An Efficient PAB-Based Query Indexing for Processing Continuous Queries on Moving Objects

Su Min Jang, Seok Il Song, and Jae Soo Yoo

ABSTRACT—Existing methods to process continuous range queries are not scalable. In particular, as the number of continuous range queries on a large number of moving objects becomes larger, their performance degrades significantly. We propose a novel query indexing method called the projected attribute bit (PAB)-based query index. We project a two-dimensional continuous range query on each axis to get two one-dimensional bit lists. Since the queries are transformed to bit lists and query evaluation is performed by bit operations, the storage cost of indexing and query evaluation time are reduced significantly. Through various experiments, we show that our method outperforms the containment-encoded squares-based indexing method, which is one of the most recently proposed methods.

Keywords—Query indexing, continuous range queries, moving objects, location-based services, mobile computing.

I. Introduction

The efficient processing of a large number of continuous range queries over moving objects is critically important in providing location-aware services and applications. Depending on whether or not queries move, the processing of continuous range queries on moving objects can be roughly classified into two categories [1]. The first category deals with stationary queries over moving objects [1]-[3], and the second category deals with moving queries on moving objects [4]. This paper focuses on the latter. The storage cost and query evaluation time of stationary queries on moving objects [1]-[3] increase rapidly with respect to the size of the monitoring region. In particular, the storage cost increases in proportion to the R^2, where R is the size of the monitoring region. Also, their overall performance is degraded remarkably when the number of continuous range queries increases whether or not continuous queries overlap. Since our method transforms the two-dimensional information of continuous range queries into one-dimensional bit lists by projection, the storage cost of our method increases linearly in proportion to $2R\times N$, where N is the number of queries. Also, our method allows query processing to take advantage of incremental updating and supports fast processing time through bit operations between two projected query set bit lists (PQSBLs).

Manuscript received Mar. 20, 2007; revised July 5, 2007.

This work was supported by the Korea Research Foundation Grant funded by the Korea Government (MOEHRD) (The Regional Research Universities Program/Chungbuk BIT Research-Oriented University Consortium.)

Su Min Jang (phone: + 82 43 261 3230, email: jsm@netdb.chungbuk.ac.kr) and Jae Soo Yoo (phone: +82 43 261 3230, email: yjs@chungbuk.ac.kr) are with the Department of Computer and Communication Engineering, Chungbuk National University, Cheongju, Rep. of Korea.

Seok Il Song (email: sisong@chungju.ac.kr) is with the Department of Computer Engineering, Chungju National University, Chungju, Rep. of Korea.

II. PAB-Based Query Indexing

In our method, two-dimensional data space is partitioned into $Rx\times Ry$ rectangular cells, where Rx and Ry denote the number of slices on each axis. Users specify Rx and Ry as optimal values for a certain application. The Rx and Ry slices along each axis are determined in such a way that all slices are equal. Then, a range query is transformed into two bit lists, or projected query bit lists (PQBLs), by projecting query rectangles to each axis. Since a bit is assigned to a slice of an axis, the size of each bit list is Rx or Ry. All bits are set as 0 in the beginning. After shadows representing the query are placed on an axis by projection, a bit whose corresponding slice overlaps the shadow is set to 1.

Figure 1 shows an example of transforming a query rectangle q into two PQBLs (PQBLx, PQBLy) on each axis. In Fig. 1(a), $Rx\times Ry$ denotes the monitoring region, which is the dotted area, where Rx and Ry is the range of each axis. The transformation is performed on each axis. Fig. 1(b) shows the transformation process on axis X. In the figure, when projecting

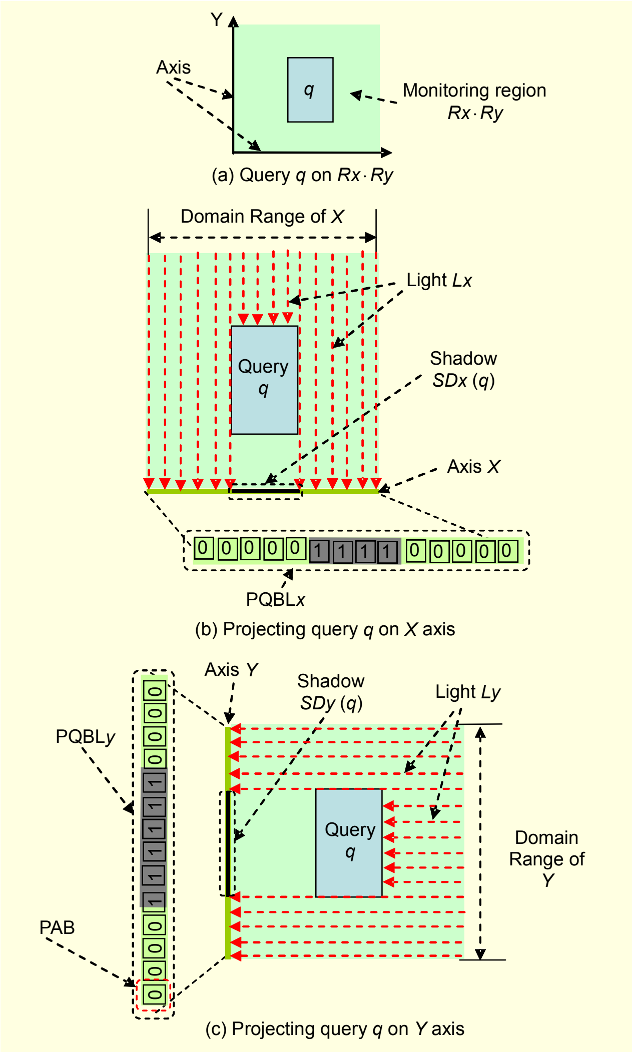

Fig. 1. Transforming range query into PQBL.

query q on axis X by the light Lx, the shadow $SDx(q)$ of q is laid on axis X. In the same way, q is projected on axis Y in Fig. 1(c). The PQBLx on axis X is $(00000111100000)_2$ as shown in Fig. 1(b) and the PQBLy on axis Y is $(00001111110000)_2$. Each bit of the PQBLs is a projected attribute bit (PAB).

To process continuous range queries, we should construct two PQSBLs, namely, PQSBLx for axis X and PQSBLy for axis Y. Figure 2 shows the process of constructing the PQSBLs. We maintain a query ID list (QL) to store query IDs, and each entry of the QL has an object list for object IDs contained in its query. The order of PQBLs in the PQSBLs is the same as that of query IDs in the QL; that is, the i-th PQBLx of PQSBLx corresponds to the i-th query ID of QL.

We define a bit list which consists of the i-th bits in PQSBLx (PQSBLy) as PQSBL$x_{row}(i)$ (PQSBL$y_{row}(i)$). For example, in Fig. 2, the PQSBL$y_{row}(5)$ is the bit list of the fifth bits in PQSBLy. We maintain the object list of a query incrementally. To do this, we find queries that no longer contain the object,

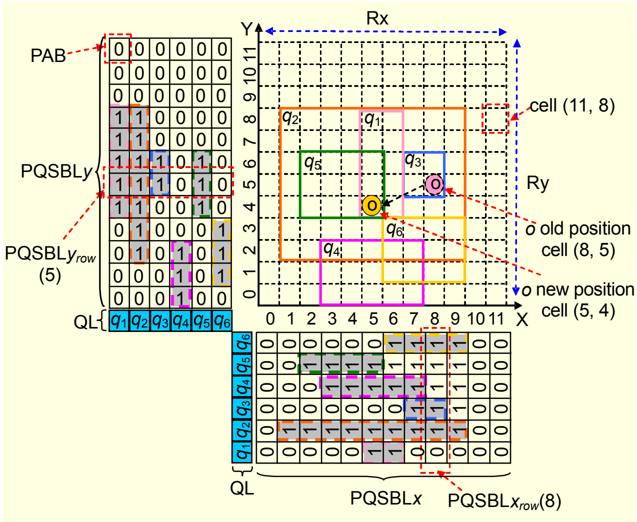

Fig. 2. Constructing PQSBLs for six queries.

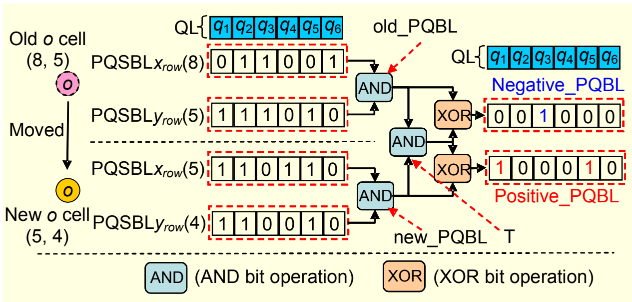

Fig. 3. Incremental query evaluation with bit operations.

which we denote the queries as NeQL (Negative QL). Queries that newly contain the object are denoted as PoQL (Positive QL).

Figure 3 shows an example of evaluating queries for a moving object o. The object o is moved from cell (8, 5) to cell (5, 4). Through an AND operation between PQSBL$x_{row}(i)$ and PQSBL$y_{row}(j)$ for an object on the cell (i, j) which is the old or new position, we can get the list of queries (old_PQBL or new_PQBL in Fig. 3) which includes the object. To find NeQL and PoQL, an AND operation between old_PQBL and new_PQBL is carried out. We denote the result of the AND operation as T. Then, to get NeQL(PoQL), we perform an XOR operation on T and old_PQBL (new_PQBL). Finally, we get Negative_PQBL $(001000)_2$ for NeQL and Positive_PQBL $(100010)_2$ for PoQL as shown in Fig. 3. With Negative_PQBL and Positive_PQBL, we adapt the object lists of the queries q_3, q_1, and q_5. That is, o is deleted from the object list of q_3 and o is added to the object lists of q_1 and q_5.

We use an approximation scheme. We expand non-integer values to the nearest integer values and get minimum bounding regions for non-rectangular query regions. Therefore, there needs to be one or more steps to find final results from candidates produced by the proposed algorithm.

To dynamically maintain the number of bitmaps for queries, the PQSBLs have *k* percent free space, which is determined by users according to the characteristics of application. If there is no free space, we reconstruct the PQSBLs in order to have *k* percent additional free space.

III. Performance Evaluation

Simulations were conducted to evaluate and compare PAB-based indexing with CES-based indexing. We measured the incremental query evaluation time and the total storage cost. We assumed that there were no changes to the query index between two query re-evaluations. The maximum side length of a CES for the CES-based query index was set to the best performance value of 16. The maximum width (height) of a query is denoted as *W*, and the size of a query was randomly and independently chosen between 1 and *W*. The new location of a moving object was calculated based on its old position and the horizontal and vertical movements. We denote the maximal movement of objects as *M*. Our experiments were performed on a 3.0 GHz Pentium IV with 1024 Mbytes of main memory running Windows XP.

Figure 3(a) shows the storage cost of both methods. The storage cost was measured with varying *R* from 512 to 8,192 and |*Q*|, which is the number of queries from 8,000 to 32,000 (|*O*| =50,000, *M*=2, *W*=50). Since the storage cost of our method is proportional to *2R* and that of CES-based index is proportional to R^2, the storage utilization of our method is superior to that of CES. Figure 3(b) shows the impact of |*Q*| on query re-evaluation time. Queries are uniformly distributed (*R*=1,024, |*O*| =50,000, *W*=50). Query evaluation time is measured with varying |*Q*| from 1,000 to 16,000 and *M* as 2 and 10. The query re-evaluation time of our proposed index is much better than that of CES with the same *M*. Figure 3(c) shows the impact of |*O*| on query reevaluation time. In this experiment, *R* and *W* were set to be the same as those shown in Fig. 3(b), and |*Q*| was set at 8,000. We measured the query re-evaluation time with varying |*O*| from 4,000 to 64,000. In this experiment, our method again outperforms CES. As the |*O*| increases, the performance gap between the two methods increases.

IV. Conclusion

Our method allows query processing to take advantage of incremental updating and supports fast processing time through bit operations between two PQSBLs while requiring little storage. Our simulation results show that PAB-based query indexing substantially outperforms CES-based query indexing in terms of incremental query evaluation time and storage cost. Also, our method is suitable for high dimensional data, since the number of PQSBLs to process continuous query on high dimensional data is equal to the number of dimensions. In our future work, we will expand our method to high dimensional data space.

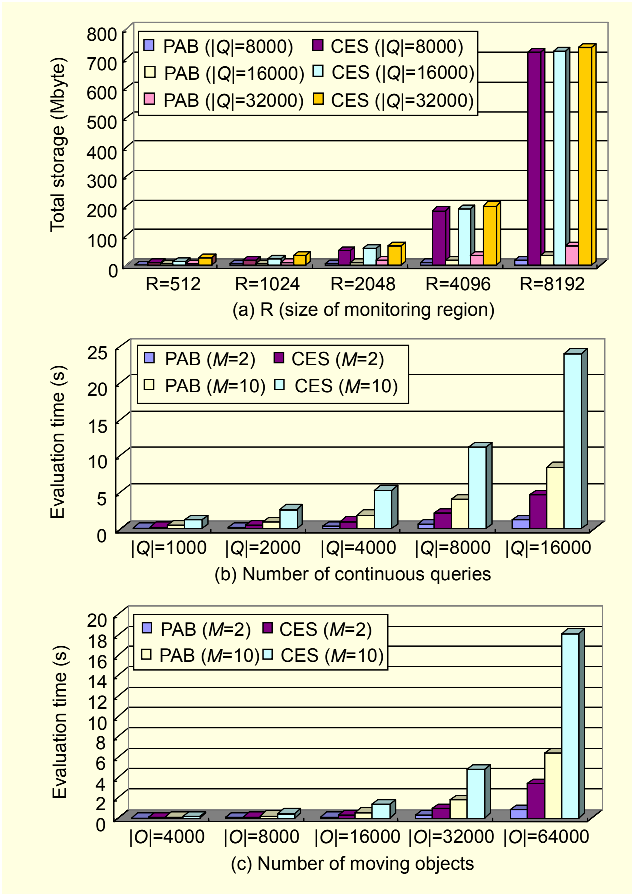

Fig. 4. (a) Impact of *R* on total index storage, (b) impact of |*Q*|, and (c) impact of |*O*| with various *M* on (re)evaluation time.

References

[1] K.L. Wu, S.K Chen, and P.S. Yu, "Incremental Processing of Continual Range Queries over Moving Objects," *IEEE Trans. Knowledge and Data Eng Journal*, vol. 18, Nov. 2006, pp. 1560-1575.

[2] K.L. Wu, S.K Chen, and P.S. Yu, "Processing Continual Range Queries over Moving Objects Using VCR-Based Query Indexes," *Proc. IEEE Int'l Conf. Mobile and Ubiquitous Systems: Networking and Services*, Aug. 2004.

[3] K.L. Wu, S.K Chen, and P.S. Yu, "Shingle-Based Query Indexing for Location-Based Mobile E-Commerce," *Proc. IEEE Int'l Conf. E-Commerce*, July 2004.

[4] M.F. Mokbel, X. Xiong, and W.G. Aref, "SINA: Scalable Incremental Processing of Continuous Queries in Spatio-Temporal Databases," *Proc. of ACM SIGMOD Int'l Conf. Management of Data,* 2004.