International Journal of Scientific & Engineering Research, Volume 5, Issue 5, May-2014
ISSN 2229-5518

Effective Path Queries with Random Sequence Controls

S.Bhuvaneswari, T.S.Poonguzhali, K.Selvakumar

Abstract-We propose novel solutions to the general optimal route query, based on two different methodologies, namely backward search and forward search. In addition, we discuss how the proposed methods can be adapted to answer a variant of the optimal route queries, in which the route only needs to cover a subset of the given categories.The paper focus on operational time along with the distance. Extensive experiments, using both real and synthetic data sets, confirm that the proposed solutions are efficient and practical, and outperform existing methods by large margins.

Index Terms-Backward Search Query Processing, Batch Backward Search, Forward Search Query Processing, Spatial databases.

1 INTRODUCTION

Consider a tourist who will have a free day to travel around India. Without much knowledge about the city, s/he searches online maps to plan for a trip. Usually, s/he has a fixed starting point, e.g., her/his hotel, and certain objectives in mind, such as visiting a museum, enjoying a few minutes in a famous temple, and dining at a fine restaurant. Meanwhile, some destinations may need to be visited in a certain order. For instance, the trip should have a museum after a restaurant. The ideal route should cover all the destinations, satisfy all order constraints, and minimize the total travel length. Searching for such a route is captured by the optimal route query which usually has a vast search space. With the dataset the starting and destination points are defined. Then the intermediate points were defined and a visit order constraints is defined. Now the query is defined (Query contains starting point, destination point, visit order points).

Mrs.S.BhuvaneswariM.C.A,ResearchScholor, School of Computing Sciences, Vels University, Chennai, India.
PH-944352281,E-mail – bhun2912@gmail.com
Mrs.S.Poonguzhali M.C.A, M.Phil.,Assistant Professor, Department of M.C.A.Vels University,Chennai, India.,
PH-9677242859,E-mail – poonguzhali.research@gmail.com
Dr. K. Selva Kumar M.E. Ph.D.[3], Assistant Professor, Department of Mathematics, University VOC College of Engineering, Anna University(Thoothukudi Campus), Thoothukudi, ndia.PH-9566604812, E-mail- k_selvakumar10@yahoo.com

In [1] the work addresses a novel spatial keyword query called the m-closest keywords (mCK) query. Given a database of spatial objects, each tuple is associated with some descriptive information represented in the form of keywords. The mCK query aims to find the spatially closest tuples which match m user-specified keywords. To answer mCK queries efficiently, we introduce a new index called the bR*-tree, which is an extension of the R*-tree. Based on bR*-tree.In [2]authors study the efficient processing of continuously moving top-k spatial keyword (MkSK) queries over spatial keyword data. State-of-the-art solutions for moving queries employ safezones that guarantee the validity of reported results as long as the user remains within a zone. We exploit tight and conservative approximations of safe zones and aggressive computational space pruning.In [3]given two sets of points P and Q, a group nearest neighbor (GNN) query retrieves the point(s) of P with the smallest sum of distances to all points in Q. Consider, for instance, three users at locations q_1 q_2 and q_3 that want to find a meeting point (e.g., a restaurant); the corresponding query returns the data point p that minimizes the sum of Euclidean distances $|pq_i|$ for $1 \le i \le 3$. Assuming that Q fits in memory and P is indexed by an R-tree, we propose several algorithms for finding the group nearest neighbors efficiently. As a second step, we extend our techniques for situations where Q cannot fit in memory, covering both indexed and nonindexed query points. An experimental evaluation identifies the best alternative based on the data and query properties.In [4] authors determine for each edge a geometric object containing all nodes that can be reached on a shortest path starting with that edge. Based on these geometric objects, the search space for online computation can be reduced significantly.In [5] this specification order there should be information about both starting point and destination point of the travelling. The optimal route queries optimize the possible routes and give the optimal route that satisfies all the constraints. This paper describes the survey on optimal route query processing, two categoriesnamely optimal route query processing and spatial search with categorical information have been considered, a discussion on technique for optimal route query with constraints and without constraint is also included. The total order needs a specification of list of points and in the same order that they should be visited but that is not required for partial order constraints.

In [6] authors propose *WaveCluster*, a novel clustering approach based on wavelet transforms, which satisfies all the requirements. Using the multi resolution property of wavelet transforms, we can effectively identify arbitrarily shaped clusters at different degrees of detail.In [7] transit node routing—that allows us to reduce quickest-path queries in road networks to a small number of table lookups. We present two implementations of this idea, one based on a simple grid data structure and one based on highway hierarchies. In [8] authors develop methods for answering Constrained Nearest Neighbor queries with different properties and advantages. We prove the optimality (with respect to I/O cost) of one of the techniques proposed in this paper.In [9] authors present an efficient method to answer top-k spatial keyword queries. To do so, we introduce an indexing structure called IR^2-Tree (Information Retrieval R-Tree) which combines an R-Tree with superimposed text signatures. We present algorithms that construct and maintain an IR^2-Tree, and use it to answer top-k spatial keyword queries.In [10] authors use novel branch&bound search algorithms with lower bounds obtained from homomorphic abstractions of the original state space. Our method is asymptoticallyoptimal. In [11]computation of shortest paths in graphsis a central requirement for many applications. Route planning in traffic networks is perhaps the best-known one. Dijkstra's algorithm [Dij59] solves this problem efficiently, but query time may be intolerably long for large graphs.

In [12] this approach incurs a significant amount of repeated computations, and, thus, is not scalable to large data sets. Motivated by this, we propose novel solutions to the general optimal route query, based on two different methodologies, namely backward search and forward search. In addition, we discuss how the proposed methods can be adapted to answer a variant of the optimal route queries, in which the route only needs to cover a subset of the given categories. In [13] optimal Sequenced Route strives to find a route of minimum length starting from a given source location and passing through a number of typedlocations in a particular order imposed on the types of the locations. We first transform the OSR problem into a shortest path problem on a large planar graph. We show that a classic shortest path algorithm such as Dijkstra's is impractical for most real-world scenarios. Therefore, we propose LORD, a light threshold-based iterative algorithm, which utilizes various thresholds to prune the locations that cannot belong to the optimal route. Then we propose R-LORD, an extension of LORD which uses R-tree to examine the threshold values more efficiently.

In [14] on the basis of spatial data structures, and R-trees in particular, we propose a multi-tree index that follows the broad concept of augmenting nodes with additional information to accelerate queries. Augmentation is examined with respect to maximal/minimal points in subtrees, the properties of which are exploited by the proposed searching algorithm to effectively prune the search space. In [15] a frequently encountered type of queryin Geographic Information Systems is to find the k nearest neighbor objects to a given point in space. Processing such queriesrequires substantially different search algorithms than those for location or range queries.In [16] authors propose an architecture that integrates network and Euclidean information, capturing pragmatic constraints. Based on this architecture, we develop a Euclidean restriction and a network expansion framework that take advantage of location and connectivity to efficiently prune the search space.

In [17] this naive approach incurs a significant amount of repeated computations, and, thus, is not scalable to large datasets. Motivated by this, we propose novel solutions to the general optimal route query, based on two different methodologies, namely backward search and forward search. In addition, we discuss how the proposed methods can be adapted to answer a variant of the optimal route queries, in which the route only needs to cover a subset of the given categories. In [18] along with the complexity of the objects such as images, molecules and mechanical parts, also the complexity of the similarity models increases more and more. Whereas algorithms that are directly based on indexes work well for simple medium-dimensional similarity distance functions, they do not meet the efficiency requirements of complex high-dimensional and adaptable distance functions. The use of a multi-step query processing strategy is recommended in these cases, and our investigations substantiate that the number of candidates which are produced in the filter step and exactly evaluated in the refinement step is a fundamental efficiency parameter. After revealing the strong performance shortcomings of the state-of-the-art algorithm for k-nearest neighbor search [Korn et al. 1996], authors present a novel multi-step algorithm which is guaranteed to produce the minimum number of candidates. In [19] authors formally define spatial preference queries and propose appropriate indexing techniques and search algorithms for them.

In [20]authors propose techniques that solve the problem by performing a single query for the whole input segment. As a result the cost, depending on the query and dataset characteristics, may drop by orders of magnitude. In addition, we propose analytical models for the expected size of the output, as well as, the cost of query processing, and extend out techniques to several variations of the problem. In [21] the problem assumes that the query point is not static, as in *k*-nearest neighbor problem, but varies its position over time. In this paper, four different methods are proposed for solving the problem. Discussion about the parameters affecting the performance of the algorithms is also presented. A sequence of experiments with both synthetic and real point data sets are studied. In the experiments, our algorithms always outperform the existing ones by fetching 70% less disk pages. In some settings, the saving can be as much as one order of magnitude.

Given a set of spatial points DS, each of which is associated with categorical information, e.g., restaurant, pub, etc., the optimal route query finds the shortest path that starts from the query point (e.g., a home or hotel), and covers a user-specified set of categories (e.g., {pub, restaurant, museum}). The user may also specify partial order constraints between different categories, e.g., a restaurant must be visited before a pub. Previous work has focused on a special case where the query contains the total order of all categories to be visited (e.g., museum! restaurant ! pub). For the general scenario without such a total order, the only known solution reduces the problem to multiple, total-order optimal route queries. As we show in this paper, this naive approach incurs a significant amount of repeated computations, and, thus, is not scalable to large data sets. Motivated by this, we propose novel solutions to the general optimal route query, based on two different methodologies, namely backward search and forward search. In addition, we discuss how the proposed methods can be adapted to answer a variant of the optimal route queries, in which the route only needs to cover a subset of the given categories. Extensive experiments, using both real and synthetic data sets, confirm that the proposed solutions are efficient and practical, and outperform existing methods by large margins.

In section 2, algorithms related to the present work are given. . The problem to be solved is defined in section 3. The section 4 gives the existing system. The proposed system is given in section 5. The algorithm proposed is given in section 6. Data interpretation is given in section 7. The experimental results are given in section 8. Performance metric of the work of this paper is highlighted in section 9. Conclusion and future work are presented in section 10.

2 RELATED ALGORITHMS

2.1 Forward Search Algorithm

Distance Calculation In this module the algorithm traces the location of each point and computes a route by repeatedly connecting the current location to the nearest point. Similarly the routes are calculated for all the points and its locations are extracted. Based on the location latitude and longitude position the distance is calculated.

2.2 Backward Search Algorithm

Distance Calculation In this module the algorithm traces the location of each point and computes a route by repeatedly connecting the last location to the nearest point. Similarly the routes are calculated for all the points and its locations are extracted in reverse manner. i.e. we calculate the distance between the point from destination to source points.

2.3 Batch Backward Search (BBS)

The batch backward search method, improves SBS by employing batch processing in the backward join operations. Specifically, both the candidate set are partitioned into clusters before participating in a backward join . The partitioning of first groups points by their category, and then for each group, the points are further partitioned into clusters based on their spatial proximity. The partitioning of route set Ri follows a similar strategy, by first grouping routes based on the categories they cover, and then clustering each group according to the locations of their start points. The clustering module in BBS must be highly efficient, since it is called during query time.

2.4 Simple Forward Search

A native implementation of forward search clearly takes time exponential to the number of points in the data set. SFS avoids this by utilizing the suffix optimality property stated in Lemma 1, and incrementally building the optimal suffix table (denoted by _), as in the backward search methods. SFS differs from the backward search solutions in that it fills _ in a different order. Specifically, in SFS all cells of _ are initialized to a special token Unknown , which indicates that the corresponding optimal suffix has not been computed yet. Then, whenever SFS backtracks from a point p, the algorithm guarantees that it has obtained the optimal suffix that starts at p, and covers the set of categories V that are not visited by the current path from the query point q to p.1 Thus, SFS stores this optimal suffix in the corresponding cell in _ (denoted by _p;V), replacing the Unknown token .

3 PROBLEM DEFINITION

This naive approach incurs a significant amount of repeated computations, and, thus, is not scalable to large data sets. Motivated by this, we propose novel solutions to the general optimal route query, based on two different methodologies, namely backward search and forward search. In addition, we discuss how the proposed methods can be adapted to answer a variant of the optimal route queries, in which the route only needs to cover a subset of the categories. The solution BFS that combines merits from both backward and forward search achieves the best performance.

4 EXISTING SYSTEM

In our Existing work has focused on a special case where the query contains the total order of all categories to be visited (e.g., museum! restaurant! pub). For the general scenario without such a total order, the only known solution reduces the problem to multiple, total-order optimal route queries. Existing work on optimal route computation focuses on greedy solutions. Existing solutions are either limited to a specific setting of the problem, or incur expensive, redundant computations.

4.1 Disadvantages

1. Could not find efficient shortest path

2 .Redundant computations

3.Time Complexity is high.

5 PROPOSED APPROACH

The proposed method implies a naïve approach to find the optimal route queries using two methodologies called Backward search and Forward Search. The backward search methodology computes the optimal routes in reverse order of its points. The forward search approach traverses the search space in a depth-first manner, and incrementally improves the bound for optimal route length. Forward search methods report results progressively, i.e., they first quickly produce one solution to the query, and then incrementally update it, until reaching the optimal one or being terminated by the user.

5.1 Advantages

- Increases performance level.
- Avoids redundant computations
- Efficient and robust solution.
- Returns optimal route and guarantees the quality of result.

5.2 **Sequence Diagram:**

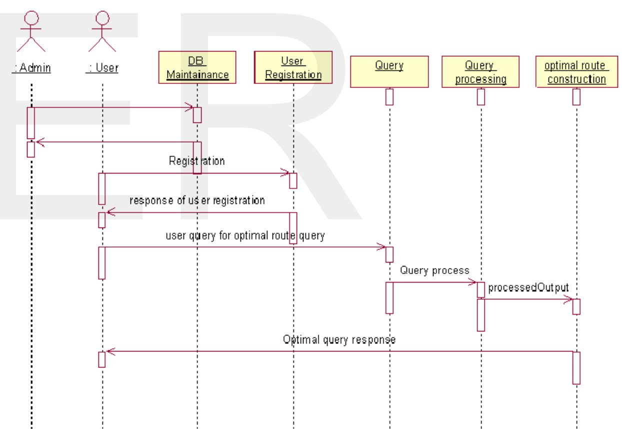

Fig: 1 Sequence Diagram

5.3 **I NPUT**

The user have to get registered into the database if he/she is a new user, else the user can log in to USER SEARCH QUERY where starting time, starting point and places to visit should be given.

5.4 **OUTPUT**

The optimal route between the categorical information along with the working hours will be displayed. For e.g if the starting point is 9:00, starting place is t-nagar, the query is to visit hotel, mall,temple and click Submit, the output will be as given.

6 PROPOSED ALGORITHM

FORWARDBACKWARD ALGORITHM

The forward–backwardalgorithm is an inference algorithm for hiddenMarkov models which computes the posterior marginal's of all hidden state variables given a sequence of observations/emissions $o_{1:t} := o_1, \dots, o_t$, i.e. it computes, for allhidden state variables $X_k \in \{X_1, \dots, X_t\}$, the distribution $P(X_k \mid o_{1:t})$. This inference task is usually called *smoothing*. The algorithm makes use of the principle of dynamic programming to compute efficiently the values that are required to obtain the posterior marginal distributions in two passes. The first pass goes forward in time while the second goes backward in time; hence the name *forward–backward algorithm*.

The term *forward–backward algorithm* is also used to refer to any algorithm belonging to the general class of algorithms that operate on sequence models in a forward–backward manner. In this sense, the descriptions in the remainder of this article refer but to one specific instance of this class.

The forward–backward algorithm computes a set of forward probabilities which provide, for all $k \in \{1, \dots, t\}$, the probability of ending up in any particular state given the first k observations in the sequence, i.e. $P(X_k \mid o_{1:k})$. In the second pass, the algorithm computes a set of backward probabilities which provide the probability of observing the remaining observations given any starting point k, i.e. $P(o_{k+1:t} \mid X_k)$. These two sets of probability distributions can then be combined to obtain the distribution over states at any specific point in time given the entire observation sequence:

$$P(X_k \mid o_{1:t}) = P(X_k \mid o_{1:k}, o_{k+1:t}) \propto P(o_{k+1:t} \mid X_k) P(X_k \mid o_{1:k})$$

The last step follows from an application of Bayes' rule and the conditional independence of $o_{k+1:t}$ and $o_{1:k}$ given X_k.

As outlined above, the algorithm involves three steps:

1. Computing forward probability
2. Computing backward probabilities
3. Computing smoothed values.

The forward and backward steps may also be called "forward message pass" and "backward message pass" - these terms are due to the message-passing used in general propagation approaches. At each single observation in the sequence, probabilities to be used for calculations at the next observation are computed. The smoothing step can be calculated simultaneously during the backward pass. This step allows the algorithm to take into account any past observations of output for computing more accurate results.

7 DATA INTERPRETATION

The user may also specify partial order constraints between different categories, e.g., a restaurant must be visited before a pub. Previous work has focused on a special case where the query contains the total order of all categories to be visited (e.g., museum! restaurant ! pub).

For the general scenario without such a total order, the only known solution reduces the problem to multiple, total-order optimal route queries. As we show in this paper, this naive approach incurs a significant amount of repeated computations, and, thus, is not scalable to large data sets.

Motivated by this, we propose novel solutions to the general optimal route query, based on two different methodologies, namely backward search and forward search. In addition, we discuss how the proposed methods can be adapted to answer a variant of the optimal route queries, in which the route only needs to cover a subset of the given categories.

8 EXPERIMENTAL RESULT

This paper is specialized to get the Optimal route to reach three different destinations along with Operational Hours. The starting time and the starting location and the different places to be visited is compulsory. There will be possible Optimal Routes along with Operational Hours of those destinations. If all the destinations cannot be covered within the Operational Hours it will be displayed as Optimal Route failed.

In the base paper the starting point timing and the places to visit along with starting place is given. Therefore the shortest path to reach the different places is taken care. Whereas in my proposed work, the Operational Hours are taken care for each destination places along with starting time, starting location and different places to visit.

In comparision with the base paper, the proposed work is added with

- Depth First Search Algorithm (Forward Search)
- Forward and Backward Search
- Operational Hours

In brief to the proposed work, so many of the researchers have explained many algorithms used to obtain an optimal route to reach different destinations like some have worked for one, some have worked for two and even some have worked for 3 different destinations. But my work is enhanced by obtaining Optimal Route for three different destinations along with their Operational Hours using Backward Forward Search algorithm.

9 PERFORMANCE METRIC

The performance of the proposed work is better than the base paper by getting the Operational Hours of the different destinations to be visited along with starting point time, starting point location and places to visit .It is shown in figures 2-3 and the results are presented in Table 1.

BFS – Optimal Route

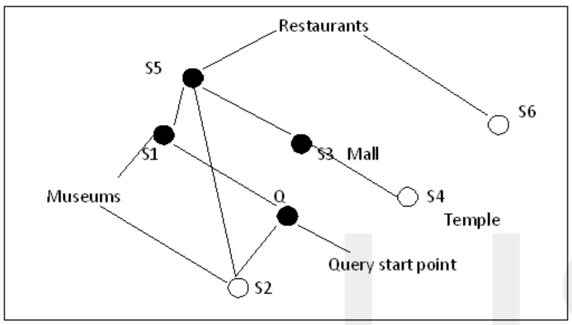

Fig: 2 Example of Optimal Route

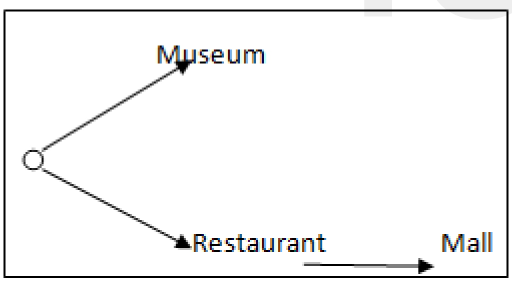

Fig: 3 Visit Order Graph

TABLE: 1

Suffix Length	Start Point	Optimal Suffix	Operational Hours
	S1	S1→S5→S3	10:30-22:30 ,10:30-22:30 ,9:30-22:30
3	S2	S2→S5→S3	9:00-18:00,10:30-20:00 ,9:30-22:30

Optimal Suffix Table used in BFS

10 CONCLUSION

This paper investigates the problem of optimal route query processing. Existing solutions are either limited to a specific setting of the problem, or incur expensive, redundant computations. Hence, we propose novel and efficient solutions, based on two methodologies: backward and forward search. The solution BFS that combines merits from both backward and forward search achieves the best performance. In thefuture, we plan to study alternative definitions of the optimal route query, that have temporal constraints (e.g., have lunch at a specified period) or maximize the number of categories to be visited given a total travel length budget.

ACKNOWLEDGMENT

The author wishes to thank Vel's University for the opportunity provided to do research. The author also thanks Mrs.Poonguzhali for her support and guidance inspiring to publish journals and all the professors of Department of Computer Science and Technology, Vel's University for their encouragement.

Dr.K.Selvakumar wish to acknowledge the financial support from Anna University, Chennai-600 025, India project for young faculty members under research support scheme.

REFERENCES

[1] D. Zhang, Y.M. Chee, A.M.A. Tung, and M. Kitsuregawa, "Keyword Search in Spatial Databases: Towards Searching by Document," Proc. IEEE Int'l Conf. Data Eng. (ICDE), 2009.

[2] D. Wu, M.L. Yiu, C.S. Jensen, and G. Cong, "Efficient Continuously Moving Top-K Spatial Keyword Query Processing," Proc. IEEE 27th Int'l Conf. Data Eng. (ICDE), 2011.

[3] D. Papadias, Q. Shen, Y. Tao, and K. Mouratidis. "Group Nearest Neighbor Queries". In ICDE 2004, 20th International Conference on Data Engineering March 30 - April 2, 2004.

[4] D. Wagner and T. Willhalm."Geometric speed-up techniques for finding shortest paths in large sparse graphs". In 11th European Symposium on algorithms,volume 2832 of LNCS, pages 776–787. Springer, 2003.

[5] Gopika N.A1, S. Deepa Kanmani2 " A Survey On Optimal Route Queries for Road Networks" .

[6] Gholamhosein Sheikholeslami Surojit Chatterjee" WaveCluster: A Multi-Resolution Clustering Approach for Very Large Spatial Databases"

[7] Holger Bast† Stefan Funke† Domagoj Matijevic† Peter Sanders‡Dominik Schultes‡ "In Transit to Constant Time Shortest-Path Queries in Road Networks".

[8] H. Ferhatosmanoglu, I. Stanoi, D. Agrawal, and A.E. Abbadi. "Constrained nearest neighbor queries". In *SSTD*, pages 257–278, 2001.

[9] I. De Felipe, V. Hristidis, and N. Rishe, “Keyword Search on Spatial Databases,” Proc. IEEE 24th Int’l Conf. Data Eng. (ICDE),2008.

[10] I. T. Hern´adv¨olgyi. “Solving the sequential ordering problem with automatically generated lower bounds”. In Operations Research Proceedings 2003, pages 355–362. Springer Verlag, September 3-5, 2003.

[11] K. M¨uller.” Design and implementation of an efficient hierarchical speed-up technique for computation of exact shortest paths in graphs. Master’s thesis, Universt¨at Karlsruhe, 2006, supervised by D. Delling, M.Holzer, F. Schulz, and D. Wagner.

[12] Li J, Yang Y.D, Mamoulis N(2013), “Optimal Route Queries with Arbitrary Order Constraints”, IEEE Trans.Computers, vol. 25, no. 5, pp. 1097- 1110.

[13] Mehdi Sharifzadeh, Mohammad Kolahdouzan, Cyrus Shahabi, “The Optimal Sequenced Route Query”.

[14] Nanopoulos A, Bozanis P(2003), “Categorical Range Queries in Large Databases”, Springer-Verlag Berlin Heidelberg, pp. 122-139.

[15] N. Roussopoulos, S. Kelley, and F. Vincent. “Nearest Neighbor Queries”. In Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data, San Jose, California, May 22- 25, 1995, pages 71–79.

[16] Papadias D,Zhang J, Mamoulis N, Tao Y(2003), “Query Processing in Spatial Network Databases”, VLDB conference.

[17] S.Karthik, “ Optimal Route Query Based On Forward Search And Backward Search”

[18] T. Seidl and H.-P. Kriegel. “Optimal Multi-Step K- Nearest Neighbor Search”. In SIGMOD 1998, Proceedings ACM SIGMOD International Conference on Management of Data, June 2-4, 1998, Seattle, Washington, USA, pages 154–165. ACM Press, 1998.

[19] Yiu M.L, Mamoulis N, Vaitis M(2007), “Top-k Spatial Preference Queries”, 7160/05E from Hong Kong RGC Miguel Gomes da Costa Júnior and Higuo.

[20] Y. Tao, D. Papadias, and Q. Shen. “Continuous Nearest Neighbor Search”. In *VLDB 2002,* Proceedings of 28th international Conference on Very Large Data Bases, August 20-23, 2002.

[21] Z. Song and N. Roussopoulos. “K-Nearest Neighbor Search for Moving Query Point”. In The Seventh International Symposium on Spatial and emporal Databases,SSTD’2001.

Mrs.S. Bhuvaneswari received the M.C.A. degree from the IGNOU University, Tamil Nadu, India, in 2007 and is currently pursuing the M.Phil degree in Computer Science at Vels University, Tamil Nadu, India. Her research interests covers Data Mining .

Mrs.T.S.Poonguzhali is working as Assistant Professor at Vels University, Tamil Nadu, India.. She has 8 years of teaching experience. Her area of interest involves Data Mining, Image Processing and Computer Applications.

Dr.K.Selvakumar received his M.E. Computer Science and Engineering from Anna University, India, Ph.D.and PostDoctorate from Bharathidasan University, India in 1993 and 1994 respectively. He is an aassistant professor at University VOC College of Engineering, Anna University (Tutucorin Campus). His research interests covers cloud computing, networking, image processing, numerical analysis, singular perturbation problems in control system and stiff.Computations. He is doing research for the last 27 years and publishing research papers regularly.