, and $T'=\min\{\text{C-MINMINDIST}(M_{P_i}, M_{Q_j}) : 1\leqslant i\leqslant m, 1\leqslant j\leqslant n\}$. When $K=1$, if $T>T'$, then T can be updated to T'.

The above heuristics can be used to prune search space and minimize the pruning distance T.

Constrained heap-based algorithm (CH) utilizes a minimum heap M_H to hold pairs of MBRs according to their C-MINMINDIST (the pair with smallest C-MINMINDIST resides on top of M_H), and a maximum heap K_H with capacity K to record the K constrained closest pairs found so far (the pair with the largest distance resides on top of K_H). CH algorithm for two R-trees with the same height is as follows.

Step 1 Set T to ∞ and initialize heaps M_H and K_H. The pair formed by roots of two R-trees is inserted into M_H.

Step 2 If M_H is empty, then stop, else let (N_P, N_Q) be the pair de-heaped from M_H. If this pair has C-MINMINDIST$>T$, then stop.

Step 3 If N_P and N_Q are internal nodes, calculate C-MINMINDIST for all possible pairs of MBRs. If $K=1$, update T using Updating Heuristic. Insert into M_H those pairs that have C-MINMINDIST$\leqslant T$. If N_P and N_Q are two leaves, calculate the distance of each possible pairs of objects. If this distance is smaller than T, update K_H and T. Repeat algorithm from Step 2.

3 Experimental Results

This section provides the results of an extensive experimentation study aiming at comparing the applicability of three algorithms and evaluating the performance of CH algorithm.

The experiments were performed using two synthetic uniform data sets including 6000 points each, which were indexed by two R*-trees. The page size was set to 4 kB and no buffer was used. The constrained spatial range was a rectangle whose position was chosen randomly. The programs were created using the Microsoft Visual C++ compiler and all experiments were run a Windows PC with 512 MB RAM.

Fig. 1(a) shows the number of page accesses of three algorithms for $K=10$. The x-axis shows the relative size of constrained spatial range (i.e., 0.2 indicates the constrained spatial range fills 20% of the entire data space). As the constraint size increases, the number of page accesses of JR decreases, however, that of RJ increases. The reason is that with constraint size increasing, the possibility that the intermediate closest pairs produced by incremental distance join are final results increases, while RJ would produce more data due to range query. Fig. 1(a) also indicates that the number of page accesses of CH changes slightly as the constraint size increases. So, CH has better applicability. This reason is that JR and RJ algorithms produce a lot of intermediate results, while CH adopts a best first search strategy and no intermediate results are produced.

For $K=20$, Fig. 1(b) illustrates the similar results.

Table 1 shows the number of page accesses and response time of CH algorithm for constraint size 0.2. Both of them increase in a sub-linear way with the increase of the K. With increasing K values, the performance of CH is not significantly affected. Therefore, CH has better scalability.

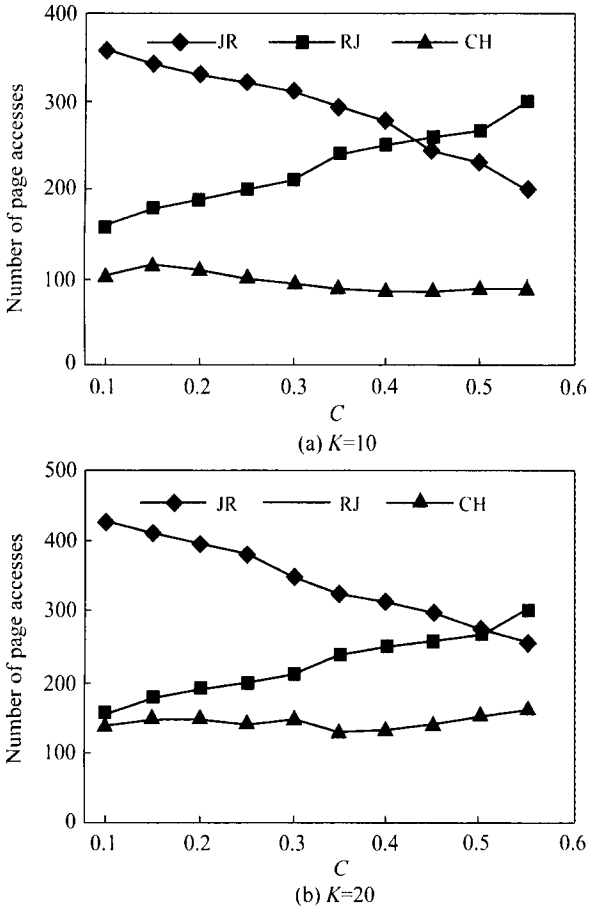

Fig. 1 Comparison of the CCPQ algorithms in terms of the number of page accesses for $K=10$ and $K=20$

Table 1 Performance of CH for $C=0.2$

K	Number of page accesses	Response time/ms
10	99	903
20	102	1039
30	109	1090
40	113	1112
50	118	1108
60	121	1115
70	125	1198
80	130	1203

4 Conclusion

In this paper, we introduced constrained K closest pairs query (CCPQ). For data sets indexed by R-trees in spatial databases, three algorithms were proposed to answer it. In terms of the execution order of range and closest pairs queries, RJ and JR were developed. After defining some distance functions, we presented constrained heap-based algorithm that used these functions to prune search space and minimize the pruning distance. Experiments on synthetic data sets show that constrained heap-based algorithm outperforms RJ and JR. The future work includes developing a cost model to estimate the cost of evaluation of CCPQ[10,11].

References

[1] Corral A, Manolopoulos Y, Theodoridis Y, *et al*. Closest Pair Queries in Spatial Databases [C] //*Proceedings of ACM SIGMOD Conference*. New York: ACM Press, 2000:189-200.

[2] Yang C, Lin K I. An Index Structure for Improving Nearest Closest Pairs and Related Join Queries in Spatial Databases [C] //*Proceedings of International Database Engineering and Applications Symposium*. Washington: IEEE Computer Society, 2002:140-149.

[3] Hjaltason G R, Samet H. Incremental Distance Join Algorithms for Spatial Databases [C] //*Proceedings of ACM SIGMOD Conference*. New York: ACM Press, 1998:237-248.

[4] Shin H, Moon B, Lee S. Adaptive Multi-Stage Distance Join Processing [C] //*Proceedings of ACM SIGMOD Conference*. New York: ACM Press, 2000:343-354.

[5] Corral A, Manolopoulos Y, Theodoridis Y, *et al*. Algorithms for Processing K-Closest-Pair Queries in Spatial Databases [J]. *Data & Knowledge Engineering*, 2004, **49**(1): 67-104.

[6] Jing Shan , Zhang Donghui, Salzberg Betty. On Spatial-Range Closest-Pair Query [C] //*Proceedings of 8th Symposium on Spatial and Temporal Databases*. New York: Springer, 2003:252-269.

[7] Ferhatosmanoglu H, Stanoi I, Agrawal D, *et al*. Constrained Nearest Neighbor Queries [C] //*7th International Symposium on Spatial and Temporal Databases*. London, UK: Springer-Verlag , 2001:257-278.

[8] Guttman. R-trees: A Dynamic Index Structure for Spatial Searching [C] //*Proceedings of the* 1984 *ACM SIGMOD International Conference on Management of Data*. New York: ACM Press, 1984:47-57.

[9] Beckmann N, Kriegel H P, Schneider R, *et al*. The R*-tree: an Efficient and Robust Access Method for Points and Rectangles [C] //*Proceedings of the* 1990 *ACM SIGMOD International Conference on Management of Data*. New York: ACM Press, 1990:322-331.

[10] Yannis T, Emmanuel S, Timos S. Cost Models for Join Queries in Spatial Databases [C] //*Proceedings of the Fourteenth International Conference on Data Engineering*. Washington: IEEE Computer Society, 1998:476-483.

[11] Yannis T, Emmanuel S, Timos S. Efficient Cost Models for Spatial Queries Using R-trees [J]. *IEEE Transactions on Knowledge and Data Engineering*, 2000, **12**(1):19-32.

□