Perspectives in Communication, Embedded-Systems and Signal-Processing (PiCES)
ISSN: 2566-932X, Vol. 2, Issue 11, February 2019
Proceedings of Knowledge Discovery in Information Technology and Communication Engineering (KITE-2018), 4th May 2018

Efficient Computing of Skyline Queries using SKY-MR+

Diksha Dwivedi, Ipsita Rakshit, Devishree S
Student, ISE Department, EPCET, Bangalore

Nandini Gowda P
Asst. professor, ISE Department, EPCET, Bangalore

Abstract: The skyline operator is introduced due to its wide range of applications but this process is challenging in case of big data. Mapreduce framework is considered for applications that are data-extensive. For parallel processing of the application SKY-MR+ with the mapreduce is used..In this process the benefit of splitting the terms is done based on estimated execution time. Dominance power filtering method is applied to effectively eliminate non-skyline points . Data partitioning is done based on the region surrounded by the quad tree.It is checked whether each and every skyline candidate point is actually a skyline point using MapReduce. Workload balancing techniques are used to make estimated execution time of all the machines same. Experiments are done to compare SKY-MR+with the state-of- the-art algorithms using MapReduce.

Keywords: Skyline queries; MapReduce algorithms; Distributed and parallel algorithms

I. INTRODUCTION

The skyline operator and its variants [1],[2],[3] have gathered a lot of attention due to its wide range of applications like querying wireless sensor networks [6] and graph analysis [7]. Today computation of skyline has become a challenge due to big data .hence algorithms to compute skyline using mapreduceis proposed [11], [12], [13], [14].

MR-GPMRS [11] consist of two phases partitioning and global skyline phase. In partitioning phase we divide the data into grid partition and prune the partition that doesn't contain any skyline point using dominance relationship. In global skyline phase a skyline point in unpruned parts are taken and points p dominated by them are collected and both are compared to determine whether they are skyline points or not. To reduce the overhead involved for computation and distributing the points a local phase is introduced which calculates the local skyline points and use it to compute global skyline phase.

MR-BNL[12],PPS-PGPS[13] and SKY-MR[14] have additional local skyline phase.SKY-MR is a state of art which compute skyline using mapreduce.to split data the algorithm SKY-MR builds sky- quadtreeusing sample data based on split threshold so if reasonable throughput is not provided then the performance degrades. SKY-MR ignores the workload balancing of the machines so performances of algorithm decreases as the number of machines increases.

Adaptive quad tree building: In this each node is split judiciously based on whether splitting is beneficial or not based on execution time.

Effective workload balancing: We balance the workload of of machines by introducing the workload balancing techniques to make the execution time of all machines similar.it is same as multiprocessor scheduling problem [15] which is NP-hard we use effective approximation algorithm [15].

Efficient skyline points: before computing the local skyline points it is important to remove as many non-skyline points to reduce the overheads. For this we use dominance power filtering method[16] which contains points which dominate many other points so it is desirable to remove the points which is dominated. This is called dominance power filtering method.

Proportionality between actual and estimated time: The workload can be balanced for both the phases if there is a correlation between actual and estimated execution time. To show this we measure pearson correlation coefficient PCC [34] and kendell's £ coefficient [35]. local skyline phase, average PCC over IND, COR and ANTI data sets is 0.37 which represent weak correlation. The average PCC for the global skyline phase is 0.74 which means strong correlation. Kendall's£ and 0.57 for the local and global skyline phases respectively. The estimated execution times of the global skyline phase show stronger correlation than local phase.

II. RELATED WORK

Many serial skyline algorithms have been introduced that use centralized indexing structure but they are not compatible to be parallelized with MapReduce because MapReduce does not let us build and access centralized indexing structures. We use state-of-the-art algorithm BSkyTree-P[24] to calculate local skyline for each partition. BSkyTree-P will split the data space in 2d partions by selecting a pivot point first. Then the points that are dominated by pivot points and then the algorithm recursively divides all the partitions into smaller ones. It then puts together all the partitions and calculates local

skyline point of every merged partition till there is only one partition left and then compute the global skyline point.

There were many skyline processing algorithms introduced like MR-GPMRS [11], MR-BNL [12], PPF-PGPS [13] and SKY-MR [14]. MR-GPMRS [11] consists of two phases partitioning phase and global skyline phase. MR-BNL [12], PPF-PGPS [13] and SKY MR consists of a third phase too called local skyline phase. In partitioning phase the space is split into partitions in sky-quadtree partitioning in SKY-MR, in PPF-PGPS they use angle based partitioning and grid partitioning in GPMRS and BNL. The machines that take part in the MapReduce framework could not be fully used because MR-BNL d machines in the local skyline phase where d is the number of dimensions. In MR-BNL only one machine is used to compte global skyline phase so the algorithm becomes inefficient. SKY-MR uses all the available machines in both faces and also uses a pruning technique along with dominance power filtering method.

In [14] shows SKY-MR is better than PPF-PGPS. SKY-MR is much better than MR-GPMRS because MR-GPMRS does not consist of local skyline phase. The algorithm PPPS uses the angle-based space partitioning [32]. PPPS recursively divides every partition into two partitions till the number of the partitions becomes the required number of CPU cores c. For each partition we compute single skyline phase. However, SKY-MR computes the global skyline points by taking every partition and use all the machines simultaneously. In [14] that SKY-MR is has better performance to the MapReduce of the algorithms in [27] and [28].

III. PRELIMINARIES

A. Skyline

Consider the d dimensional data set D={P1,P2,P3..,PD}. A point pi is represented pi(1),pi(2),pi(3)….pi(D)},where pi(k) is kth coordinate of point p. A point pi dominates point pj hence it is notified as pi<pj. To proceed this mechanism we try to satisfy two conditions, they are:(a) for each k , $1 \leq k \leq d$, then we have $pi(k) \leq pj(k)$,(b) there exists k with $1 \leq k \leq d$ such that pi(k) <pj(k) satisfies. We denoted pi is not dominate to pj by writing pi(/<)pj.

Table 1. List of Notations

Notation	Description
S	A sample of d-dimensional data D (i.e., $S \subseteq D$).
$SL(P)$	The skyline of a set of points P.
vp_n	The virtual max point of a node n of *sky-qtree*$^+$*s* or *sky-quadtrees*.
$region(n)$	The region covered by a node n of *sky-qtree*$^+$*s* or *sky-quadtrees*.
$P(n)$	The set of points $p \in D$ located in $region(n)$.
$S(n)$	The set of sampled points $p \in S$ located in $region(n)$.
$C(n)$	The set of candidate split points $p \in SL(S)$ located in $region(n)$ (i.e., $C(n) = SL(S) \cap S(n)$).
$R(n)$	The set of all local skyline points which dominate vp_n of a leaf node n and are located in the other leaf nodes (i.e., $\bigcup_{n' \neq n} \{p' \in SL(P(n')) \mid p' \prec vp_n\}$).
$t(Q)$	The estimated execution time of *SKY-MR*$^+$ with a *sky-qtree*$^+$ Q.

B. The Mapreduce framework:

The mapreduce framework is the open source equivalent hadoop. It is widely used for parallel computation for clusters of machine. In hadoop the value or the data is represented as the key value pairs. This data is then divided into fixed size called chunks and this each and every chunk is then assigned to a task called mapper task.

After this the map function is invoked to the each pair of the chunk, which produce an partial output, and then this output will grouped in shuffling phase and then the final output is made by sending the reducer task. The user can modify the output by using the partition class. The reducer function is used to generate more key value pairs. The setup function is set before map function and reduces function, and cleanup function is executed after map function and reduces function. The main function of the hadoop in single master machine.

C. SKY-MR: The state of the artalgorithm

There are three phase as follows:

a) Sky-quad tree building phase

The SKY-MR builds a sky-quad tree .The sample data is given. Splitting the data space into many partitions. Then the d-dimensional data space is the sub-divided recursively into 2d equi-sized sub-regions each of which is associated with a node of the sky-quad tree.

The pruning of the quad tree is done until each section will have the predefined number of points p .This region is called the split threshold. According to dominance power relationship every node without skyline point is considered as the pruned.

b) The local skyline phase

For each and every un pruned leaf node n of the sky-quad tree, the local skyline of P(n),is denoted by SL(P(n)) and is computed where P(n) is all points in the region represented by n.

Virtual max point is to reduce the dominance power filtering and sky filter points are computed after the local skyline is computed.

c) Global skyline phase

Each and every local skyline point in unpruned leaf node is compared to the global skyline points. If the the local skyline point is dominating the global skyline point the then the particular point is consider as the global skyline point.

d) Drawbacks

When we apply the split threshold the output generated by the this mechanism gives maximum points in each node, which affects the system performance.The split threshold has adverse effect in the network over transmission over the duplicate nodes and the costly.By considering the workload balancing there is degrade of the performance in the system.

IV. SKY-MR+: OUR SKYLINE COMPUTATION ALGORITHM

We develop pseudo code for SKY-MR+ which is similar to the SKY-MR[14]. We generate the mapreduce function to proceed the process.SKY-MR+ consists of sky-qtree+ building , local skyline and global skyline phases.

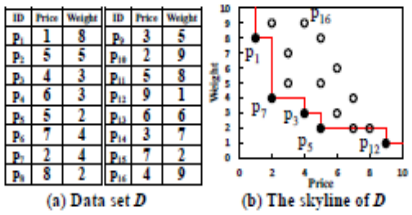

Fig 1. An example of a skyline

A. Sky-qtree+ building phase

A sample S from data D is produced, we then implement the procedure SKYQTREE+ with the sample S and the number of machines m which is used to build a sky-qtree+.

The pseudocode of SKY-QTREE+ is given below. When start calculating the parameters A and B ,which is used to initializes a sky-qtree+ Q and a maxheap M . Each a every node n of the sky-qtree+ has two attributes S and C which is used respectively as the sample points and the candidate split points in region(n).The candidate is used to split points of the root node n and is usedset to the skyline of S by implementing BSkyTree-P [24] .It next finds the estimated total execution timebt(Q) of the local and global skyline phases . For each and every leaf node nextracted from the max heap M, we decided whether n should be split or not.

Local balance SKY MR+ is called to assign each and every node to a particular machine and this mechanism takes place by workload balancing algorithm. L-MR-SKY finds the unpurned leaf nodes in the assigned nodes.AL is identified by SKY-MR+ , and is loaded into the main memory of the system. Each mapper function is associated with point p.

If the given p belongs to the region of an unpruned leaf node np of Q and if p is not dominated by every dominating point in Hi, p is added into Ui and is updated with p is updated in Hi. After the map function is done is the cleanup is initialized.

V. EXPERIMENTS

A. Experimental environment:

We constantly experiment the performance of the tested algorithms using the parameters in the table. We add the source code of MR-GPMRS used in [11] and re generated the code with a minor optimization. We stop reporting the system when it exceeds more hours.

Data set: For experimental study, we evaluate the algorithms with synthetic data and real time data. We started generating the synthetic data sets by anti-correlated, independent and correlated distributions, and it is denoted as ANTI, IND and COR.

ANTI, IND and COR are used to evaluate the performance of the skyline algorithms [11], [12], [13], [14]. As d increases, the number of skyline points increases exponentially for ANTI and IND data sets.

Function SKY-QTREE$^+$(S, m)
S: a sample of data, m: the number of machines available
1. (A, B) = FindConstant(S);
2. $Q.root$ = InitTree();
3. $Q.root.S = S$; // sample points of $Q.root$
4. $Q.root.C$ = BSkyTree-P(S); // candidate split points of $Q.root$
5. M = InitMaxHeap();
6. M.add($Q.root$);
7. $\hat{t}(Q)$ = ComputeEstimatedTime(Q, A, B);
8. **while** !M.isEmpty() **do**
9. n = M.extractMax();
10. s = FindSplitPoint(n);
11. Q' = Q.Split(n, s);
12. $\hat{t}(Q')$ = ComputeEstimatedTime(Q', A, B);
13. **if** $\hat{t}(Q) > \hat{t}(Q')$ **then** // split node n with respect to s
14. $\hat{t}(Q) = \hat{t}(Q')$;
15. $Q = Q'$;
16. $n.child$ = AllocateChild();
17. **for** each node n_c in $n.child$ **do**
18. $region(n_c)$ = setRegionOfChild(n, s);
19. $n_c.S$ = getPointsInRegion($region(n_c)$, $n.S$);
20. $n_c.C$ = getPointsInRegion($region(n_c)$, $n.C$);
21. **if** $region(n_c).min = s$ **then**
22. $n_c.pruned$ = **true**
23. **else if** $|n_c.C| \neq 0$ **and** $region(n_c).max \neq s$
24. M.add(n_c);
25. **return** Q;

Fig 2. Sky Quad Tree+ algorithm

B. Performance Analysis

a) Synthetic Data Sets

Default values of |S|, k and ρ: To determine the values of a sample size |S| for SKY-MR+ and SKY-MR, we executed both algorithms with different values of |S| from 100 to 10,000. We also varied dominating pointsk from 10 to 1,000 and split threshold ρ from 10 to 40. The average execution times of SKY-MR+ and SKY-MR with three data sets along with default values are given Fig 3.

By using a sample S, we build a sky-qtree+ and estimate the execution time of SKY-MR+ for workload balancing. When |S| decreases. The performance of SKY-MR+ also comes down since a small sample cannot reflect the data distribution accurately and so the workloads of machines maybe skewed due to the wrong estimated execution time. The execution time of building a sky-qtree+ grows with increasing |S| since the costs of computing SL(S) and find the split points of each node of the sky-qtree+ increase. As the size of the dominating point set k grows in size, the number of points taken out by the dominating point set and the cost of managing the dominating point set increase. Selecting a small value or a large value of k is not of use. In Fig. 3(a), SKY-MR+ showed the best performance when |S| = 400 and k = 100. Thus, we select 400 and 100 as the default values of |S| and k. Fig. 8(b), SKY-MR shows a similar design as of SKY-MR+ with increasing |S|. For SKY-MR, we select 200 and 10 as the default values of |S| and ρ, respectively. We would like to show that SKY-MR+ is less changed by changing the parameter values than SKY-MR.

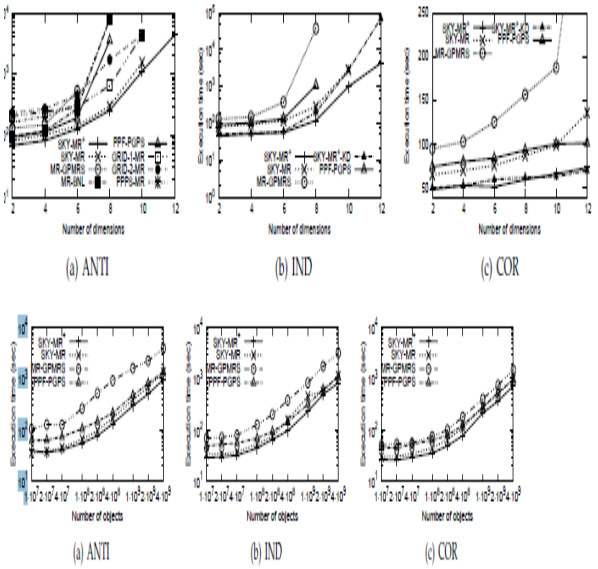

Fig 3. Parameter comparison of SKY-MR+ and SKY-MR

b) Real-lifedataset

The data distribution and statistics of the real-life data set are fixed, we vary only a few parameters for the real-life data set HEPMASS.Default values of|S|, k and ρ: With the first 10 features of HEPMASS (i.e., d = 10 and |D| = 1.05×107), we executed SKY-MR+ and SKY-MR with varying |S| from 100 to 10,000. K is also varied for SKY-MR from 10 to 1,000 and ρ for SKY-MR from 10 to 40.

C. The effects of optimization techniques

The execution time (in seconds) of the SKY-MR+ is calculated with the help of dominance power filtering (D),SKY-MR+ uses workload balancing only (B).

When we use both the techniques we represent it as (ALL) and when we don't use the techniques we represent it as (NONE).When the workload balancing is not there in the SKY-MR+ we automatically rely on the hadoop partitioning. Hence we partition the local skyline and global skyline with default technique. When d is smaller and the COR data , the number of skyline is smaller. The workload balancing techniques increases with increase in skylines. The execution time of the machines is reduced by balancing the workloads of available machines.

VI. CONCLUSION

We study the parallel skyline computation using MapRe-duce and make the algorithm SKY-MR+. A sky-qtree+ is made first with an adaptive quadtree building technique to use the dominance relationships between regions andapply the dominance power filtering method to prune out non-skyline points in advance. SKY-MR+ partitions the data based on the regions divide by the sky-qtree+ and calculate the candidate skyline points for every partition. Finally, we check whether every skyline candidate point is actually a skyline point in every partition independently. To make the estimated execution times of all available machines to be similar, we develop workload balancing techniques. Our experimental results confirm the effectiveness and scalability of SKY-MR+.

REFERENCES

[1] S. B¨orzs¨onyi, D. Kossmann, and K. Stocker, "The skyline operator," in ICDE, 2001, pp. 421–430.

[2] D. Papadias, Y. Tao, G. Fu, and B. Seeger, "An optimal and progressive algorithm for skyline queries," in SIGMOD, 2003.

[3] E. Dellis and B. Seeger, "Efficient computation of reverse skyline queries," in VLDB, 2007, pp. 291–302.

[4] J. Lee, S. won Hwang, Z. Nie, and J.-R. Wen, "Navigation system for product search," in ICDE, 2010.

[5] T. Lappas and D. Gunopulos, "Efficient confident search in large review corpora," in ECML/PKDD (2), 2010.

[6] G. Wang, J. Xin, L. Chen, and Y. Liu, "Energy-efficient reverse skyline query processing over wireless sensor networks," TKDE, vol. 24, no. 7, 2012.