PRIVACY AWARE SPATIAL QUERIES

K.B. Priya Iyer
Research Scholar, Sathyabama University
Associate Professor, MOP Vaishnav College for Women (Autonomous), India
priya_balu_2002@yahoo.co.in

Dr. V. Shanthi
Professor, Department of M.C.A
St. Joseph College of Engineering, India
drvshanthi@yahoo.co.in

Abstract
The advancement in mobile communications and its integration with Geographical Information System result in tremendous increase in Location aware computing. Users thirst for Geo-Point of interest leads to exploration of different classes of spatial queries like nearest neighbor, range queries etc in location based computing. Each query type is unique and there is no frame work to combine these spatial queries. In this paper, we introduce a PASQAR: Privacy aware Spatial Query Assessor on Road Networks that processes the different types of queries based on user inputs. Further PASQAR masks the user identity using encryption technique. The experimental evaluation reflects result of applying various optimization techniques in query processing and proves the efficiency of PASQAR model.

Keywords
Spatial queries, Spatial Databases, Nearest Neighbor, Skyline Queries, Shortest Path, GIS, Location based services, GPS.

1. Introduction

Smartphone are revolutionizing the world of communications with their wide range of services offering to millions of users world-wide. With integration of global positioning system (GPS) and GIS, the Smartphone not only provide people communication but also disseminate temporal and spatial information. The Smartphone have become common in people's daily life. The people's network is further strengthening by technologies like Wi-Fi, Bluetooth and satellites.

A modern smart phone generally contain, a camera; accelerometer; microphone; speakers; and Wi-Fi, Bluetooth, and cellular connectivity. These also have opened new possibilities of developing Location based Applications. In modern geographic information systems, spatial queries represent important class in location aware computing. These spatial queries let users to query the public servers to retrieve their point of interest relative to their location. The server processes the request and sends back the query result to the user. With the increase in user search for geo-location, the different types of spatial queries like Nearest Neighbor, Aggregate queries, Range queries, skyline queries, Keyword queries etc are supported by modern location based apps. In spite of providing enhanced functionalities, these Location based apps opens security and privacy issues. While the user enjoys the service, they pay the penalty of disclosing their private data to these public location based servers.

All the previous studies on spatial queries are directed to either nearest neighbor or range or skyline or keyword queries. There is no single framework for processing all the queries under single application. Futher travel time based apps for point of interest are limited to nearest neighbor and skyline. Also privacy aware optimized query processing framework is lacking in previous studies. We combine four types of spatial queries [36][37][38][39] namely Goal directed kNN queries, goal directed skyline queries, predictive skyline queries, spatial Boolean skyline queries under single framework and the expert system processes the algorithm based on the user query request.

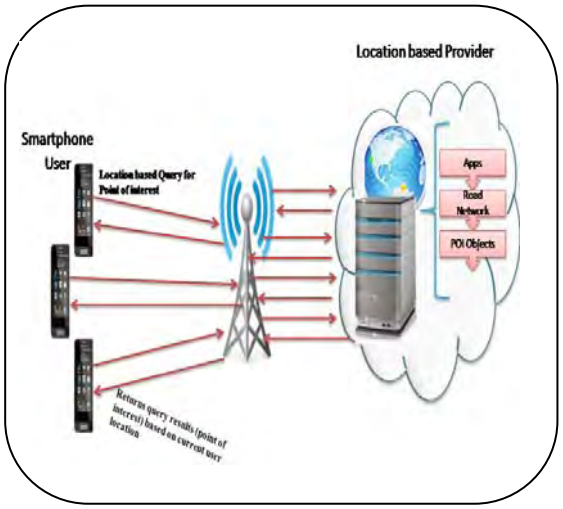


Figure 1: LBS Architecture

To sum up we make the following contributions:

1. PASQAR uses Expert System rules to evaluate user queries.
2. Finds data objects for any spatial query by considering distance and travel time as metric and gives shortest path to reach the point of interest..
3. Privacy is maintained through encryption technique.
4. Different Query optimization techniques are applied on spatial queries to increase the performance of the PASQAR.
5. Query caching in PASQAR helps in reduction of query processing cost of future queries.

The reminder of this paper is organized as follows. In section 2, we review the related work on both skyline queries and developing intelligent transportation System. In section 3, we formally defined Predictive Skyline query in Road Networks. In section 4, we introduce Fuzzy based approach for predictive skyline data objects in road networks. Section 5 presents the results of our experimental evaluation of our proposed approaches with a variety of Spatial Network with large number of data and query objects. Finally section 6 concludes the paper with future research.

2. Related work

2.1 kNN Queries in Euclidean Space

In the past, numerous algorithms have been proposed to solve kNN problem in Euclidean space. Most of these algorithms use R-Tree structure for query processing. The Tree based structure lack in frequent location update of moving objects. Later space based indexing structure (grid) are used. All of these approaches consider distance between static objects as a function of Euclidean distance.

2.2 kNN Queries in Spatial Networks

In [33], Papadias et al. introduced Incremental Network Expansion (INE) and Incremental Euclidean Restriction[IER] methods to support kNN queries in spatial networks. In [21], Kolahdouzan and shahabi proposed voronoi diagrams to partition the spatial network to voronoi polygons, one for each data object. In [6], cho et al. presented a system UNICONS where the main idea is to integrate the precomputed k nearest neighbors into dijkstra algorithm. In [16], Huang et al. proposed a island approach to knn. In-Route Nearest Neighbor was first proposed by Shekar et al. in [47] to search minimum detour distance on the way to destination. In [62], Zaiben chen et al. Best Fit Network expansion is introduced for network distance. All of the above studies consider network distance as static where in real fast world shortest path computation depends on travel time.

2.3 Single-source Skyline query in Euclidean space

Skyline query processing has been studied extensively in recent years. The *skyline* operator was first introduced into the database community by Borzsonyi et al. [3]. Borzsonyi et al. [3] propose the Block-Nested-Loops algorithm (BNL) and the Extended Divided-and-Conquer algorithm (DC). Both algorithms processes the entire

object set for retrieving the skyline data. In [5], the Sort-Filter-Skyline algorithm (SFS) progressively report skyline points by pre-sorting the entire dataset according a preference function. Tan et al.[51] propose a bitmap-based method which transforms each object to bit vectors. Bitmaps can be very large for large values. This method cannot guarantee a good initial response time. Kossmann et al. [22] propose an online nearest neighbor skyline query processing method which can progressively report the skyline points in an order according to user's preference. The objects in the dominated subspace are pruned, and the objects in each non-determined subspace form a to-do list. To remedy this problem, Papadias et al. [32] propose an R-tree based algorithm, Brand and Bound Skyline (BBS), which retrieves skyline points by browsing the R-tree based on the best-first strategy. BBS only visits the intermediate nodes not dominated by any determined skyline points. This method has more efficient memory consumption than the method in [22]. Another study by Lin et al. [26] processes a skyline point query against the most recent N elements in a data stream.

2.4 Skyline variants

There are many variants of the traditional skyline query. Pei et al. [34] and Yuan et al. [58] proposed methods to compute skylines in all possible subspaces. Tao et al. [52] gave an efficient algorithm to calculate skylines in a specific subspace. Dellis and Seeger [9] proposed a reverse skyline query, which obtains those objects that have the query point as skyline, where each attribute is defined as the absolute difference from objects to query point along each dimension. In the context of uncertain databases, Pei et al. [35] proposed the probabilistic skyline over uncertain data, which returns a number of objects that are expected to be skylines with probability higher than a threshold. The most relevant problems to our work are the dynamic skyline[32], spatial skyline [45], multi-source skyline on road networks [10], multi-preference path planning approach[23] and continuous probabilistic skyline queries over uncertain data streams[15]. Specifically, Papadias et al. [32] applies BBS algorithm to retrieve skyline points, where dynamic attributes of data objects are computed by a set of dimension functions. However, only Euclidean distance was considered for dimension functions. Similarly, the method proposed for multi-source skyline on road networks [10] also utilizes geometric information of data objects during the pruning, which is thus limited to road network application. In [17], retrieves the top-k skyline keyword queries in road networks. In [54], retrieves skyline keyword queries for moving objects.

2.5 Travel time Studies

In [8], Cooke and Halsey introduced the first time-dependent shortest path (TDSP) solution where they formulated the problem in discrete time and use dynamic programming. In [18], Kanoulas et al. introduced a Time-Interval All Fastest Path (allFP) algorithm in time-dependent networks where all the paths from the source to a destination node are enumerated which incurs exponential running time in the worst case. In [60], Ugur et al. proposed time dependent spatial network databases where network expansion suffers in large networks.

There has been a significant amount of work identifying the importance of accurate travel time predictions in a transportation system. From the travelers' perspective, accurate travel time predictions reduce the uncertainty in decision making about departure time and route choice, which in turn reduce travelers' stress and anxiety. From the operators' point of view, travel time prediction models may be used to determine the reliability of a transportation system. Consequently, travel time prediction methods are central to Advanced Traveler Information Systems (ATIS) and Advanced Traffic Management Systems (ATMS) [54][2]. Bus travel times are the result of nonlinear and complex interactions of many different constituent factors influencing either demand (e.g. passenger's demand or traffic flow) or capacity (e.g. accidents, weather condition, route characteristics) [28]. Mean-variance estimation method was also used in [29] to predict bus travel time variability. Travel Time Prediction Using floating Car Data Applied to Logistics Planning by taking into account that routes of long-range trips are not completely given in advance but are rather unknown and subject to change[48]. short-term prediction of highway travel times, which represent an accurate estimation of the expected travel time for a driver commencing on a particular route based on the fusion of different types of data that come from different sources (inductive loop detectors and toll tickets) and from different calculation algorithms[46]. Car following model using a fuzzy inference system (FIS) to simulate and predict the future behavior of a Driver-Vehicle- Unit (DVU) based on a new idea for estimating the instantaneous reaction of DVU, as an input of fuzzy model[19].

In summary, previous studies on spatial queries on road networks are limited to either Euclidean space or metric space for a specific application not under a single framework. In contrast, our work focuses on the travel time as a metric for spatial query search which play a vital role in road networks. User privacy is achieved through encryption. It also helps future queries computation time by caching technique.

3. System Model

In this section, we describe the road network and system model; define the predictive skyline query search in spatial networks. We assume a spatial network [California Road Network], containing set of static data objects as well as query objects searching their skyline objects. We assume all road maps and daily traffic data are maintained by cloud server.

3.1 Road Network

We model the underlying road network as a weighted undirected graph G = (V,E) where E is an Edge set of road segments in the road network, V is the Vertex set of intersection points of the road segment and each edge is given travel time of its corresponding road segment as weights.

Figure 2: Road Network

In this model (Figure 1), we consider our system with a mobile environment in which mobile user is able to communicate with the service provider through wireless communication infrastructure e.g.: Wi-Fi.

3.2 Query Definitions

Example1 (GD-kNN): if a user wants closest petrol shop in the direction of his journey.

Example2 (GD-skyline): if a user wants restaurant closest to a hospital in the direction of his journey.

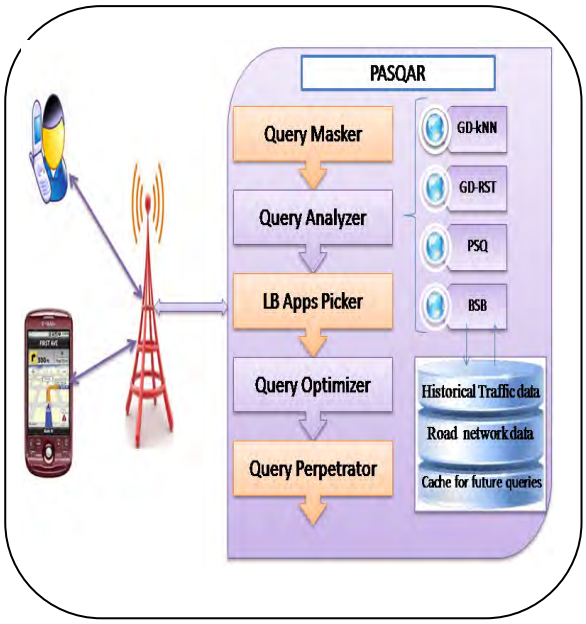
Example3 (Predictive Skyline Query): if a user wants to know the travel time for his future journey schedule.

Example4 (Boolean skyline boundary query): if a tourist wants to find a low price A2B restaurant nearer to a temple and close to a beach.

4. Algorithm

As a pre-computation, the road network is partitioned into grids to reduce the search space in finding out the nearest neighbor vertex of the user location. We apply a road network clustering approach to efficiently compute the nearest vertex of the query origin and reduce the search space by considering only nodes in user direction of travel.

Figure 3: PASQAR Architecture



The PASQAR consists of five phases namely Query Masker, Query Analyzer, LB Apps Picker, Query Optimizer and Query perpetrator.

Phase I: Query Masker

User privacy is achieved through the Query Masker. The given user query is encrypted through encryption.

The wrapped query is sent to Peer; the Peer then forwards the query to the LBS public server. The server decrypts the wrapped query and process the query. The query results are in turn encrypted by the server and are sent back to the Peer. The Peer then forwards the wrapped query results to the User. The User uses his decryption algorithm to get the exact query result or his desired point of interest. The advantage of this Query Masker is the peer identity is only stored in the server and on the other side the query initiator user preferences are even not known to the peer as only wrapped query is passing in the hands of the peer.

Phase II: Query Analyzer

This phase is responsible for analyzing the User query. The given query is parsed into segments and make ready for the expert system engine to process.

Phase III: LB Apps Picker

The Expert system Engine gets the query segments from query Analyzer and applies the rules for selecting the appropriate algorithm. The following is set of rules for processing the given user query:

Phase IV: Query Optimizer

Query optimization plays a crucial role in retrieving information from the public server. It is a technique of analyzing several query execution patterns and a fine query pattern is identified for further process. The factors that affect the optimization are CPU time, amount of memory space, storing and fetching time etc. this helps in faster query processing, lesser cost per query, high performance of the system, efficient usage of database. PASQAR optimizations are:

a) Search Space reduction: In finding the nearest vertex of the query origin, the search space is reduced by the road network clustering technique. Instead of searching the entire database, the nodes in the user cluster are examined in finding nearest neighbor node.

b) Goal directed search: Instead of searching the entire database for finding the point of interest, only the objects in the user direction of travel is considered. This helps in greater reduction of database access.

Algorithm 1: GD kNN (upos,p,dist,time,goal,n)

/* upos: query origin (latitude,ongitude) , p: point of interest, dist: user distance search limit, time: user travel time search limit, goal: user direction of travelling, n: number of point of interest */

```
1.  u ← NearestVertex(Upos)
2.  k=0
3.  while true do
4.  minnode ←Leastcostneighbournode(u)
5.  hashpath(i) ← addnodetohashpath(minnode)
6.  if (checkpoi(minnode) ) then
7.   Dispshortestpath(hashpath(i))
8.   k=k+1
9.   end if
10.  Prev ←u
11. u ← explore_nextshortestpathnode()
12. if (k>=n)
13.  break
14. while end
```

Algorithm 2: GD-RST (uloc,pref[],A[])

/* upos: query origin (latitude,ongitude) , pref[], an array consisting of user query preferences Q1,Q2,..Qn and A[], an array of their corresponding attributes a1,a2,..an */

```
1. u ← NearestNeighborNode(uloc)
2. C[i]={0
3. C[i] ←Generate(Q1)
4. Heapset[i] ← addQPtoheap(C[])
5.  if (checkskyline(Heapset[]) ) then
6.       Dispskylineobject()
7.      Cacheskyline(C[], Heapset[])
8.  end if
```

c) Skyline Objects/Keyword queries: While executing skyline/keyword queries, the objects are filtered such that skyline/keyword object matches are executed first than retrieving all objects.

d) Multiple Queries: When multiple queries are fired within same region different POI or different region with same POI, these are joined and excuted once in the database instead of individual executions which decreases the network cost.

e) Caching: The queries are cached which decreases the cost of future queries.

Phase V: Query Perpetrator

As decided by the LB Apps picker, the appropriate algorithm is executed by the PASQAR. The query results are sent back to the Peer in an encrypted form by the Query Masker which forwards again to the User. The User decrypts the results, gets the desired point of interest and shortest path to the POI.

Algorithm 3: PSQ (uloc,pref[],A[],journeyschedule)

/* upos: query origin (latitude,ongitude) , pref[], an array consisting of user query preferences Q1,Q2,..Qn and A[], an array of their corresponding attributes a1,a2,..an, journey schedule: date and travel time of journey */

```
1. u ← Getvertex(uloc/loc)
2. C[i]={0}
3. Predicted_doj ← fuzzymatlab_datepredictor()
4. Predicted_time ← fuzzymatlab_traveltimepredictor()
5. C[i] ←Generate(Q1
6. Heapset[i] ← addQPtoheap(C[])
7.  if (checkpredictiveskyline(Heapset[]) ) then
8.        Dispskylineobject()
9.       Queryrecorder(C[], Heapset[])
10. end if
```

Algorithm 4: BSB (uloc,pref[],A[],B[],boundary)

/* upos: query origin (latitude,ongitude) , pref[], an array consisting of user query preferences Q1,Q2,..Qn, A[], an array of their corresponding attributes a1,a2,..an , B[] array of boolean operators for user objects, boundary is spatial region coordinates */

```
1. u ← StartNode(uloc)
2. C[i]={0}
3. C[i] ←FilterkeywordObject(Q1)
4. Heapset[i] ← addQPtoheap(C[])
5.  if (checkBSB(Heapset[]) ) then
6.        DispBSBobject()
7. end if
```

5. Experimental Evaluation

5.1 Experimental Setup

We conducted experiments on California road network which contains 21,050 nodes and 21693 edges. The algorithm is implemented in Java and tested on Windows Platform with Intel Core 2 CPU and 80GB memory. The main metric we adopt is CPU time that reflects how much time the algorithm takes in processing a skyline query. The input map is extracted from Tiger/Line files that are publicly available [67].

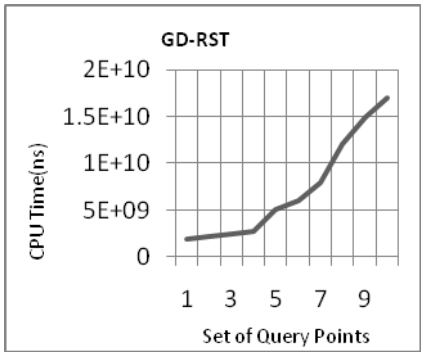


Figure 4(a)

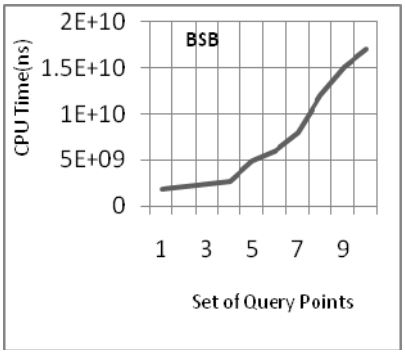


Figure 4(b)

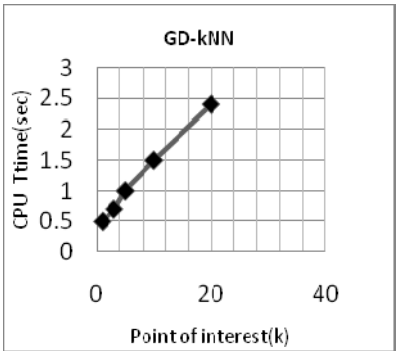


Figure 4(c)

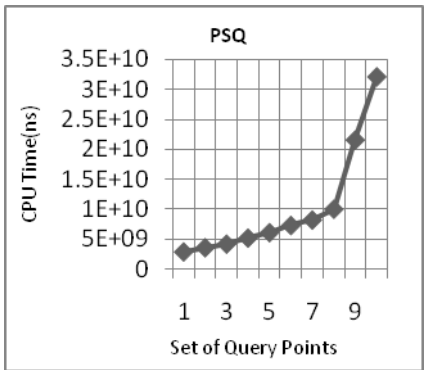


Figure 4(d)

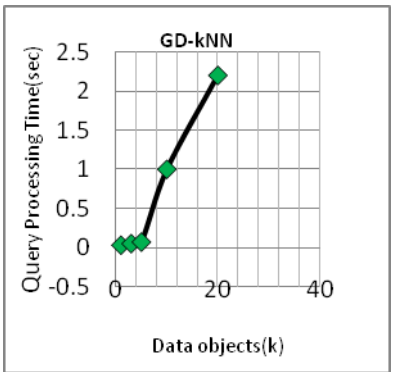


Figure 4(e)

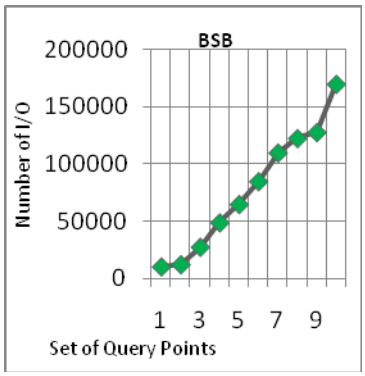


Figure 4(f)

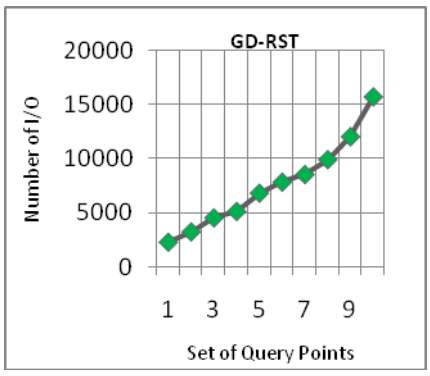


Figure 4(g)

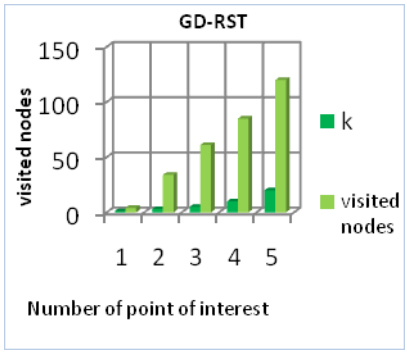


Figure 4(h)

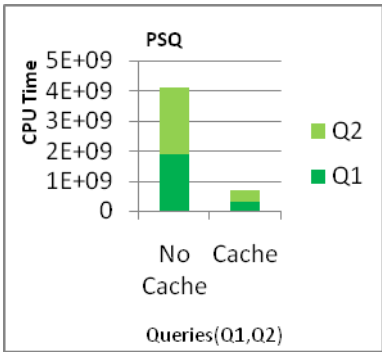


Figure 4(i)

5.2 Results

i) Impact of CPU time on set of Query Points

Fig 4(a), 4(b), 4(c), 4(d) shows the impact of CPU on set of query points with GD-kNN, GD-RST, PSQ and BSB algorithms.

ii) Impact of k on Query Processing Time: With this experiment in Fig. 4(e), we show the impact of number of point of interest with their query processing time (seconds) in GD-kNN algorithm.

iii) Impact of I/O on **set** of Query Points: With this experiment in Fig. 4(f), 4(g), we show the impact of I/O on number of Query points in GD-RST and BSB algorithm.

iv) Impact of Cache in query computation: Fig 4(i), we show the impact of cache technique in computing future queries. The figure shows for processing future queries (i.e. if same query is asked by same user/another user at

any point of time and if query is in cache then the result of the query is taken from cache instead of computing it again. The CPU time reduction in using Cache technique is shown here.

v) Impact of k on visited nodes: In Fig. 4(h), we show the impact of k on number of nodes accessed on both time, distance based query.

6. Conclusion

In this paper, we propose a framework for executing different types of queries on road networks. The SQAR analyses the user query based on the inputs and automatically executes the corresponding algorithm. User identity is further maintained by the encryption technique. Query optimization is also achieved by the SQAR model. The algorithm also caches the query results to reduce the computation cost for future queries.

Additional future works includes query for moving objects, different types of transportation mode, road types.

References

[1] Baihua Zheng, K.C.K. Lee, and Wang-Chien Lee, "Location-Dependent Skyline Query," in 9th International Conference on Mobile Data Managemen, Beijing, 2008, pp. 148-155.

[2] B. Bartin and K. Ozbay. 2010. Determining the optimal configuration of highway routes for real-time traffic information: A case study. IEEE Transactions on Intelligent Transportation Systems, vol. 11, no. 1, pp. 225–231.

[3] S. Borzsonyi, D. Kossmann, and K. Stoker. The skyline operator.ICDE, 2001.

[4] X. Cao, G. Cong, C. S. Jensen, and B. C. Ooi. Collective spatial keyword querying. In SIGMOD, 2011.

[5] J. Chomicki, P. Godfrey, J. Gryz, and D. Liang. Skyline with processing. ICDE, 2003.

[6] H.-J. Cho and C.-W. Chung. An efficient and scalable approach to cnn queries in a road network. In VLDB, 2005.

[7] L.Cooke and E.Halsey. The Shortest route through a network with time dependent intermodal transit times. In journal of Mathematical Analysis and Applications, 1966.

[8] B. C. Dean. Algorithms for minimum cost paths in time-dependent networks. In Networks, 1999.

[9] E. Dellis and B. Seeger. Efficient computation of reverse skyline queries. In Proc.

[10] Ke Deng, Xiaofang Zhou, and Heng Tao Shen, "Multi-source skyline query processing in road networks," in IEEE 23rd International Conference on Data Engineering, Istanbul, 2007, pp. 796 – 805.

[11] B. Ding, J. X. Yu, and L. Qin. Finding time-dependent shortest paths over large graphs. In EDBT, 2008.

[12] S. E. Dreyfus. An appraisal of some shortest-path algorithms. In Operations Research Vol. 17, No. 3, 1969.

[13] B. George, S. Kim, and S. Shekhar. Spatio-temporal network databases and routing algorithms: A summary of results. In SSTD, 2007.

[14] YK Huang, ZW Chen. Continuous k-nearest neighbor query over moving objects in road networks. Advances in Data and Web Management, 2009 – Springer.

[15] Hui Zhu Su, En Tzu Wang and Arbee L. P. Chen. Continuous probabilistic skyline queries over uncertain data streams. Database and Expert Systems Applications, 2011 Springer.

[16] H.X., J.C.S., and S. Saltenis. The island approach to nearest neighbor querying in spatial networks. In SSTD, 2005.

[17] João B. Rocha-Junior_and Kjetil Nørvåg Top-k Spatial Keyword Queries on Road Networks. EDBT 2012, March 26–30, 2012.

[18] E. Kalnoulas, Y.Du, T. Xia, and D.Zhang. Finding fastest paths on a road network with speed patterns. In ICDE, 2006.

[19] Khodayari A, Kazemi R, Ghaffari A,Braunstingl R. 2011. Design of an improved fuzzy logic based model for prediction of car following behavior. IEEE International Conference on Mechatronics (ICM), On Page(s): 200 – 205.

[20] M. R. Kolahdouzan and C. Shahabi. Continuous k-nearest neighbor queries in spatial network databases. In STDBM, 2004.

[21] M. Kolahdouzan and C. Shahabi. Voronoi-based k nearest neighbor search for spatial network databases. In VLDB, 2004.

[22] D. Kossmann, F. Ranmsak, and S. Rost. Shooting stars in the sky: An online algorithm for skyline queries. VLDB, 2002.

[23] Kriegel, H.-P.; Renz, M.; Schubert, M. Route skyline queries: A multi-preference path planning approach. in IEEE 26th International Conference on Data Engineering (ICDE),2010.

[24] R. Kung, E. Hanson, Y. Ioannidis, T. Sellis, L. Shapiro, and M. Stonebraker. Heuristic search in data base system. Proc.1st International Workshop on Expert Database Systems, 1986.

[25] G Li, Y Li, LC Shu . CkNN query processing over moving objects with uncertain speeds in road networks. Web Technologies and Applications, 2011 – Springer.

[26] X. Lin, Y. Yuan, W. Wang, and H. Lu. Stabbing the sky: Efficient skyline computation over sliding windows. ICDE, 2005.

[27] Mehdi Sharifzadeh, Cyrus Shahabi, and Leyla Kazemi, "Processing spatial skyline queries in both vector spaces and spatial network databases," ACM Transactions on Database Systems, vol. 34, no. 3,pp. 1-43, August 2009.

[28] E. Mazloumi, G. Currie, and G. Rose. Using GPS data to gain insight into public transport travel time variability," Journal of Transportation Engineering, vol. 136, no. 7, 2010, pp. 623–631.

[29] E. Mazloumi, G. Rose, G. Currie, and S. Moridpour. Prediction intervals to account for uncertainties in neural network predictions: Methodology and application in bus travel time prediction, Engineering Applications of Artificial Intelligence, 2010.

[30] Orda and R. Rom. Shortest-path and minimum-delay algorithms in networks with timedependent edge-length. J. ACM, 1990.

[31] D. Papadias, Q. Shen, Y. Tao, and K. Mouratids. Group nearest neighbor queries. ICDE, 2004.

[32] D. Papadias, Y. Tao, G. Fu, and B. Seeger. Progressive skyline computation in database systems. ACM Trans.Database Syst., 30(1):41–82, 2005.

[33] D. Papadias, J. Zhang, N. Mamoulis, and Y. Tao. Query processing in spatial network databases. In VLDB, 2003.

[34] J. Pei, W. Jin, M. Ester, and Y. Tao. Catching the best views of skyline: a semantic approach based on decisive subspaces. In Proc. 31th Int. Conf. on Very Large Data Bases, pages 253–264, 2005.

[35] J. Pei, B. Jiang, X. Lin, and Y. Yuan. Probabilistic skylines on uncertain data. In Proc. 33th Int. Conf. on Very LargeData Bases, pages 15–26, 2007.

[36] Priya Iyer K.B, V. Shanthi. Goal Directed Relative Skyline Queries in Time Dependent Road Networks. IJDMS, Vol.4, No.2, April 2012.

[37] Priya Iyer K.B, V. Shanthi. Geo-calendar based Predictive Skyline Queries using Fuzzy Inference Engine. IJCA, Volume 45, Number 15, May 2012.
[38] Priya Iyer K.B, V. Shanthi. Intelligent Path Finder for goal directed queries in Road Networks. Proceedings of MNCApps, AIEEE, August 2012, In press.
[39] Priya Iyer K.B, V. Shanthi. Spatial Boolean Skyline Boundary Queries in Road Networks. Proceedings of ICCCNT, July 2012, India. In press.
[40] Priya Iyer K.B, V. Shanthi. Location based Emergency Response Management System through Satellites in Road Networks. Proceedings of EEC 2012, July 2012. In press.
[41] Priya Iyer K.B, V. Shanthi. Location based Disaster Management System through Smart phones. Proceedings of ACCT, July 2012, India.
[42] M Safar, D Ibrahimi. Voronoi-based reverse nearest neighbor query processing on spatial networks. Multimedia Systems, 2009 – Springer.
[43] H. Samet, J. Sankaranarayanan, and H. Alborzi. Scalable network distance browsing in spatial databases. In SIGMOD, 2008.
[44] C. Shahabi, M. R. Kolahdouzan, and M. Sharifzadeh. A road network embedding technique for k-nn search in moving object databases. Geoinformatica, 2003.
[45] M. Sharifzadeh and C. Shahabi. The spatial skyline queries. VLDB, 2006.
[46] F. Soriguera F. Robusté. 2011. Highway travel time accurate measurement and short-term prediction using multiple data sources. Transportmetrica, Volume 7, Issue 1, pages 85-109.
[47] S.Shekar and J.S. Yoo. Processing in-route nearest neighbor search. In proceedings of GIS, pages 9-16, 2003.
[48] Simroth, A. Za hle, H. Travel Time Prediction Using Floating Car Data Applied to Logistics Planning. IEEE Transactions on Intelligent Transportation Systems, Volume: 12, Issue: 1, On Page(s): 243 – 253.
[49] Z. Song and N. Roussopoulos. K-nearest neighbor search for moving query point. In SSTD, 2001.
[50] Y. Tao and D. Papadias. Time-parameterized queries in spatio-temporal databases. In SIGMOD, 2002.
[51] Y. Tao, D. Papadias, and Q. Shen. Continuous nearest neighbor search. In VLDB, 2002
[52] Y. Tao, X. K. Xiao, and J. Pei. SUBSKY: Efficient computation of skylines in subspaces. In Proc. 22th Int. Conf. on Data Engineering, page 65, 2006.
[53] K.-L. Tan, P.-K. Eng, and B. Ooi. Efficient progressive skyline conputation. VLDB, 2001.
[54] D. Wu, M. Yiu, C. Jensen, and G. Cong. Efficient continuously moving top-k spatial keyword query processing. In ICDE, 2011.
[55] Yang Du, Donghui Zhang, and Tian Xia, "The Optimal-Location Query," in 9th International Symposium Advances in Spatial and Temporal Databases, Angra dos Reis, 2005, pp. 163-180.
[56] M. Yang, Y. Liu, and Z. You. 2010. The reliability of travel time forecasting. IEEE Transactions on Intelligent Transportation Systems, vol. 11, no. 1, pp. 162–171.
[57] M. Yiu, N. Mamoulis, and D. Papadias. Aggregate nearest neighbor queries in road networks. TKDE, 17(6):820–833, 2005.
[58] Y. Yuan, X. Lin, Q., W. Wang, J. Xu Yu, and Q. Zhang. Efficient computation of the skyline cube. In Proc. 31th Int. Conf. on Very Large Data Bases, 2005.
[59] Ugur Demiryurek, Farnoush Banaei-Kashani and Cyrus Shahabi. Towards K-Nearest Neighbour Search in Time-Dependent Spatial network databases.
[60] K Xuan, G Zhao, D Taniar, M Safar . Voronoi-based multi-level range search in mobile navigation. Multimedia Tools 2011 – Springer.
[61] J. S. Yoo and S. Shekhar. In-route nearest neighbor queries. GeoInformatica, 9(2):117–137, 2005.
[62] Zaiben Chen, Heng Tao Shen, Xiaofang Zhou and Jeffrey Xu Yu. Monitoring Path Nearest Neighbour in Road Networks. SIGMOD 2009.
[63] G Zhao, K Xuan, W Rahayu, D Taniar. Voronoi-based continuous k nearest neighbor search in mobile navigation. Industrial 2010.
[64] B. Zheng, J. Xu, W.-C. Lee, and D. L. Lee, "Grid-partition index: a hybrid method for nearest-neighbor queries in wireless location-based services," VLDB J., vol. 15, no. 1, pp. 21–39, 2006.
[65] J. Zhang, M. Zhu, D. Papadias, Y. Tao, and D. L. Lee. Location-based spatial queries. In SIGMOD, 2003.
[66] B. Zheng and D. L. Lee. Semantic caching in location-dependent query processing. In SSTD, 2001.
[67] Tiger/Line: ww.census.gov/geo/www/tiger/.