Analyze and Update Data Method for Skyline Query Processing Against Distributed Data Sites

[1]Nagalakshmi. Panchakatla, [2]D. Umadevi, [3]Shaheda Akthar
[1,2,3]Dept. of CSE, Sri Mittapalli College of Engineering, Guntur, AP, India

Abstract
The skyline of a multidimensional point set is a subset of interesting points that are not dominated by others. In this paper, We are introducing the Analyze and Update data Method which enhances the constrained Skyline query processing against distributed data sites approach and user centric approach. User can easly compares the result data dynamically. we follow constrained skyline queries in a large-scale unstructured distributed environment, where relevant data are distributed among geographically scattered sites.A partition algorithm that divides all data sites into incomparable groups such that the skyline computations in all groups can be parallelized without changing the final result, then develop a novel algorithm framework called PaDSkyline for parallel skyline query processing among partitioned site groups. The method also employ intragroup optimization and multifiltering technique to improve the skyline query processes within each group. In particular, multiple (local) skyline points are sent together with the query as filtering points, which help identify unqualified local skyline points early on a data site. In this way, the amount of data to be transmitted via network connections is reduced, and thus, the overall query response time is shortened further. Cost models and heuristics are proposed to guide the selection of a given number of filtering points from a superset. A cost efficient model is developed to determine how many filtering points to use for a particular data site. The results of an extensive experimental study demonstrate that our proposals are effective and efficient.

Keywords
Constrained Skyline Query, Filtering Point, Distributed Query Processing, Query Optimization and Generalization.

I. Introduction

GIVEN a multidimensional point set, a skyline query [1], returns all interesting points that are not dominated by any other points. A point pt1 is said to dominate pt2 if pt1 is not worse than pt2 in every single dimension but better than pt2 in at least one dimension. The implication of "better" varies in different contexts. For example, "better" can mean "smaller" or "larger" in value comparison, and "earlier" or "later" in date comparison. Because of their powerful capability of retrieving interesting points from a large multidimensional data set, skyline queries are well suitable for applications like multiple criteria optimization. Skyline queries play an important role in multi criteria decision making and user preference applications. For instance, a tourist can issue a skyline query on a hotel relation to get those hotels with high stars and cheap prices. Most works on skyline queries [1-7], so far have assumed a centralized data storage, and been focused on providing efficient skyline computation algorithms on a sole database. This assumption, however, fails to reflect the distributed computing environments consisting of different computers, which are located at geographically scattered sites and connected via Internet. For example, a stock trader needs to know which stocks worldwide are worth investing, based on the trading records of the previous day. For this purpose, she needs to access multiple stock information databases available at different places like New York Stock Exchange, London Stock Exchange, Tokyo Stock Exchange, etc. For each single stock, the agent needs to take into consideration multiple attributes like last trade price, change, last close price, estimated price, volume, etc. Therefore, a skyline query against those distributed databases will help the agent get those interesting stocks. Another example is online comparative shopping. In such example cases, however, directly applying existing centralized skyline approaches to process distributed skyline queries would incur large overheads. Such approaches assume a sole relation as input, and lack adaptations or optimizations specific to distributed computing environments. In this work, we intend to efficiently process constrained skyline queries [6], in such widely distributed environments and we can also analyze and update data mechanism to get quality results. Given a distributed environment without any overlay structures, our objective is efficient query processing strategies that shorten the overall query response time and analyze and dynamically update the resultant data. We first speed up the overall query processing by achieving parallelism of distributed query execution. Given a skyline query with constraints, all relevant sites are partitioned into incomparable groups among which the query can be executed in parallel. The parallel execution also makes it possible to report skyline points progressively, which is usually desirable to users. Within each group, specific plans are proposed to further improve the query processing involving all intragroup sites. On a processing site, multiple filtering points are deliberately picked based on their overall dominating potential from the local skyline. They are then sent to other sites with the query request, where they help identify more unqualified points that would otherwise be reported as false positives, and thus, reducing the communication cost between data sites.

We make the following contributions in this paper:

1. We follow a specific partition algorithm that divides all relevant sites into groups such that a given query can be executed in parallel among all those site groups. It elaborate on the intragroup query execution strategies,then give a parallel distributed skyline algorithm, together with a cost model to estimate the overall query response time.
2. We follow detail heuristics for selecting a given number of multiple filtering points in distributed query processing such that the amount of data to be transmitted via the network is reduced.
3. We enhancing a cost-efficient model for dynamically determining the number of filtering points to be sent to a particular site such that the benefit of using filtering points is maximized.
4. We introducing anlyze and update data results dynamically methods for user centric approach.
5. We conduct an extensive experimental study on both synthetic and real data sets, and the results demonstrate the effectiveness, efficiency, and robustness of our proposals.

II. Related Work

Borzonyi et al. [1], introduced the skyline operator into database systems with algorithms Block Nested Loop (BNL) and Divide-and-Conquer (D&C). Chomicki et al. [2], proposed a Sort-Filter-

Skyline (SFS) algorithm as a variant of BNL. Tan et al. [7] proposed two progressive algorithms: Bitmap and Index. The former represents points in bit vectors and employs bitwise operations, while the latter utilizes data transformation and Bþ-tree indexing. Kossmann et al. [5] proposed a Nearest Neighbor (NN) method. It identifies skyline points by recursively invoking R_-tree-based depthfirst NN search over different data portions. Papadias et al. [6] proposed a Branch-and-Bound Skyline (BBS) method based on the best-first nearest neighbor algorithm [8]. Godfrey et al. [4] provided a comprehensive analysis of previous skyline algorithms without indexing supports, and proposed a new hybrid method with improvement. All these works assume centralized data storage. Deviating from skyline queries in the centralized setting, Balke et al. [9], addressed skyline operation over web databases, where different dimensions are stored in different data sites. Their algorithm first retrieves values in every dimension from remote data sites using sorted access in round-robin on all dimensions. This continues until all dimension values of an object, called the terminating object, have been retrieved. Then all nonskyline objects will be filtered from all those objects with at least one dimension value retrieved. Differently in this paper, our work deals with distribution of data horizontally partitioned. Wuet al. [10] proposed a parallel execution of constrained skyline queries in a CAN [11], based distributed environment. By using the query range to recursively partition the data region on every data site involved, and encoding each involved (sub)region dynamically, their method avoids accessing sites not containing potential skyline points and progressively reports correct skyline points. Wang et al. [12], developed Skyline Space Partitioning (SSP) approach to compute skylines on a tree-structured P2P platform BATON. SSP partitions the skyline space into regions and maps them in a single-dimensional order, which allows regions to be distributed to different peer nodes according to BATON protocols. Our proposal in this paper differs from these two pieces of work in that we do not assume any overlay availability on top of the original network. Huang et al. [13], proposed techniques for skyline query processing in MANETs. Lightweight devices in MANETs are able to issue spatially constrained skyline queries that involve data stored on many mobile devices. Queries are forwarded through the whole MANET without routing information. They proposed a filtering-based data reduction technique that reduces the data transferred among devices.

Our work, assuming a wired large-scale distributed environment, is also different from this work in a MANET setting. Zhu et al. [21] proposed a feedback-based distributed skyline (FDS) algorithm within a network setting similar as ours. However, FDS only deals with skyline queries without any constraints that are considered in our work. Also, FDS is focused on minimizing the network bandwidth use, and therefore, it uses a multiple-round feedback mechanism to prune the unqualified skyline candidates. When the network becomes large, the delay (i.e., the query response time) of FDS will increase considerably. In contrast, our work applies a one-round filtering technique that is aimed to reduce the network delay.

A. Skyline Processing Against Distributed Data Sites

In a previous work [13], a skyline query is forwarded among mobile peers via multiple hops in a MANET. Whereas in a wired environment, connections are end-to-end. Because of such wired connections between each pair of sites, a constrained skyline query can be sent out to all peer sites and then each site can execute the query on its own data set simultaneously. This naive execution plan might benefit from the parallelism among all peer sites Following the data reduction principle in semijoin in

distributed query processing [14], Huang et al. [13], proposed to transfer a single local skyline point with the query request among mobile peers, which acts as a filter to identify those unqualified ones in peers' local skylines. this single filtering point to multiple ones since wired connections are much faster and more reliable than wireless channels. How to choose multiple filtering points will be detailed in Section IV. In this section, we focus on finding a good distributed skyline query execution plan that minimizes the total execution time. the order to execute a distributed skyline query among multiple sites really matters because an appropriate execution order can filter more data points earlier, and thus, reducing not only communication cost but also the local processing cost in the subsequent execution. To balance the parallelism and filtering when processing a Distributed Skyline Query (DSQ) and get short overall response time.

The Parallel Distributed Query Execution and Parallel Distributed Skyline Algorithm are already discussed. We followe the method of data reduction using multiple filtering points. One single filtering point is transferred and changed from peer to peer in [13]. During the query forwarding and processing, the single filtering point is dynamically changed once another point is found to be more powerful in filtering out unqualified candidates. Data sites in this work, in contrast, are wired with considerably steady and high bandwidth compared to wireless MANETs. This allows us to use multiple filtering points among sites, as data transmission cost via wired connections is lower and multiple filtering points are expected to have higher filtering power. Consequently, we need to decide which and how many skyline points should be used as filtering points in the distributed skyline query processing such that tum bin the benefit is maximized. The previous method formalize the dominating region of multiple skyline points with respect to constraints, and address how to select a given number of filtering points initially. We focus on study how many filtering points are adequate for a particular data site.

III. Proposed Model

A. Analyze and Update Data Dynamically Method

The proposed method is completely based on analysis and updating of data over parallel distributed query execution where the user can generalize the data dynamically which leads to end user centric results where we can clearly compare results in different aspects. The proposed method follows cost model of dynamic filter points for estimate the analysis and updating of data.

B. Cost-Efficient use of Dynamic Filtering Points

The number of filtering points to K and use it in a system-wide way. In practice, data space and local skyline cardinality can be different from site to site. A fixed number of filtering points for all data sites can cause some problems. For some individual data sites, the number of filtering points can be too large and become overkill. This unfortunately incurs unnecessary data transmission cost because a number of unused filtering points are transferred via the network. some other sites that have large data spaces and large local skylines may need more filtering points to achieve good filtering effect. As a result, using a fixed number of filtering points on all data sites may be too rigid to shorten the overall query response time. A cost-efficient model to estimate how many filtering points are adequate for an individual site in terms of filtering power. Based on this model, we accordingly vary the

number of filtering points for each site in the process of distributed skyline computation.

C. Enhanced Cost Model

Principally, what we need to do is to ensure that more local skyline points are filtered by K filtering points such that sending all these K filtering points does pay off. In order to evaluate the cost-efficiency of given K filtering points, we first define the concept of filtering ratio as follows:

Definition 2. Given K filtering points, the filtering ratio on site Si, termed as Fri(KÞ), is the percentage of local skyline points in Skyi that are filtered by K filtering points.

Fri (K) = l FltedSki l / l Ski l (1)

Here, l Ski l is the cardinality of local skyline on Site Si, and = l FltedSki l is the number of local skyline points filtered by those K filtering points. With the definition of filtering ratio, we are able to measure the benefit of the K filtering points on a particular site according to the following formula:

Ei (K) = Fri(K). l Ski l – K (2)

To make the K filtering points cost-efficient for a particular data site, we need to ensure Ei (K)> 0 As a matter of fact, the larger Ei (K) is, the more benefit we obtain from sending the K filtering points. With this formula, we can find a minimum K that maximizes the benefit of K filtering points for a particular site Si. For this purpose, we need to predict l Ski l ,Fri (K) in advance for a particular site Si. Assuming that each site provides data cardinality and distribution, in addition to its MBR, we can estimate the local skyline size jSkyij on Si according to [15-16, 18, 20]. The relevant result in [16], is used in the experiments. Dynamic Update of Multiple Filtering Points After a site Si receives a query request with a set of filtering points Fflt, it will execute a local query processing. To take advantage of the filtering points, the local processing can be implemented in two ways. In an integrated way, Fflt is checked against every candidate point pt met during the skyline computation, any dominated pt is ignored, and any dominated si in Fflt is removed from the set. In a separate way, a local skyline is computed first, and then it will be compared with Fflt to filter out those unqualified candidates and dominated sis. The integrated way is seamlessly applicable to those centralized skyline algorithms that are not based on data transformation [1-2, 4-6], whereas the separate way is applicable to all existing skyline algorithms. A simple yet efficient way is to treat all points in these two sets equally and select K filtering points from scratch. Alternatively, we can decide the most appropriate K value and pick K ones from Ski U F"flt. For this purpose, the cost model proposed. can be used for a further site Si to which the query is sent further. We use this method in multilevel for analysis purpose ,we use update strategy method for automatic updating for the dynamic results to get best result.

IV. Experimental Study

In this section, we evaluate our distributed skyline query mechanism with extensive experiments. All the simulation experiments are conducted on a Linux Server with two Intel(R) Xeon(TM) 2.80 GHz processors and 1.0 GB RAM. We use two kinds of synthetic data sets, i.e., independent and anti correlated data sets, and a real-life data set of NBA players' statistics (http://databasebasketball.com), which contains 16,644 records of 17 attributes and approximates a correlated data distribution. In our preliminary work [3], we have studied the performance of algorithm PaDSkyline by comparing it with Naive and Random approaches. The experimental results demonstrate the superiority of PaDSkyline. Therefore, we do not include the results from the previous work. In this section, we aim at investigating the technical extensions proposed and evaluating their performance within our PaDSkyline framework with Analyze and Update scheme.

The parameters of the experiments are listed in Table 1, and the default parameter values are given in bold. Same as the results presented in previous experimental study [3], the proposed methods yield similar performance tendency on different data sets. We only detail the results on independent data set for ease of presentation in this paper.

Table 1: Parameters Used in Experiments

Parameter	Values
Dimensionality	2, **3**, 4, 5
Number of sites	**1,000**, 2, 000, ..., 10, 000
Filtering point percentage	10%, 20%, ..., **50%** ..., 90%
Distance threshold ratio δ	10%, 20%, **30%**, ..., 90%
Cardinality of each site	**1,000**
Number of queries	**50**

We consider the following four performance metrics:

Query Traffic, which counts the number of data points sent in the network during the query processing. Data Reduction Efficiency, the efficiency of multiple filtering points in local skyline computation in terms of the data reduction rate DRR [13]. The DRR is the proportion of data points reduced by the filtering points to the points in the unreduced local skyline. Where Ki is the number of filtering points sent to a processing site, and m is the network size.

Response Time, which records the overall query processing time, from the moment when a query is issued to the moment when the final result is obtained. Precision, which indicates how much data returned to the query originator are really useful in the final result.

A. Comparison of Different Methods

In this section, we demonstrate the results of different methods for performance study, i.e., two basic methods MaxSum and MaxDist, their improved versions IpvMax-Sum and ipvMaxDist, and Predict which can automatically select filtering points according to the cost-efficient model in[a]. For IpvMaxSum and ipvMaxDist,we set the initial distance threshold as 30 percent according to our previous experiment, which yields near-optimal performance.

B. Performance on Query Traffic

We first evaluate the performance of various methods on Query Traffic against two important factors, i.e., network size and dimensionality. In fig 2, it is clear that Predict algorithm incurs much less query traffic than other algorithms, due to its dynamical filtering points selection strategy. The Predict method can automatically select "optimal" number of filtering points using the proposed cost model. In our experiments, the "optimal" percentage of filtering points varies from 10 to 60 percent on different sites. We can also see that the improved version can yield better performance, e.g., IpvMaxSum is more efficient than MaxSum, due to the usage of distance threshold _, which makes its filtering points more powerful in terms of filtering capability.

C. Performance on Data Reduction Rate

In this experiment, we measure the performance on data reduction rate (DRR). The higher DRR of an algorithm has, the more unqualified points it filters out. Fig. 3, shows that Predict algorithm

again outperforms other algorithms. In Predict, every site utilizes the cost model to select the right filtering points for their next sites, which is expected to cover the next site's data space quite well in an approximate way. This can produce a tailor-made filter set for a specific site with maximized filtering capability, compared with the other methods with preassigned number of filtering points. The sites generally have different data distributions, and therefore, a fixed number cannot be optimal for all the sites in the distributed environment.

D. Performance on Response Time

We further study the response time of our five algorithms. The response time is calculated at the moment when the query is issued till the final results are retrieved. It includes query propagating time, skyline computing time, filtering point selection time, and result transferring time, etc. Fig. 4, shows that the performance differences among these algorithms are not significant as that on Query Traffic because the costs of query propagation and local skyline computation are similar for different approaches, and the cost variations of filtering point selection and intermediate data transfer are the main reason of performance difference. The performance of Predict algorithm remains the best because of its effective communication cost reduction, although its automatic filtering point generation may incur more computational cost. However, such local computational cost overhead on sites is marginal compared with data transmission cost in the network. The IpvMaxSum and IpvMaxDist are comparable with the original ones, as the reduced data transmission cost can well offset the additional distance evaluating processes on filtering point selection.

E. Performance on Average Precision

In the last experiment, we study the performance of average precision of the returned results to the query issuers. Fig. 5, shows that the Predict algorithm performs the best among all competitors, due to its powerful filtering capability and high data reduction rate, which prevents more unqualified points from being transferred to the query issuer. The gap here is not significant as the result on DRR because the metric Precision only counts the effectiveness of final results rather than the intermediate results transferred among sites. The improved versions yield better performance than original MaxDist and MaxSum, respectively. Although they deploy the same number of filtering points for skyline processing, the filtering points are more effective with the enhancement of distance threshold.

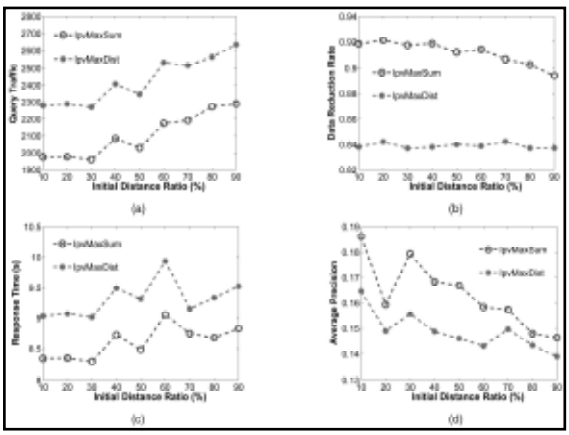

Fig. 1: The Effect of Initial Distance Threshold. (a) Query Traffic. (b) Data Reduction Rate. (c) Response Time. (d) Precision

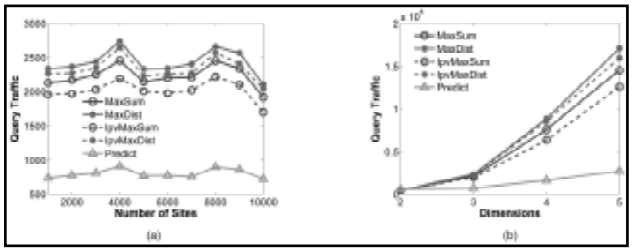

Fig. 2: Performance on Query Traffic. (a) Network Size. (b) Dimensionality

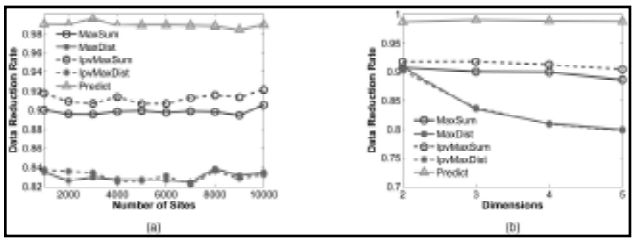

Fig. 3: Performance on Data Reduction Rate. (a) Network Size. (b) Dimensionality

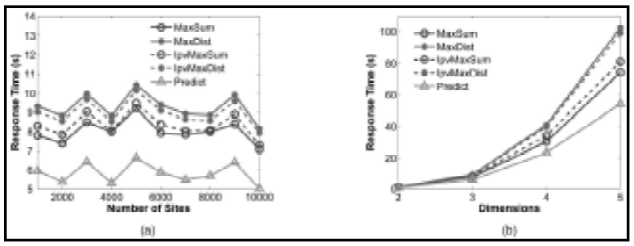

Fig. 4: Performance on Response Time. (a) Network Size. (b) Dimensionality

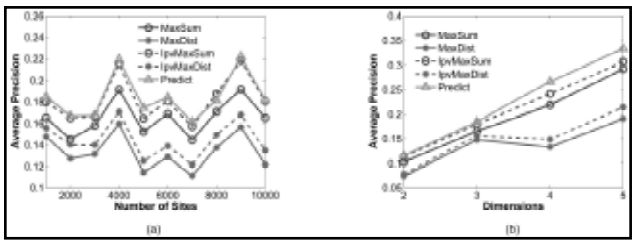

Fig 5. Performance on Average Precision. (a) Network Size. (b) Dimensionality

V. Conclusions and Futrue Work

In this paper, we have addressed the problem of constrained skyline query processing against distributed data sites and Analyzed way of viewing results and updating the results dynamically to get best result.. To accelerate the query processing, we partition all relevant sites into incomparable groups and parallelize the query processing among all groups. We select local skyline points and send them as filtering points together with the query to relevant data sites, in order to prevent more data from being transmitted through the network. Furthermore, a dynamic filtering points selection strategy is proposed based on a novel cost-efficient model. Extensive experimental results demonstrate the efficiency and effectiveness of our proposals in a distributed network environment

References

[1] S. Borzonyi, D. Kossmann, K. Stocker,"The Skyline Operator", Proc. Int'l Conf. Data Eng. (ICDE), pp. 421-430, 2001.

[2] J. Chomicki, P. Godfrey, J. Gryz, D. Liang, "Skyline with Presorting", Proc. Int'l Conf. Data Eng. (ICDE), pp. 717-816, 2003.

[3] B. Cui, H. Lu, Q. Xu, L. Chen, Y. Dai, Y. Zhou,"Parallel Distributed Processing of Constrained Skyline Queries by Filtering", Proc. Int'l Conf. Data Eng. (ICDE), 2008.
[4] P. Godfrey, R. Shipley, J. Gryz,"Maximal Vector Computation in Large Data Sets", Proc. Int'l Conf. Very Large Data Bases (VLDB), pp. 229-240, 2005.
[5] D. Kossmann, F. Ramsak, S. Rost,"Shooting Stars in the Sky: An Online Algorithm for Skyline Queries", Proc. Int'l Conf. Very Large Data Bases (VLDB), pp. 275-286, 2002.
[6] D. Papadias, Y. Tao, G. Fu, B. Seeger,"An Optimal and Progressive Algorithm for Skyline Queries", Proc. ACM SIGMOD, pp. 467-478, 2003.
[7] K.-L. Tan, P.-K. Eng, B.C. Ooi,"Efficient Progressive Skyline Computation", Proc. Int'l Conf. Very Large Data Bases (VLDB), pp. 301-310, 2001.
[8] G. Hjaltason, H. Samet,"Distance Browsing in Spatial Database", ACM Trans. Database Systems (TODS), Vol. 24, No. 2, pp. 265-318, 1999.
[9] W.-T. Balke, U. Gu¨ ntzer, J.X. Zheng,"Efficient Distributed Skylining for Web Information Systems", Proc. Int'l Conf. Extending Database Technology (EDBT), pp. 256-273, 2004.
[10] P. Wu, C. Zhang, Y. Feng, B.Y. Zhao, D. Agrawal, A.E. Abbadi,"Parallelizing Skyline Queries for Scalable Distribution", Proc. Int'l Conf. Extending Database Technology (EDBT), pp. 112-130, 2006.
[11] S. Ratnasamy, P. Francis, M. Handley, R. Karp, S. Shenker,"A Scalable Content-Addressable Network", Proc. ACM SIGCOMM, pp. 161-172, 2001.
[12] S. Wang, B.C. Ooi, A.K.H. Tung, L. Xu,"Efficient Skyline Query Processing on Peer-to-Peer Networks", Proc. IEEE Int'l Conf. Data Eng. (ICDE), pp. 1126-1135, 2007.
[13] Z. Huang, C.S. Jensen, H. Lu, B.C. Ooi,"Skyline Queries against Mobile Lightweight Devices in MANETs", Proc. Int'l Conf. Data Eng. (ICDE), pp. 66, 2006.
[14] P.A. Bernstein, N. Goodman, E. Wong, C.L. Reeve, J. James, B. Rothnie,"Query Processing in a System for Distributed Databases (SDD-1)", ACM Trans. Database Systems (TODS), Vol. 6, No. 4, pp. 602-625, 1981.
[15] S. Chaudhuri, N.N. Dalvi, R. Kaushik,"Robust Cardinality and Cost Estimation for Skyline Operator", Proc. Int'l Conf. Data Eng. (ICDE), p. 64, 2006.
[16] P. Godfrey,"Skyline Cardinality for Relational Processing", Proc. Int'l Symp. Foundations of Information and Knowledge Systems. (FoIKS), pp. 78-97, 2004.
[17] X. Lin, Y. Yuan, Q. Zhang, Y. Zhang,"Selecting Stars: The k Most Representative Skyline Operator", Proc. Int'l Conf. Data Eng. (ICDE), pp. 86-95, 2007.
[18] Y. Lu, J. Zhao, L. Chen, B. Cui, D. Yang,"Effective Skyline Cardinality Estimation on Data Streams", Proc. Int'l Conf. Database and Expert Systems Applications (DEXA), pp. 241-254, 2008.
[19] Y. Tao, L. Ding, X. Lin, J. Pei,"Distance-Based Representative Skyline", Proc. Int'l Conf. Data Eng. (ICDE), pp. 892-903, 2009.
[20] Z. Zhang, Y. Yang, R. Cai, D. Papadias, A. Tung,"Kernel-Based Skyline Cardinality Estimaiton", Proc. ACM SIGMOD, pp. 509-522, 2009.
[21] L. Zhu, Y. Tao, S. Zhou,"Distributed Skyline Retrieval with Low Bandwidth Consumption", IEEE Trans. Knowledge and Data Eng., Vol. 21, No. 3, pp. 384-400, Mar. 2009.

P. Nagalakshmi, I am having two years experience in teaching from 2008-2010, iam having very much intrest in research oriented projects and in data mining and data ware housing.

Name: D.Umadevi
Designation: Associate Professor, Dept. of CSE&IT in Sri Mittapalli College of Engineering Tummalapalem, Guntur.
Teaching Experience: 12 Years & doing Ph.D.

Name: Dr. Shaheda Akthar
Designation: Professor, Dept. of CSE & IT in Sri Mittapalli College of Engineering Tummalapalem, Guntur.