CS615: Group 04
Distributed Skylines

Devendra Kumar Luna
Student Id 23
15111015
dkluna@iitk.ac.in
Dept. of CSE

Nishit Majithia
Student Id 26
15111024
nishitm@iitk.ac.in
Dept. of CSE

Indian Institute of Technology, Kanpur

Final report
18th November, 2015

Abstract

The project involves Skyline Query processing in Distributed Environment. Given a large data set, it is inefficient to process all data items sequentially one by one. Therefore we propose a simple method to perform the Skyline query on large data sets using distributed systems.

1 Introduction and Problem Statement

Processing a large amount of data can be highly time consuming and performing skyline queries on such a data set can be a tedious task . So this project aims for reducing the total time taken in computing the skylines over a large data set by diving the data into two or more groups and processing those groups on separate systems and computing local skyline set of each group and ultimately combining those local sets to compute and obtain a global skyline set.

We have taken data sets containing over 1 lac data items . Two systems are used to compute the skylines on this data set. The first System acts as a master system and generates a query to compute skylines over the data set . Both the systems run BNL algorithm in order to compute the local skylines.

The objective is to process all the data elements in parallel and hence reduce the total time of computation.

1.1 Related Material

The datasets taken for testing have large number of data elements .

these data sets are taken from: UCI Machine Learning Repository

2 Algorithm or Approach

We propose a basic algorithm which uses the data distribution and parallel processing concepts. It uses a basic skyline computing algotrithm. The Skyline query is processed over multiple nodes(Systems) in parallel to get the time efficiency . There is one Master node which generates the skyline query and other nodes are known as slave nodes which, on being invoked by master node, process the data to compute skylines. The master node maintains pointers to the data set file to divide it into equal parts and keeps the first part for itself on which it invokes a skyline algorithm . All the pointers of other parts are transmitted to slave nodes over the cluster and these slave nodes in turn read their respective parts from the file and invoke basic Skyline Algorithm such as Block Nested Loop(BNL) on it's data and computes the local skylines . After successfully computing the local skylines , all the slave nodes transmit back the local skyline set to the master node . Master node merges all the local skyline sets into a global set and invokes skyline algorithm on this global set which gives the resultant global Skylines.

Algorithm 1 Distributed Skyline

1: Use BeoWolf clustering approach to make a cluster of nodes to process data in parallel

2: Master Node generates a query
3: Invoke all the the nodes of the cluster
4: Give ids to all the other nodes
5: Make a pointer in the data set which equally divides the data among all the nodes including itself(i.e. master node)

6: **for** each node of the cluster **do**
7: Read the data from the data set file using pointer
8: Store the data in local set
9: Invoke BNL algorithm on it's local set
10: Store the skylines into local skyline set
11: Pass the local skyline set to the master node
12: **end for**

13: Merge all the local skyline sets into it's local skyline set
14: Invoke BNL on merged local skyline set
15: return GLobal Skyline set

3 Results

The Distribution algorithm has been compared with Block Nested Loop (BNL) algorithm and the results are represented graphically in the figures shown below.

Analysis 1: As shown in the Figure 1 , Distributed Algorithm is compared with BNL where data size is kept constant as 1,00,000 and No. of dimensions are varied from 3 to 9. The analysis shows that for lesser no. of dimensions both the algorithms more or less take the same amount of computation time but as the no. of dimensions are increased, the time of BNL algorithm increases steeply and for high no. of dimensions Distributed takes almost half of the time that BNL takes.

Analysis 2:In figure 2, No. of dimensions are kept constant and data size is varied from 1000 to 1,00,000. When the data size is very small, the BNL algorithm is slightly better than Distributed algorithm which makes sense as Distributed algorithm has overhead of communication between the nodes which becomes more apparent when the computation time is less.As the data size is increased the Distributed algorithm performs better and when data size is very high, it proves to be highly efficient than BNL.

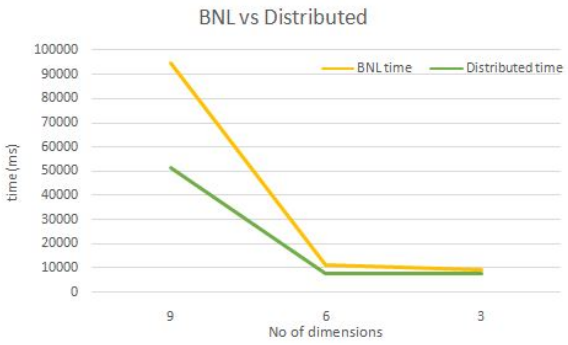

Figure 1: Time comparison with constant data size.

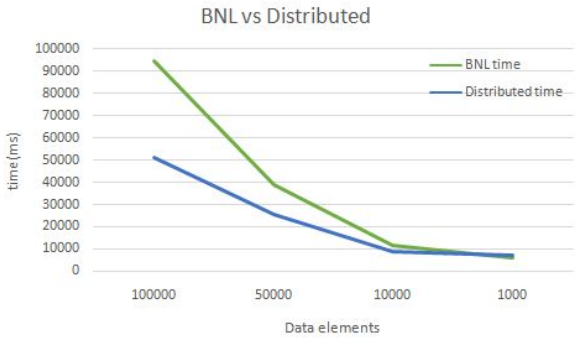

Figure 2: Time comparison with constant no. of dimensions.

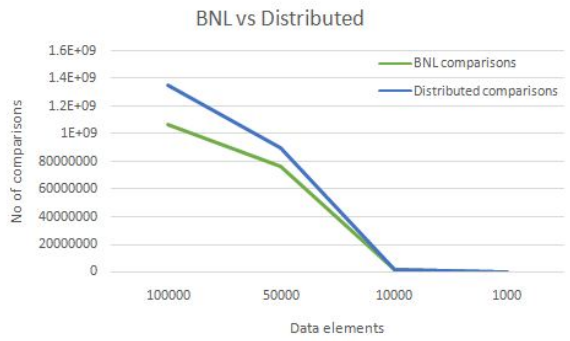

Figure 3: No. of comparisons

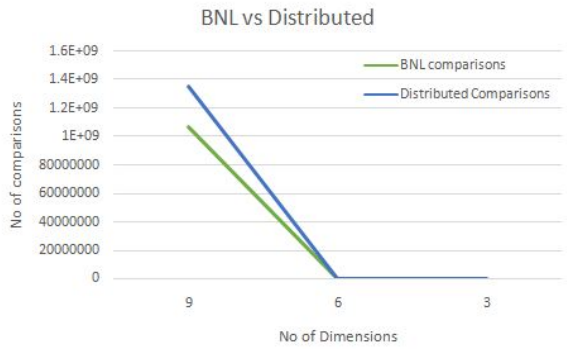

Figure 4: No. of comparisons.

Analysis 3:Figure 3 and Figure 4 show the analysis of No. of comparisons in both the algorithms.In Figure 3 the no. of dimensions are kept constant and data size is varied from 1000 to 1,00,000,similarly in Figure 4 the data size is kept constant and no. of dimensions are varied from 3 to 9. The analysis shows that no. of comparisons in Distributed algorithm will always be higher than BNL. Is is because all the nodes pass the local skyline sets to master node which merges those sets and forms a global set which is to be processed again to compute final skylines. So local skylines are processed again which increases the no. of comparisons.

4 Conclusions

From the analysis, following conclusions are made:

1. When the data size and/or no. of dimensions are very large, Distributed algorithm is the most efficient.

2. When data size is too small or data is highly co-related , the Distributed algorithm is not efficient because of the communication overhead.

3. When Data is highly anti co-related , the Distributed algorithm is not very efficient as the local skylines will be almost same as input. 4. Distributed algorithm is best suited for independent data sets.

Scope for future work:

This basic Distributed algorithm can be extended to accommodate a Data Analyzer which will first analyze the data for the suitability of Distributed algorithm , if the data is not suitable then the master node can process it locally instead of distributing it to other nodes.

Also, this basic approach includes BNL as the core skyline computation algorithm which further can be extended to accommodate other efficient algorithms.

References

UCI Machine Learning Repository