ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 3, March 2016

Scalable Distributed Processing of K nearest Neighbour Queries over Moving Objects

Prof.A.N Banubakode , Gajanan Payghan , Kokane Anirudh, Panchal Suraj, Kamble Rohit

Department of IT, JSPM's Rajarshi Shahu College of Engineering, Pune, Savitribai Phule Pune University, India

ABSTRACT: Many applications involving moving objects is the task of processing k-nearest neighbour (k-NN) queries. Most of the existing approaches to this problem are designed for the centralized setting where query processing takes place on a single server; it is difficult, if not impossible, for them to scale to a distributed setting to handle the vast volume of data and concurrent queries that are increasingly common in those applications. To address this problem, we propose a suite of solutions that can support scalable distributed processing of k-NN queries. We first present a new index structure called Dynamic Strip Index (DSI), which can better adapt to different data distributions than exiting grid indexes. Moreover, it can be naturally distributed across the cluster, therefore lending itself well to distribute processing. We further propose a distributed k-NN search (DKNN) algorithm based on DSI. The advances of GPS technology and wide-ranging usage of wireless communication devices have facilitated the collection of large amount of spatiotemporal data. Of special interest are data related to moving objects. By modelling and analysing moving objects data, we learn about the moving objects behaviour and even predict their future locations. Discovery of patterns and prediction of future movement can greatly influence different fields. Devising the correct inference is a scientific and computational challenge. Fundamentals of moving objects' representation and space partitioning are presented. Markov models are described in general along with their application to spatiotemporal prediction. Process of prediction and possible enhancements are shown.

KEYWORDS -Moving Objects, k nearest neighbourquery,Distributed query processing, Privacy-preserving, Moving objects.

I. INTRODUCTIONS

All moving things in the real world can comprise spatiotemporal data, containing time and space attributes simultaneously [2]. Data that are moving in time with changes of their location or shape describe a moving object. In many applications, knowing moving objects location in advance is very appreciable. Discovery of patterns and prediction of future movement can greatly influence different fields. Some of examples are analysing wild animals' movement in order to predict their migrations, monitoring vehicles and analysing their movement in order to predict traffic congestions, predicting the movement of a mobile user roaming around and changing access points to assure given level of quality of service in wireless networks or analyzing and predicting the movement of aircraft in combat in order to develop defending strategies. There are also situations in which determining the exact position of a moving object is not possible, for example when the moving object enters a shadow area of GPS, and estimating the location by referencing the previous ones which were provided in visible regions, becomes necessary. As technology advances, we encounter more available data on moving objects, thus increasing our ability to mine spatiotemporal data [3]. We can use these data to analyze moving objects behavior and to predict their future locations. k nearest neighbor (k-NN) queries over moving objects is a fundamental operation in many location-based applications.

II. MOTIVATION OF THE PROJECT

Distribution data naturally arise in countless domains, such as meteorology, biology, geology, industry and economics. However, relatively little attention has been paid to data mining for large distribution sets. Given n distributions of multiple categories and a query distribution Q, we want to find similar clouds (i.e., distributions), to discover patterns, rules and outlier clouds. For example, consider the numerical case of sales of items, where, for each item sold, we

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 3, March 2016

record the unit price and quantity; then, each customer is represented as a distribution of 2-d points(one for each item he/she bought). We want to find similar users, e.g., for market segmentation, anomaly/fraud detection. We propose to address this problem and present Search, which includes fast and effective algorithms for similarity search in large distribution datasets.

III. GOALS AND OBJECTIVES

An effective self-destruction technique is used data securely provided to the user and provides usability, performances. Many location-based applications require constant monitoring of k-nearest neighbor (k-NN) queries over moving objects within a geographic area. Existing approaches to this problem have focused on predictive queries, and relied on the assumption that the trajectories of the objects are fully predictable at query processing time. We relax this assumption, and propose two efficient and scalable algorithms using grid indices. One is based on indexing objects, and the other on queries. For each approach, a cost model is developed, and a detailed analysis along with the respective applicability is presented. The Object-Indexing approach is further extended to multi-levels to handle skewed data. We show by experiments that our grid-based algorithms significantly outperform R-tree-based solutions. Extensive experiments are also carried out to study the properties and evaluate the performance of the proposed approaches under variety of settings

1. Computing k-NN, in two ways, by using a dynamically updated grid-based index structure to index objects or queries.
2. Generalizing the object indices to hierarchical structures. This is shown to improve query performance and robustness for skewed object distributions.
3. Extending the query processing to incrementally maintain the query answers. This further improves the performance when the objects' movement is localized, i.e. the velocity is in general bounded

IV. RELEVANT MATHEMATICS ASSOCIATED WITH THE PROJECT

S={ O = {o1, . . . , on} , R 2, posi, Pi = hp (t0 i , . . . , p tk i , . . .), tj< tj+1 , p tj i = posi(tj), t , qi = (x, y, k)};
Where,
O= set of points
R 2= Euclidean space
On= position of object
posi= spatial positions
Pi= the time-ordered sequence of position updates issued by oi
ptj i =position update
t= generic time
qi= generic k-NN query
k =closest objects
(x, y)= center
We consider a set of points O = {o1, . . . , on} moving in a two-dimensional Euclidean space R 2 , where the position of object oi is given by the function posi : R≥0 → R 2 mapping time instants into spatial positions. These points model objects that issue position updates and queries as they move. Let Pi = hp t0 i , . . . , p tk i , . . .i, tj< tj+1, be the time-ordered sequence of position updates issued by oi , where p tj i = posi(tj) is a position update. For a generic time t, t ≥ t0, the most recently known position of oi before t is denoted by pˆ t i , and defined as follows: pˆ t i = p tm i ∈ Pi if tm < t ≤ tm+1
The goal of a generic k-NN query qi = (x, y, k), is to find the k closest objects to its center (x, y). Since we are dealing with moving objects, and the processing time t 0 of the query may be later than its issuing time, we have to consider the most recently known positions of objects.

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

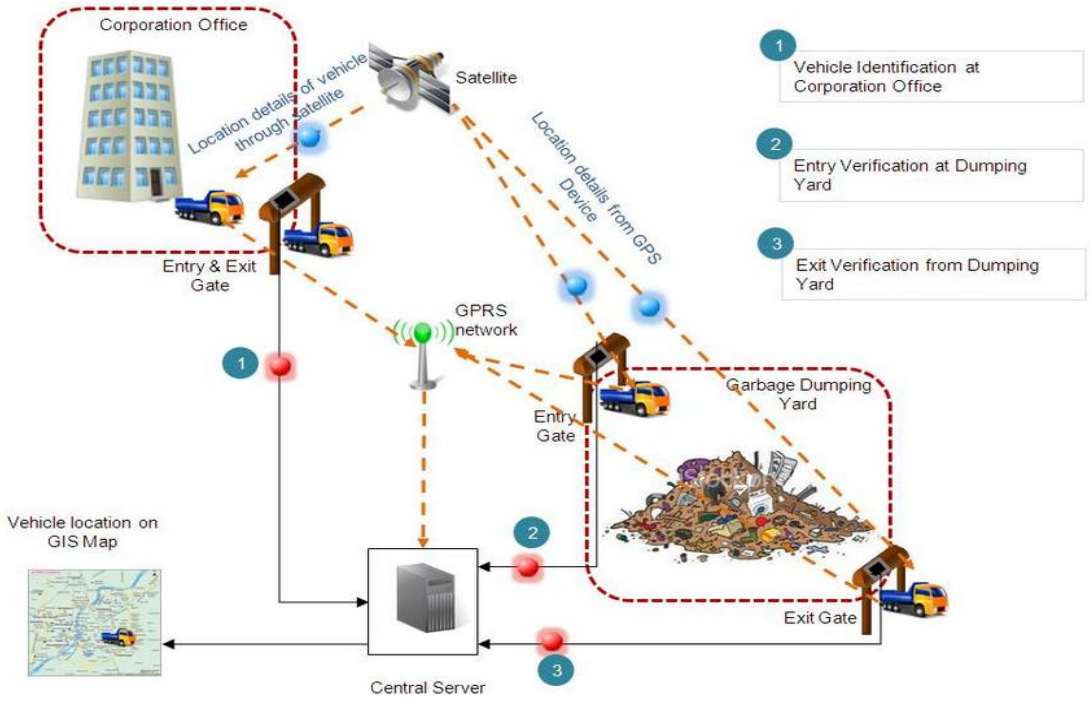
Vol. 5, Issue 3, March 2016

V. LITERATURE SURVEY

A major challenge for processing k-NN queries lies in the sheer volume of data and concurrent queries. A recent study estimates that the global pool of generated personal location data was at least 1PB in 2009 and that it is growing by about 20 percent a year. The abundance of such data gives rise to a variety of location-based applications and services, which must be able to effectively handle a large quantity of user-initiated concurrent queries (many of which are k-NN queries) in addition to managing the vast volume of data. For example, the number of users of We Chat, a free social networking app installed on smart phones, has exceeded 300 million as of January 2013 . One of its functionalities is to allow the users to locate their nearest fellow users upon request. In the new era of big data, it is imperative to find solutions that can effectively support the processing of many concurrent k-NN queries over large volumes of moving objects data. [1]Scalable Searching is a new but increasingly mature model of enterprise IT infrastructure that provides on-demand high quality applications and services from a shared pool of configuration cloud resources .The data mining, individuals or enterprises, can outsource their local complex data system into the data to avoid the costs maintaining a private storage as possesses powerful functionality and flexibility. However, some problems may be caused in this circumstance since the web application possesses full control of the outsourced data. [5]

Querying scheme supporting both user and user to access and update the data and dynamic update over data. We make contributions mainly in two aspects: prediction and showing store the data securely and distribute data securely to the user for more accurate result and . In term of accuracy, we adopt the vector space model combined with cosine measure, which is popular in information retrieval field, to evaluate the similarity between search request and document and acquire accurate search result instead of undifferentiated result. For the efficiency aspect, we propose data mining technique. The Self-destruction the data after the time allocation period data will cannot be access and attribute parameters will be set to the user So particular user will access the data. Our designed scheme can provide insertion and deletion update. We propose a secure scheme to meet privacy requirements in the threat model. [5]

VI. ARCHITECTURE OF THE SYSTEM



ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 3, March 2016

Explanation-

The large volume of data and heavy query workloads call for new scalable solutions. We propose DSI, a distributed strip index, and DKNN, a distributed k-NN search algorithm, to address this challenge. Both DSI and DKNN are designed with distributed processing in mind, and can be easily deployed to a distributed system. DSI is a data partitioning index and is able to adapt to different data distributions. Based on DSI, we present the DKNN algorithm that can directly determine a region that is guaranteed to contain the k-NNs for a given query with only two iterations. This has a clear cost benefit when compared with existing approaches, such as grid-based methods, which require an uncertain number of iterations. We show how the proposed index and algorithm can be implemented with S4. Extensive experiments confirm the superiority of the proposed method. We would like to explore how to evaluate continuous k-NN queries over moving objects using the strip index. For a given k-NN query q, it is very possible that its result (a list of objects) remains relatively stable when objects move with reasonable velocities. Therefore, it is promising to investigate how the k-NN results can be incrementally updated as objects move.

VII. EXPERIMENTAL SET-UP

We used three different synthetically generated data sets in our experiments. They contain the same number of objects but with increasing degree of skewness as shown in Figure. 1% of the objects in the skewed data set are distributed uniformly over the whole region, and the other 99% of the objects form four Gaussian clusters with randomly selected centers and standard deviation 0.05. The highly-skewed data set contains ten Gaussian clusters with randomly selected centers and standard deviation 0.02. We also used a data set generated based on the road networks of the state of Illinois. Objects start near the major intersections, and then randomly move along the roads. A snap shot of the simulation is shown in Figure. Only 1,000 objects are shown in the figure, but up to 100,000 objects are used in the actual experiments.

Verification of Analytical Results

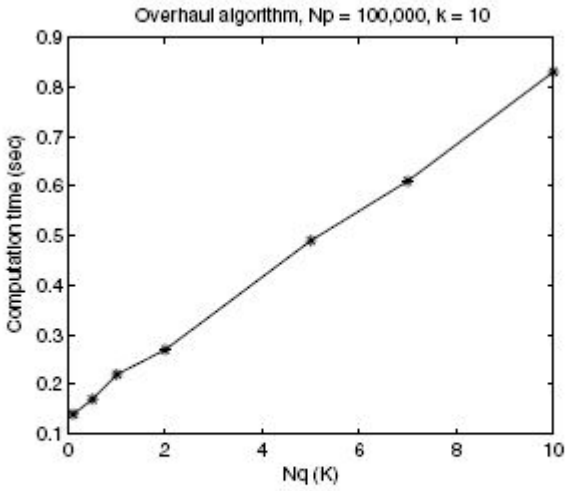
To verify the analytical results1 obtained in Section 3, we conduct experiments on data sets with the uniform data set and uniformly distributed queries. In all the experiments, we assume that *v*max is 0.005 unless otherwise specified. This means that an object's displacement in both *x*and *y* directions in each cycle is uniformly distributed within the range [−0.005, 0.005]. For the overhaul one-level Object-Indexing algorithm, shows that indeed the computation time is linear with respect to the number of queries *NQ*, verifying our analysis. The index-building and query-answering times shown in verify our analytical expectation: building the index requires linear time with respect to *NP* while query answering time is nearly constant. In Section 3, we described an incremental index maintenance algorithm, and showed that its performance depends on *v*max and δ. It. compares the performance of the incremental maintenance algorithm with that of the overhaul rebuild method as *v*max varies. Naturally, the cost of a complete rebuild of the index does not depend on *v*max, while the cost of incremental update goes up as *v*max increases. In this experimental setting (*NP* = 100, 000,*NQ* = 5, 000), the cross-over point of these two methods is at around *v*max = 0.0015. This implies that when the displacement of objectsbetween consecutive cycles is relatively small, it is worthwhileto use the incremental update algorithm, while when the objects move rapidly, a complete rebuild is preferred. It shows the performance of incremental query answering with object-index. Note that in Equation 1, when *NP* is small, the second term ($b1\mu\sqrt{NP}$) dominates, and therefore the cost of query answering is of $O(\sqrt{NP})$. However, as *NP* increases, the third term ($b2\mu 2NP$) becomes more significant, which propels the query answering cost into the order of $O(NP)$. This is confirmed in Figure 13, where the cost of query answering is of $O(\sqrt{NP})$.

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

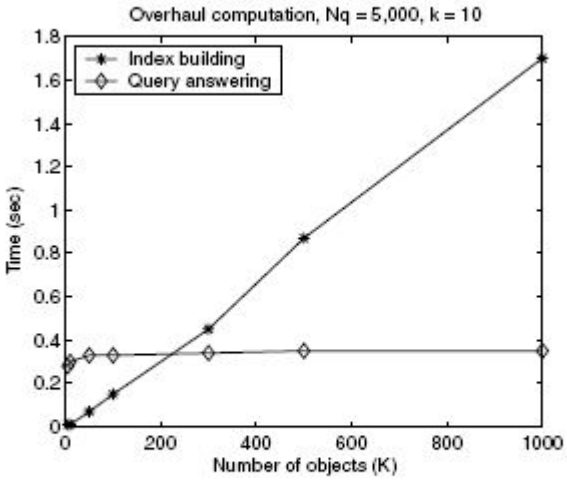
International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 3, March 2016



(a) Performance of overhaul computation w.r.t N_Q



(b) Performance of overhaul computation w.r.t N_P

Number of objects (*NP)* is less than 50,000, while the costincreases linearly with respect to *NP* for *NP* greater than 100,000. Figure 14 confirms our analysis for the index building time in Query-Indexing. The index building time shows a similar trend with respect to *NP* as the query answering time does in Object-Indexing, as pointed out in Section 3. The performance of Object-Indexing and Query- Indexing algorithms is compared.

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 3, March 2016

VIII. CONCLUSION

The problem of processing k-NN queries over moving objects i s fundamental in many applications. The large volume of data an d heavy query workloads call for new scalable solutions. We propose DSI, a distributed strip index, and D KNN, a distributed k-NN search algorithm, to address this challenge. Both DSI and DKNN aredesigned with distributed processing in mind, an d can be easily deployed to a distributed system. DSI is a data partitioning index and is able to adapt to different data distributions. Based on D SI, we present the DKNN algorithm that can directly deter mine a region that is guaranteed to contain the k -NNs for a given query with only two iterations. This has a clear cost benefit when compare d with existing approaches, such as g rid-based methods, which require an uncertain number of iterations. We show how the proposed index and algorithm can be implemented with S4. Extensive experiments confirm the superiority of the proposed method.

REFERENCES

[1] J. Manyika, M. Chui, B. Brown, J. Bughin, R. Dobbs, C. Roxburgh, and A. H. Byers, Big Data: The Next Frontier for Innovation, Competition, and Productivity. New York, NY, USA: McKinsey Global Institute, 2011.
[2] Wechat [Online]. Available: http://en.wikipedia.org/wiki/ WeChat, 2014. [3] X. Yu, K. Pu, and N. Koudas, "Monitoring k-nearest neighbor queries over moving objects," in Proc. 21st Int. Conf. Data Eng., 2005, pp. 631–642.
[4] B. Zheng, J. Xu, W.-C. Lee, and L. Lee, "Grid-partition index: A hybrid method for nearest-neighbor queries in wireless locationbased services," VLDB J., vol. 15, no. 1, pp. 21–39, 2006.
[5] K. Mouratidis, S. Bakiras, and D. Papadias, "Continuous monitoring of spatial queries in wireless broadcast environments," IEEE Trans. Mobile Comput., vol. 8, no. 10, pp. 1297–1311, Oct. 2009.
[6] J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," Commun. ACM, vol. 51, no. 1, pp. 107–113, 2008.
[7] G. S. Iwerks, H. Samet, and K. Smith. Continuous k-nearestneighbor queries for continuously moving points with updates. In VLDB, 2003.
[8] D. V. Kalashnikov, S. Prabhakar, and S. E. Hambrusch. Main memory evaluation of monitoring queries over moving objects. Distrib. Parallel Databases, 15(2):117–135, 2004.
[9] G. Kollios, D. Gunopulos, and V. J. Tsotras. Nearest neighbor queries in a mobile environment. In STDM, 1999.
[10] F. Korn, N. Sidiropoulos, C. Faloutsos, E. Siegel, and Z. Protopapas. Fast nearest neighbor search in medical imagedatabases. In VLDB, 1996.
[11] M. L. Lee, W. Hsu, C. S. Jensen, B. Cui, and K. L. Teo. Supporting frequent updates in r-trees: A bottom-up approach. In VLDB, 2003.
[12] D. Papadias, Q. Shen, Y. Tao, and K. Mouratidis. Group nearest neighbor queries. In ICDE, 2004.
[13] K. Raptopoulou, A. Papadopoulos, and Y. Manolopoulos. Fast nearest-neighbor query processing in moving-object databases. GeoInformatica, 7(2):113–137, 2003.
[14] N. Roussopoulos, S. Kelley, and F. Vincent. Nearest neighbor queries. In SIGMOD Conference, 1995.
[15] S. Saltenis, C. S. Jensen, S. T. Leutenegger, and M. A. Lopez. Indexing the positions of continuously moving objects. In SIGMOD Conference, 2000.