RESEARCHERID
THOMSON REUTERS

[Chen* 5(6): June, 2018]

ISSN 2349-4506
Impact Factor: 3.799

Global Journal of Engineering Science and Research Management

TOP-K DOMINATING QUERY PROCESSING OVER DISTRIBUTED DATA STREAMS

Guidan Chen *, Yongheng Wang

*College of Information Science and Engineering, Hunan University, Changsha, China
College of Information Science and Engineering, Hunan University, Changsha, China

KEYWORDS: Top-K dominating query, streaming data, k-skyband, spark streaming

ABSTRACT

Data stream has been widely used in lots of modern applications such as Social networks and the Internet of things. Aiming at the problem of Top-k dominating query in distributed data stream, a distributed Top-k query algorithm based on Spark Streaming framework is proposed. Based on partitioning, double pruning techniques are implemented on the data. Local and global pruning can significantly reduce the number of candidate sets, reduce the computational overhead and space costs, and improve the query efficiency. Experimental results show that the algorithm has good performance and scalability.

INTRODUCTION

With the rapid development of social networks, Internet of Things, mobile Internet, etc. huge amount of data has been produced, and the era of big data has arrived [1]. Big data has the characteristics of huge amount of data, high real-time requirements, low value density, and the value of data will decrease with the passage of time. Therefore, how to process data streams in real time and obtain more valuable information is the key to streaming big data research. Top-k queries and skyline queries are most widely used in streaming data environment .

In a Top-k query, a ranking function F is provided for determining the score of each object and k objects with the F scores are returned [2] .The great advantage of Top-k queries is that users can control the number of data objects in the result set by specifying the parameter k. The drawback of the queries, on the other hand, is that it is not always easy for users to specify an appropriate ranking function [3].

skyline queries overcome this drawback because this query does not require any ranking functions. The skyline is composed of the objects that are not dominated, based on a domination relationship involving the values in each dimension. Through skyline query, users can obtain data objects that are not worse than others. However, users cannot control the size of the result set.

Top-k dominating queries, which have been receiving much research attention recently, return the k data objects that dominate the highest number of data objects in a given dataset, it is a combination of Top-k and skyline queries: it uses a ranking function to rank points (as in Top-k query) and it uses the dominance relationship (as in skyline query).Due to its dominance-based ranking function , it does not require any ranking functions, which not only

controls the size of the result set, but also overcomes the problem that the traditional Top-k queries score function is not well specified.

In this paper, a distributed Top-k dominate query algorithm based on the Spark Streaming computing framework is proposed—FDTop-k (Filter-based Distributed Top-k dominating query) algorithm. Firstly, the original data is divided into data by using the method of hash map. Filter-based algorithm is used in each partition to prune the original data set to get the local candidate data set, and then the local candidate set is further improved by using the attribute of RDD itself. The pruning gets a global candidate set, and then finds the result set of the Top-k query according to the dominant score of the global candidate set. Finally, a large number of experiments based on real data sets proves the efficiency of the algorithm.

The rest of this paper is organized as follows. Section 1 discusses existing Top-k processing methods in various scenarios. Section 2 introduces filter-based distributed Top-k dominating query algorithm. We show the results of our experiments in Section 3. Finally, in Section 4, concludes the paper with directions for future work

RELATED WORK

Efficient Top-k query processing techniques have been studied in both centralized [4] and distributed databases [5] . The research of early Top-k queries mainly centered on centralized relational database. TA algorithm (threshold algorithm), NRA algorithm [6], Stream-Combine algorithm [7], Minimal probing algorithm [8], Upper and Pick algorithm [9] is a typical representative of this type of algorithm. This type of algorithm is mainly aimed at a single database environment and does not consider the distributed environment. Although it is improved, the communication cost is still large. With the increasing amount of data, distributed Top-k queries have been widely concerned by scholars. a three-phase uniform threshold (TPUT) [10] method is proposed . The TPUT algorithm increases the bandwidth consumption and node load of the entire network and does not apply to the actual network. The KLEE [11]algorithm improves the TPUT by reducing the communication cost in the network and reducing the load between the nodes. It improves the query efficiency and leads to a decrease in the quality of the results, and the bandwidth saving and communication steps become Inversely. Top-k dominating queries have first been proposed in [12]. Multi-dimensional Top-k dominating queries have been proposed in [13]. Sliding window Top-k dominating query processing over distributed data streams has been studied in [14]. Top-k dominating queries in distributed environments have been studied in [15]. Top-k dominating queries in metric space have also been proposed [16] .Compared with the existing work, our work is mainly to use the current popular big data platform Spark streaming to study Top-k dominating queries on distributed data streams. Based on the implementation of partitions, we implement double shearing on the partitioned data. Branch technology to process data on distributed data streams in real time and return query results to users. This method not only reduces the computational overhead and space overhead, but also reduces the processing delay and improves the real-time performance of the query.

RESEARCHERID
THOMSON REUTERS

[Chen* 5(6): June, 2018]

ISSN 2349-4506
Impact Factor: 3.799

Global Journal of Engineering Science and Research Management

FILTER-BASED DISTRIBUTED TOP-K DOMINATING QUERY ALGORITHM

Query Framework and Problem Description

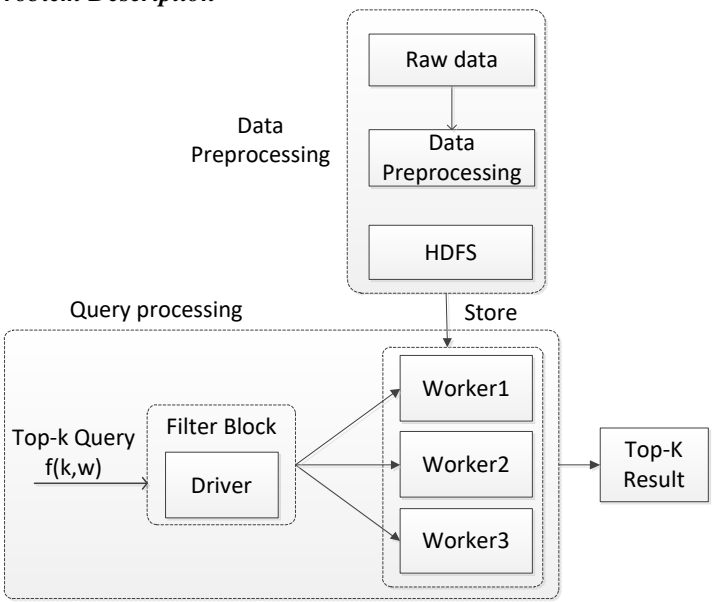

Figure 1 Top-k query process on spark framework

The Top-k dominate query is mainly divided into two parts: (1) data preprocessing; (2) query processing. The basic framework of big data processing adopted in this paper is the HDFS +spark framework, as shown in Figure 1.

During the data preprocessing stage, in order to achieve distributed parallelism, the raw data needs to be divided. The basic principle of data division is to divide the data evenly into all nodes as much as possible, In order to ensure that the data on each server contributes to the final result, the article adopts the method of hash mapping to divide the objects with the same hash value into the same partition. it is simple and convenient, and each object can be Map to a unique partition. The divided data is mapped to different blocks, each block is marked and then stored on the HDFS .

During the query processing stage, for a query f(w, k), the driver node accepts the parameters k and w. According to the algorithm in the paper, part of the data set is selected instead of all the data sets for query. Then we process the data set through the Spark cluster and return the Top-k result.

Our proposed approach is based on some observations which are derived from an important concept of dominance and k-skyband [12]. We first define those concepts.

D={d_1, d_2, …, d_n} is an n-dimensional space, d_1, d_2, …, d_n are attributes of space D, S is a set of points on space D, for any point $p \in S$, p is The value on the attribute $d_i (1 \le i \le n)$ is denoted as $p.val_i$. Without loss of generality, a small value is preferable and objects are independent of each other.

Definition 1 (Dominance) given data points p and q , p is said to dominate q , denoted as $p \prec q$, if they satisfy the following condition.

$$(\forall d_i \in D, p.val_i \le q.val_i) \wedge (\exists d_j \in D, p.val_j < q.val_j) \quad (1)$$

Definition 2 (Top-k dominating queries) Consider a data point $p \in S$, its score(p) is calculated as follows.

$$score(p)=|\{p, q \in S | p \prec q\}| \quad (2)$$

The Top-k dominating query returns the k data objects with the highest score, namely TOPK.

Definition 3 (dominated score) Given a data point $p \in S$, its dominated-score $\varphi(p)$ is calculated as follows.

$$\varphi(p) = |\{q \in S \mid q \prec p\}| \quad (3)$$

Definition 4 (k-skyband) The k-skyband returns points in the set S that are dominated by no more than k points .

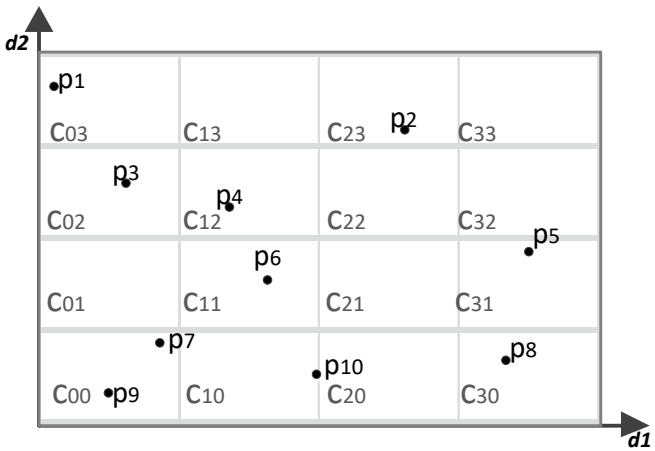

Figure 2 A two-dimensional dataset and grid cells

Example 1 Figure2 illustrates a two-dimensional dataset of ten data points, Take figure 2 as an example to illustrate dominance , k-skyband queries ,Top-k dominating queries . According to definition 1, p3 dominates p2 because all the attribute values of p3 are less than those of p2. According to definition 3, the set of k-skyband(if a user sets 2 as k, namely 2-skyband) is{p_1, p_3, p_4, p_6, p_7, p_8, p_9, p_{10}}, these points are dominated by not more than 2 other points. In this case, if a user sets 2 as k, TOPK = {p7, p9}, because their scores are 4 and 8 respectively and larger than those of the others.

From definitions discussed above and the study in[12],[14][15], we have observed important properties and Lemmas.

Property 1 TOP-K belongs to k-skyband.

Property 2 Any data object p∈S, if its k-skyband is greater than k, then p ∉ TOP-K.

Lemma 1 A data object whose k-skyband is not less than k cannot be final answer.

Lemma 2 If a data object p is dominated by other object whose k-skyband is not less than k, p cannot be final answer.

Motivated by above properties and Lemmas, two important strategies could propose to significantly prune unnecessary data points.

(1)A data object with whose k-skyband not less than k has a high possibility to prune data objects of other servers, even if the dominated score of the data objects are less than k (from Lemma 1).

(2)Data objects included in the local k-skyband have a high possibility to be included in the final answer, and can prune unnecessary data points because data points in the k-skyband can dominate many data points (from Lemma2).

The algorithm for DominatedScoreCompute (P, Q, k, D) is represented in Algorithm 1.

Algorithm 1 DominatedScoreCompute(P , k, D)

```
Input: P , Q    // P and Q are sets of objects
1  for ∀p ∈ P do
2     for ∀q ∈ P do
3        If q≺ p then
4          φ(p) ← φ (p)+1 //φ(p) is dominated score of p
5          If  φ(q) ≥ k then
6             φ(p) ← φ(p)+ φ(q)
7          if  φ(p) ≥ k  then
8             break
```

Lines 3-4, if $q \prec p$, namely p is dominated by p, $\varphi(p)$ plus 1. Lines 5-6, if $\varphi(q) \geq k$, $\varphi(p)$ updated. Lines 7–8, $\varphi(p) \geq k$ can be safely pruned away as they will not be TOPK candidate, so we can reduce computation cost by avoiding such unnecessary computation of dominance relationship.

Filter-based Distributed Top-k Dominating Query Algorithm

TOP-K belongs to k-skyband . It only needs to iterate through the k-skyband objects of each partition without traversing all the objects of the partition, avoiding the calculation of dominance of other unrelated objects, which greatly improving the efficiency of the query. Divide the data set into each partition node and execute each algorithm separately. The candidate set is obtained by screening the original data set, reducing the comparison between unnecessary data sets and speeding up the Top-k dominating query.

In in first step of the algorithm, local k-skyband obtained through the local pruning algorithm. In the process of local pruning, if the classical k-skyband algorithm in [17] is adopted, the result of the candidate set will be very large, because this algorithm maintains a superset of k-skyband. When merging partitions, RDD will generate a lot of computation cost that will affect the performance of the algorithm. Motivated by [15], we use a more efficient pruning strategy to reduce the number of candidate sets. Filter-based solution algorithm, it can reduce the size of candidate of *TOP-K* by exploiting SFSKY(Subspace Filtered Skyline) and τ-skyband in *k*-skyband and updated *TOP-K* incrementally and efficiently.

SFSKY is the set where p is a subspace skyline object in the set of data objects with dominated-score not less than k. $SKYB^{\tau}$ is the set where dominated score is less than τ in the local data object set, and we define τ as follows[15].

$$\tau = \alpha \times k \qquad (0 \leq \alpha \leq 1) \qquad (4)$$

It is intuitive that $SKYB^{\tau}$ which applies the case where $\alpha = 1$, is costly because local k-skyband contains some data points not in the final result. Actually, small α, e.g., $\alpha < 0.5$ would be sufficient because data objects with small dominated-score have a high dominance power. The detailed algorithm is shown in Algorithm 2.

Algorithm 2 Filter-basedCompute(P,k,D)

Input: P,k

1 DominatedScoreCompute(P, k, D)

2 FT← $\{\forall p \in P | \varphi(p) \geq k, \nexists q \in P$ where $\varphi(q) \geq k$ *and* $q \prec p\}$

3 SFSKY←{ ∀p∈FT|∄q∈FT, q≺p in subspaces except for the entire space of D}

4 $SKYB^{\tau} \leftarrow \{\forall p \in P | \varphi(p) < \tau\}$

5 **for** ∀p ∈ SKYB **do**

6 $\varphi(p) \leftarrow 0$

In the second step, global k-skyband is obtained through the global pruning algorithm.

After the local pruning ,The candidate may not be the global k-skyband in the end, so it is necessary to further filter the candidate set to obtain the global k-skyband. According to the k-skyband RDD obtained by the first step pruning, use the Cartesian product of RDD itself to obtain the key-value form of <recl,rec2>; then Map compares the dominance relationship between the two records if recl ≺ rec2, then becomes <recl,1>, otherwise <recl,0>. Finally, ReduceBykey gets the number of dominating objects for each record and gets the global k-skyband.

In the third step, top-k dominating is solved.

The dominant score of the global candidate set is calculated and the Top-k result set is found.

The detailed algorithm is shown in Algorithm 3.

Algorithm 3 Filter-based Distributed Top-K dominating query algorithm

Input: Q //Query input
Output: TOPK // Top-K dominating query set
1 i=1;
2 TOPK={};
3 Original_Rdd=sc.textFile()
4 Del_Rdd=Orignal_Rdd.map(Q)
5LocalSky=Del_Rdd.MapPartition(Filter-basedCompute(,k,D))
6 GlobalSky=LocalSky.Cartesian(LocalSky).map().reduceBykey.filter(iter.value<k)
7 domScore=Del_Rdd.Cartesian(S).map.reduceBykey(_+_)
8 TOPK= domScore.top(k)

The algorithm uses the class Scala language. Line 3 obtains the original data set. Line 4 processes the data to get Del_Rdd , which is stored in each partition. Line 5 is to use the MapPartition interface provided by Spark to execute the Filter-based solution method to solve the k-skyband separately in each partition. Line 6 is the k-skyband candidate set that merges each partition. Line 7 is to get the dominant score of the global k-skyband. Line 8 gets the Top-k dominating result set.

EXPERIMENTAL RESULTS

Setting

The experiment was conducted on a four-node cluster of sparks. Spark is built on Hadoop, using Hadoop's Yarn Explorer and HDFS. One of the nodes is a master node and the other three are worker nodes. The node configuration is as follows: CPU: Xeon E5-2620, memory: 8GB, hard disk: 1TB, operating system: Ubuntu 14.04.2 (64bit), and the node running the spark-2.0.2 version.

The experimental data uses the real data set of the New York Taxi Trip Report released by the DEBS2015 grand challenge [18]. The data was publicly available by Chris Whong in March 2014. Provided data consists of reports of

taxi trips including medallion, starting point, drop-off point, corresponding timestamps, and information related to the payment. The data file is sorted chronologically according to the drop off datetime.

Experiment Analysis

The algorithm Filter-based distributed Top-k Algorithm, abbreviated as FDTop-k algorithm, proposed in this paper is compared with the classic algorithms in Top-k query, such as minTop-k [19] and k-skyband [17]. It mainly includes the comparison of the average number of candidate sets, running time, memory consumption, and scalability.

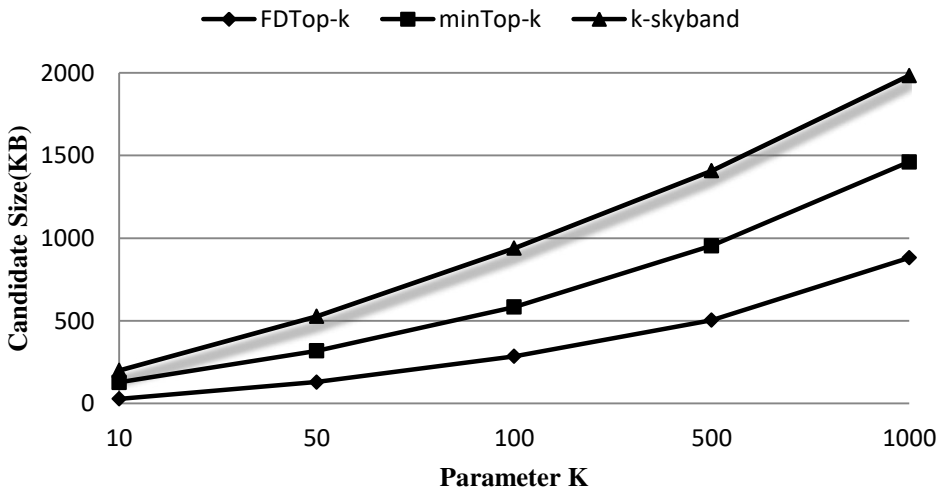

Figure 3 Parameter K and Candidate Size

Figure 3 shows the effect of parameter k on the candidates size. As can be seen from the figure, the number of candidate sets that FDTop-k needs to maintain is the smallest, followed by minTop-k, and the k-skyband is the largest. The algorithm proposed in this paper only uses a small part of the high-quality candidate set because of the double pruning technique. Therefore, the candidate set size of this algorithm is much smaller than the minTop-k and k-skyband algorithms.

Figure 4 shows the effect of parameter k on memory consumption. As can be seen from the figure, the FDTop-k consumes less memory space because the FDTop-k only needs to maintain a small candidate set object after the two-level pruning technique, and the occupied memory space is smaller. minTop-k needs to maintain all k-skyband

objects, and it needs extra memory space to maintain the lbp pointer. Therefore, FDTop-k has smaller storage space than minTop-k. The k-skyband maintains a superset of k-skyband, so it needs the most storage space.

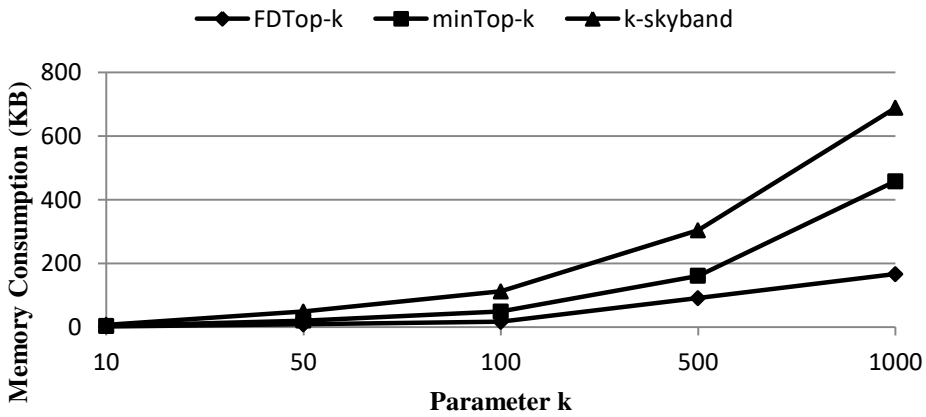

Figure 4 Parameter K and Memory Consumption

Figure 5 shows the effect of parameter k on run time. As can be seen from the figure, the performance of FDTop-k has always been relatively good. As the value of K increases, the running time of minTop-k and k-skyband increases rapidly. Because k-skyband takes O(k) computational cost to remove non-k-skyband objects. The minTop-k algorithm With the increase of k, fewer objects can be discarded in the arrival window.

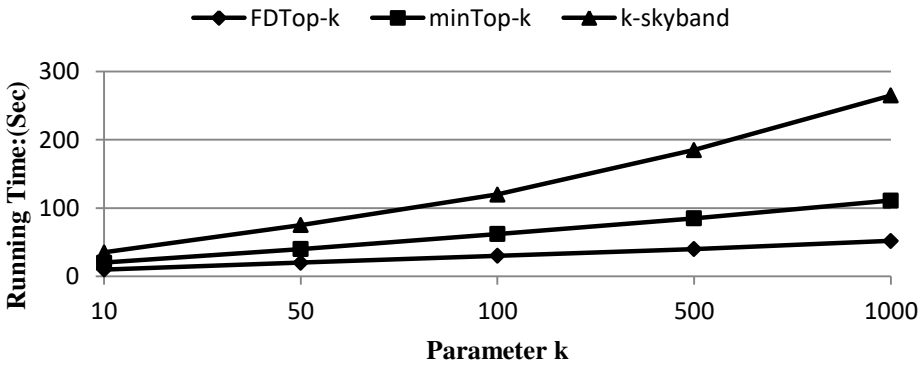

Figure 5 Parameter k and Running Time

Keeping the amount of data unchanged and changing the number of nodes in the cluster, it can be seen from Fig. 6 that as the number of nodes in the cluster increases, the running time tends to decrease as a whole, indicating that the scalability of the method is better.

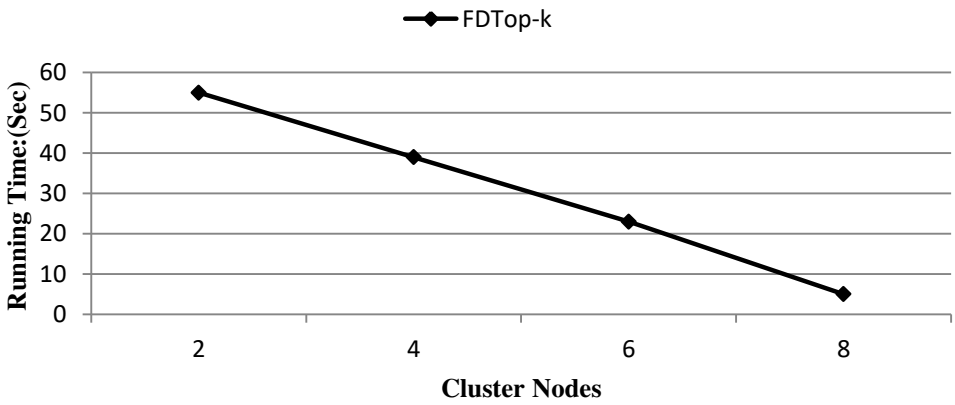

Figure 6 Algorithm Scalability

CONCLUSION

Paper proposed an efficient approach for Top-k dominating queries in data streams, which utilizes a double pruning techniques, it efficiently prunes unnecessary data points and collects the candidate set of the final answer. We evaluated our proposed methods by experiments on real data, and the results show that our proposed methods significantly reduce the communication cost and computation cost compared with the competitors. In the near future, we would like to explore more Top-k dominate models and propose partitioning algorithms accordingly

ACKNOWLEDGEMENTS

This work was Funded by Changsha Municipal Science and Technology Plan Project "Research on Autonomous Knowledge Acquisition Technology for Streaming Big Data (kq1706020)".

REFERENCES

1. Wang Y Z. Network Big Data:Present and Future[J]. Chinese Journal of Computers, 2013, 36(6):1125-1138.
2. Ilyas I F, Beskales G, Soliman M A. A survey of top-k query processing techniques in relational database systems[J]. Acm Computing Surveys, 2008, 40(4):1-58.
3. He Z, Lo E. Answering Why-not Questions on Top-k Queries[C]// IEEE, International Conference on Data Engineering. IEEE, 2012:750-761.
4. Akbarinia R, Pacitti E, Valduriez P. Best Position Algorithms for Top-k Queries[C]// International Conference on Very Large Data Bases. 2007:495-506.
5. Akbarinia R, Pacitti E, Valduriez P. Reducing network traffic in unstructured P2P systems using Top- k, queries[J]. Distributed & Parallel Databases, 2006, 19(2-3):67-86.
6. Fagin R, Lotem A, Naor M. Optimal aggregation algorithms for middleware. Journal of Computer and System Sciences, 2003,66(4):614–656. [doi: 10.1016/S0022-0000(03)00026-6
7. Güntzer U, Balke W, Kießling W. Towards efficient multi-feature queries in heterogeneous environments. In: Proc. of the Int'l Conf. on Information Technology: Coding and Computing (ITCC 2001). Piscataway: IEEE, 2001. 622–628. [doi: 10.1109/ITCC.2001.918866
8. Chang KCC, Hwang SW. Minimal probing: Supporting expensive predicates for top-k queries. In: Proc. of the 2002 ACM SIGMOD Int'l Conf. on Management of Data. New York: ACM Press, 2002. 346–357. [doi: 10.1145/564691.564731
9. Bruno N, Chaudhuri S, Gravano L. Top-K selection queries over relational databases: Mapping strategies and performance evaluation. ACM Trans. on Database Systems, 2002,27(2):153–187. [doi: 10.1145/568518.568519
10. Cao P, Wang Z. Efficient top-K query calculation in distributed networks. In: Proc. of the 23th Annual ACM Symp. on Principles of Distributed Computing. New York: ACM Press, 2004. 206–215. [doi: 10.1145/1011767.1011798
11. Michel S, Triantafillou P, Weikum G. KLEE: A framework for distributed top-k query algorithms. In: Proc. of the 31st Int'l Conf. on Very Large Data Bases. New York: ACM Press, 2005. 637–648.
12. Papadias D, Tao Y, Fu G, et al. Progressive skyline computation in database systems[J]. Acm Transactions on Database Systems, 2005, 30(1):41-82.

13. Man L Y, Mamoulis N. Multi-dimensional top-k dominating queries[M]. Springer-Verlag New York, Inc. 2009.
14. Amagata D, Hara T, Nishio S. Sliding window top-k dominating query processing over distributed data streams[J]. Distributed & Parallel Databases, 2016, 34(4):535-566.
15. Amagata D, Sasaki Y, Hara T, et al. Efficient processing of top-k dominating queries in distributed environments[J]. World Wide Web-internet & Web Information Systems, 2016, 19(4):545-577.
16. Tiakas E, Valkanas G, Papadopoulos A N, et al. Metric-Based Top-k Dominating Queries.[C]// Autonomous Decentralized Systems, 2005. ISADS 2005. Proceedings. IEEE Xplore, 2014:552-556.
17. Shen Z, Cheema M A, Lin X, et al. Efficiently Monitoring Top-k Pairs over Sliding Windows[C]// IEEE, International Conference on Data Engineering. IEEE, 2012:798-809.
18. Jerzak Z, Ziekow H. The DEBS 2015 grand challenge[C]// Acm International Conference. ACM, 2015:266-268.
19. Yang D, Shastri A, Rundensteiner E A, et al. An optimal strategy for monitoring top-k queries in streaming windows[C]// EDBT 2011, International Conference on Extending Database Technology, Uppsala, Sweden, March 21-24, 2011, Proceedings. DBLP, 2011:57-68.