An Efficient Multi-layer Grid Method for Skyline Queries in Distributed Environments

He Li, Sumin Jang, and Jaesoo Yoo

Department of Information and Communication Engineering,
Chungbuk National University, Cheongju, 361-763, Korea
{lihe,jsm,yjs}@cbnu.ac.kr

Abstract. The skyline query has been received much attention as an important operator in database systems for multi-preference analysis and decision making. Most of the previous works have focused on processing skyline queries on centralized data sets. However, the related data of real applications are practically scattered at several different servers. The skyline query computation in distributed environment is needed to gather a large number of data from the connected servers. The existing methods for a skyline query in distributed environment have two problems: (i) They have slow processing time for a skyline query. (ii) Most of the transferred data among servers in the network are unnecessary. In this paper, we propose a multi-layer grid method for efficiently processing skyline queries in distributed environments (MGDS). The proposed method minimizes the unnecessary transferred data using the grid mechanism. Experiments based on various data sets show that our proposed method outperforms the existing methods.

Keywords: Skyline query, distributed skyline query, grid method, distributed data.

1 Introduction

In the recent years, the interests on skyline query processing have been significantly increased since the skyline results can be used in many applications with multi-dimensional data set. Given a data set D containing objects D={p_1, p_2 … p_n}, the skyline operator returns all objects p_i such that p_i is not dominated by another object p_j.

Most of the previous skyline literatures [1], [2], [3] have primarily focused on providing efficient skyline algorithms on centralized data set. In practice, however, the vast numbers of independent data are often collected from multiple sources that stored in distributed servers. Figure 1 shows an example of skyline query in distributed environment. Assume a set of distributed servers {S_1, S_2, S_3, and QS}. Each server stores a set of tuples that is a fraction of the entire data set. The skyline query is initiated by users through query server (*QS*). The result of the skyline query is evaluated by gathering all of local data from the connected servers. A real-life example of the skyline queries in distributed environments is the online comparative shopping, in which a user needs to get good bargains from many different shopping sites according to

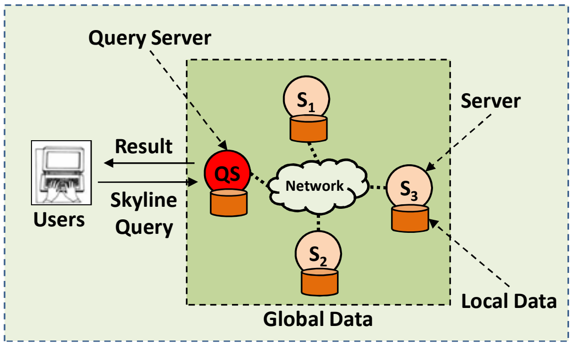


Fig. 1. The skyline queries in distribute environments

multiple criteria like price, quality, guarantee, etc. Such multiple criteria can be captured by a skyline query. In such cases, directly applying existing techniques would incur large overhead. Then, a distributed architecture is needed.

A naïve approach to process the distributed skyline queries is to send the skyline queries to all of the connected servers which in turn process the skyline queries locally and report the results to *QS*. The *QS* merges the received results and evaluates the global skyline result. This approach needs to transmit and process an excessive unnecessary data which is local skyline data but not global skyline data. Joao B. et al. in [4] proposed a grid-based strategy for distributed skyline query processing (AGiDS), which use a grid-based data structure to capture the data of each server. Instead of sending the local skyline data, the local cell information which contains local skyline data are firstly sent to the *QS*. The dominated local cells at *QS* are eliminated. Then only the local skyline data within the non-dominated cells are transferred to the *QS*. However, if the cells of the local servers transferred to *QS* are overlapped, lots of unnecessary data need to be processed.

In this paper, we propose a multiple layer grid method for skyline queries (MGDS) in the widely distributed environments. The proposed method assumes that each server shares a common grid structure. The *QS* first gathers the cell information which contains local skyline data. If the cell information is overlapped, we propose to generate a multiple layer grid based on the overlapping cells. The dominated cells of the multiple layers grid are eliminated. According to the multiple layers grid mechanism, more unnecessary local data are filtered out before transferred to the query server for processing.

The remainder of the paper is organized as follows. Section 2 surveys the previous related works. Section 3 presents the details of the proposed method. Section 4 contains an experimental evaluation that demonstrates the superiority of our proposed MGDS method. Finally, Section 5 concludes this paper.

2 Related Work

In [1], Borzsonyi et al. first introduced the skyline operation in database systems and proposed two solutions based on Block Nested Loop (BNL) and Divide and Conquer

(D&C). The nearest neighbor (NN) algorithm [2] indexes the data set with an R-tree. NN utilizes nearest neighbor queries to find the skyline results. In [3], the branch and bound skyline (BBS) algorithm was proposed. BBS is also based on nearest neighbors search and outperforms the NN approach. However, all these works assume centralized data storage.

Different from the skyline queries in centralized setting, skyline queries processing in the distributed and decentralized environments have been received considerable attention recently. Balke et al. [5] addressed skyline operation over multiple distributed sources, they consider that the underlying relation is vertically partitioned, i.e. each server keeps only an attribute of the relation. In this work, we focus on horizontal partitioning, where a server has all the attributes, but stores only a subset of all the tuples. Wang et al. [6] developed a skyline space partitioning (SSP) approach to compute skyline on a tree-structured p2p platform BATON. For this method, a server cannot freely decide the tuples in its own storage. Our techniques allow arbitrary horizontal partitioning. Cui et al. [7] study skyline queries in a distributed environment. They propose the use of MBRs (Minimum Bounding Regions) to summarize the data stored at each server. According to the MBRs of all servers, incomparable groups are assigned. The skyline is computed within each group using specific plans. In [8], a feedback-based distributed skyline (FDS) algorithm is proposed, which computes skyline in the no particular overlay network with economical bandwidth cost. The FDS algorithm is bandwidth efficient as the querying computer transmits to each server the precious information that prevents the delivery of a large number of non-skyline points. However, it requires several round-trips to compute the skyline, which incurs high response time. Joao B. et al. in [4] proposed a grid-based strategy for distributed skyline query processing (AGiDS). The response time of AGiDS is fast as it adopts the parallel computing over the distributed servers. However, if the cells of the local servers transferred to *QS* are overlapped, this method cannot efficiently reduce the unnecessary local data that are transferred from local servers for processing the global skyline.

3 The Proposed Method

3.1 Motivation

As mentioned before, when the cells of local servers transferred to the *QS* are overlapped, the AGiDS method leads to the transmission of unnecessary data. Therefore, we propose a new multi-layer grid method which processes skyline queries in distributed environments (MGDS). We assume that each server shares a common grid and the grid can include the entire data set. Given a set of distributed servers S= {S_1, S_2, S_3..., S_i}. Each server S_i stores a set of tuples that is a part of the entire data set and has the capability of computing the local skyline set based on the stored data points. The server who produces a skyline query is called query server (*QS*). Without loss of generality, we assume that smaller values are preferred in the skyline operator. In order to evaluate skyline queries efficiently, the proposed MGDS method uses three kinds of dominance relationships among the cells of grid. We consider that each cell of a 2-dimensional grid has a lower left corner coordinate value and a top right corner coordinate value.

Three kinds of dominance relationships among cells of grid

- $cell_i$ is dominated by $cell_j$, if the lower left corner coordinate of $cell_i$ is dominated by the top right corner coordinate of $cell_j$.
- $cell_i$ is overlapped with $cell_j$, if the lower left corner coordinate of $cell_i$ equals the lower left corner coordinate of $cell_j$.
- $cell_i$ dominates $cell_j$, if the top right corner coordinate of $cell_i$ dominates the lower left corner coordinate of $cell_j$.

If $cell_i$ is dominated by $cell_j$, which means that all the data points of $cell_i$ is dominated by any data point of $cell_j$. We define the cells which contain skyline data as region-skyline, as shown in Figure 2, the shaded area. If the region-skyline overlaps at different servers, we define these regions as overlap region-skyline, e.g. the cell A and B of Figure 2. If the overlap region-skyline contains more data points (*rCount*) than the predefined threshold value k (the value of k is defined according to the practical application), it is defined as hot overlap region-skyline, e.g. the cell B of Figure 2.

3.2 The Processing of MGDS Algorithm

The proposed MGDS method is comprised of three basic stages: planning, analyzing and execution. At the beginning of the planning stage, a skyline query can be initiated by a query server (*QS*). When receive the skyline query, each server S_i computes its local region-skyline by using an existing grid algorithm. The *QS* contacts all the connected servers and obtains the region-skyline information. In the analyzing phase, the received cells are analyzed and the global region-skyline is evaluated at *QS*. If the hot overlap region-skyline is occurred at the connected servers, it can be handled by creating an upper layer grid. As shown in Figure 2, if cell B is hot overlap region-skyline both at server S_1 and S_2, it is converted to an upper layer grid. The data points of cell B are managed by the upper layer grid and the local region-skyline of the upper layer grid is computed. Then, *QS* requests the local region-skyline of the upper layer grid and computes the global region-skyline. If the hot overlap region-skyline still exists on the upper layer grid, more layers grid can be generated. Notice that, though cell A is a overlap region-skyline both at S_1 and S_2, it is not converted to a upper layer grid, this is because there are only a small amount of data in cell A both at S_1 and S_2, and processing these data is efficient than processing the upper layer grid. In the execution stage, only the data points existed in the global region-skyline of each server are requested for the final global skyline computation. Since most of unnecessary local skyline data points are filtered out, the MGDS method reduces both the communication and processing cost.

Next, we illustrate the MGDS method by means of an example depicted in Figure 2. Assume that each server has a 2-dimensional data set and the grid consists of 3*3 cells. In the planning stage, the cells A, B, D of S_1's grid and A, B, C of S_2's grid are evaluated as local region-skyline and sent to the *QS*. In the analyzing stage, the received local region-skylines which dominated at *QS* are eliminated. Because cell B is hot overlap region-skyline, it needs to be converted to an upper layer grid, the data within cell B is managed by the upper layer grid and the local region-skyline B-1, B-2, B-3, B-4 and B-5 of the upper layer grid is evaluated at S_1 and S_2 respectively.

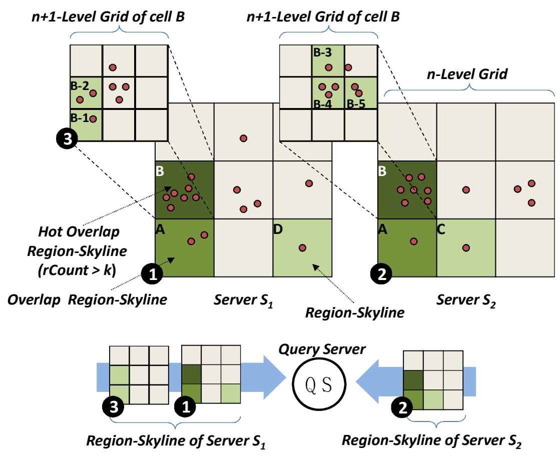


Fig. 2. The processing of the proposed MGDS method

The *QS* gathers the local region-skyline of the upper layer grid and evaluated the global region-skyline. As the cells B-3, B-4 and B-5 of S_2's grid are dominated by the cells B-1 and B-2 of S_1's grid which is not global region-skyline. Therefore, in this example, the global region-skyline is cells A, D, B-1, B-2 of S_1's grid, and cells A, C of S_2's grid. In the execution stage, only the local skyline data within cells A, D, B-1, B-2 of S_1's grid, and cells A, C of S_2's grid are transferred to the *QS* for final global skyline computation.

4 Experiment Evaluation

4.1 Experimental Environment

In this section, we study the performance of our proposed method. We present the experimental results comparing AGiDS method [4] and the naive method with our proposed MGDS method. We conducted our experiments on a desktop PC running on Windows XP professional. The PC has an Intel Core2 Duo 2.66GHz CPU and 1GB memory. All of the experiments were coded in Java. Table 1 shows the parameters for experiments evaluation.

In this experiment, there are two critical types of data distributions that stress the effectiveness of skyline methods have been used, the anti-correlated and uniform data set. The anti-correlated data represent that the data points that have a high value in one dimension and have a low value in one or all of the other dimensions. And the uniform data set is the data distributed in the arbitrary work space. All of the experiments are evaluated based on these two synthetic data sets.

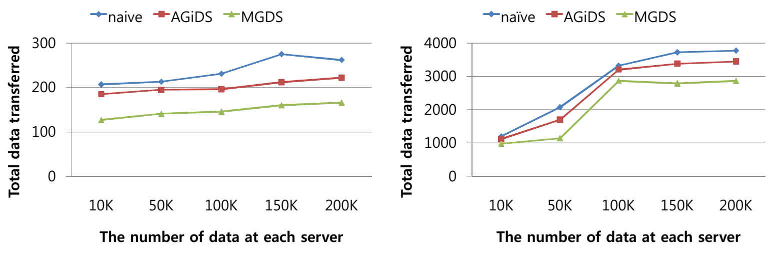
4.2 Experimental Results

We consider that all of the distributed servers are connected and each server posses an equal number of data points, the data points distributed in 30% range of the grid at each server. In the first experiment, the performance is measured by the amount of data transmitted over the network. We examine the performance of the methods by varying the number of data tuples from 10K to 200K at each server and the dimension of the data is set to 2. Figure 3 shows the results for the amount of transmitted data with respect to the number of data tuples. The skyline queries in anti-correlated data set transmit more data than in uniform data set. The reason is that the skyline data of anti-correlated data set is larger than the uniform data set. The MGDS method outperforms the AGiDS method and the naive method since more non-promising data points are filtered out by the multiple layer grids mechanism.

Figure 4 shows the response times of the three schemes according to the data size. In this experiment, the dimension of the data is set to 2 and the data size is varied from 10,000 to 200,000 at each connected server. From the results we can see that, the response time increases sharply when the data tuples are increased. This is because the increasing data size needs high processing cost which prolong the processing time.

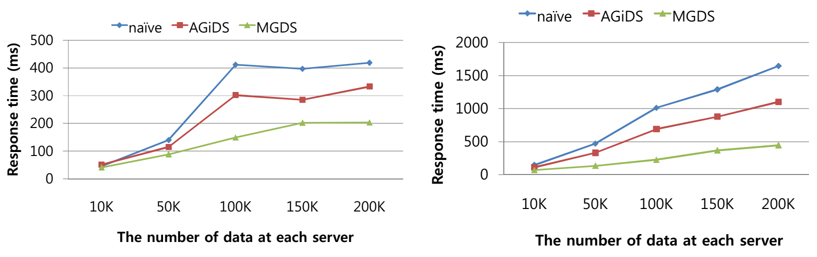
Table 1. The parameters for experiments evaluation

Parameter	Values
The number of dimensions	2~5
The number of servers	10
The amount of data at each server	10K~ 200K
The size of the grid	10*10



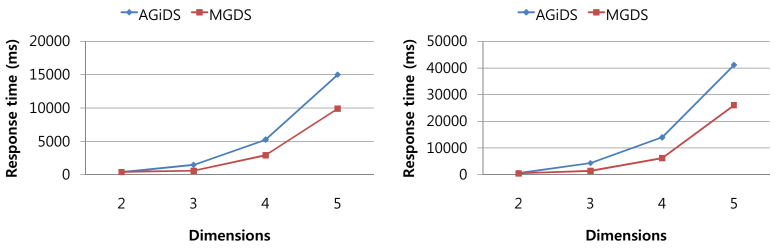
(a) Uniform data set (b) Anti-correlated data set

Fig. 3. Evaluation of the total transferred data for various data size



(a) Uniform data set (b) Anti-correlated data set

Fig. 4. Evaluation of the response time for various data size



(a) Uniform data set (b) Anti-correlated data set

Fig. 5. Evaluation of the response time for various data dimensions

The response time of MGDS is better than the response time of naïve and AGIDS for anti-correlated data and uniform data. This is because less data is processed.

As shown in Figure 5, the response time of the AGiDS method and MGDS method are compared according to the number of dimensions. In this experiment, the data size is set to 10,000. We study the results of the different methods with increasing the dimensions of the data set at each server. The response time of AGiDS increases more rapidly than the MGDS method when the dimensions increase. The reason is that with the increase of the dimensions more data points need to be processed.

5 Conclusions

This paper studies skyline queries over the horizontally partitioned data set. As the relevant data are scattered at several servers, the skyline query in distributed environment requires gathering a large number of data from the distributed servers that connected by the network. The proposed MGDS method employs a multi-layer grid mechanism to process the distributed data. It can filter out much of the unnecessary data which need to be transmitted to the querying server for final skyline result computation. The experimental results have shown that the proposed method is efficient than the existing methods.

Acknowledgments. This work was supported by the Ministry of Education, Science and Technology Grant funded by the Korea Government" (The Regional Research Universities Program/Chungbuk BIT Research-Oriented University Consortium).

References

1. Borzonyi, S., Kossmann, D., Stocker, K.: The skyline operator. In: 17th International Conference on Data Engineering, pp. 421–430. ICDE Press, Heidelberg (2001)
2. Kossmann, D., Ramsak, F., Rost, S.: Shooting stars in the sky: An online algorithm for skyline queries. In: 28th International Conference on Very Large Data Bases, pp. 181–184 (2002)
3. Papadias, D., Tao, Y., Fu, G., Seeger, B.: An optimal and progressive algorithm for skyline queries. In: ACM-SIGMOD International Conference on Management of Data, pp. 467–478. ACM Press, San Diego (2003)

4. Rocha-Junior, J.B., Vlachou, A., Doulkeridis, C., Norvag, K.: AGiDS: A Grid-based Strategy for Distributed Skyline Query Processing. In: Data Management in Grid and Peer to Peer Systems, Second International Conference, Globe 2009, Linz, pp. 12–23 (2009)
5. Balke, W.T., Güntzer, U., Zheng, J.X.: Efficient distributed skylining for web information systems. In: Hwang, J., Christodoulakis, S., Plexousakis, D., Christophides, V., Koubarakis, M., Böhm, K. (eds.) EDBT 2004. LNCS, vol. 2992, pp. 256–273. Springer, Heidelberg (2004)
6. Wang, S., Vu, Q.H., Ooi, B.C., Tung, A.K.H., Xu, L.: Skyframe: A framework for skyline query processing in peer-to-peer systems. VLDB Journal 18, 345–362 (2009)
7. Cui, B., Lu, H., Xu, Q., Chen, L., Dai, Y., Zhou, Y.: Parallel distributed processing of constrained skyline queries by filtering. In: 24th International Conference on Data Engineering, ICDE 2008, Cancun, pp. 546–555 (2008)
8. Zhu, L., Tao, Y., Zhou, S.: Distributed skyline retrieval with low bandwidth consumption. In: TKDE, pp. 384–400 (2009)