WUJNS
Wuhan University Journal of Natural Sciences

Vol. 11 No. 5 2006 1383-1388

Article ID:1007-1202(2006)05-1383-06

A Filter-Based Uniform Algorithm for Optimizing Top-k Query in Distributed Networks

□ ZHAO Zhibin, YAO Lan, YANG Xiaochun, LI Binyang, YU Ge†
College of Information Science and Engineering, Northeastern University, Shenyang 110004, Liaoning, China

Abstract: In this paper we propose a Filter-based Uniform Algorithm (FbUA) for optimizing top-*k* query in distributed networks, which has been a topic of much recent interest. The basic idea of FbUA is to set a filter at each node to prevent it from sending out the data with little chance to contribute to the top-*k* result. FbUA can gain exact answers to top-*k* query through two phrases of round-trip communications between query station and participant nodes. The experiment results show that FbUA reduces network bandwidth consumption dramatically.

Key words: filter; top-*k*; distributed networks
CLC number: TP 311.133.1

0 Introduction

Top-*k* query in centralized system has achieved great success. It is widely used in search engine to provide users with the most matching contents[1]. However, it is more challenging to process top-*k* query in distributed networks, where data is saved on the nodes that are geographically separate. Though the nodes are connected with communication links, considering networks traffic, it is prohibitively expensive to get data together.

There are two types of top-*k* query in distributed environment. They both have wide ranges of applications. The first is to find the *k* highest ranked objects. In this scenario, there are several objects being monitored by nodes. Its task is to find the *k* objects with the highest aggregation values such as summation[2]. For example, the organizers of 1998 FIFA Soccer World Cup set up 30 mirrored servers with identical copies of the Web content (about 20 000 Web pages) for balancing concurrent access. Now, the administrator poses a query: which ten Web pages are most accessed across all servers? It is a top-10 query, and to answer it we need to sum up the hit counts of each page and pick out the ten pages with the highest summation. In this direction, many algorithms have merged and prove to be efficient in bandwidth consumption, such as TA, TPUT, KLEE and TJA[3-6]. The second is to find the *k* nodes with the highest phenomenon values. In this scenario, there is only one monitored object and each participant node will produce only one physical phenomenon value at a certain time. Let us take FIFA '98 as an example again, the

Received date: 2006-03-25
Foundation item: Supported by the National Natural Science Foundation of China (60503036, 60473073) and Fok Ying Tong Education Foundation (104027)
Biography: ZHAO Zhibin(1975-), male, Ph.D. candidate, research direction: distributed system, sensor network. E-mail: zhaozhibin@ise.neu.edu.cn

† To whom correspondence should be addressed. E-mail: yuge@ise.neu.edu.cn

administrator initiates another query: Among all the thirty servers which ten have the highest load? It is also a top-10 query. However, it is obvious that the second one is different from the first. To answer it, the simplest method is to assemble all the load data to the query station and pick out the ten with the highest values. Applications in wireless sensor networks generally belong to the second scenario[7,8].

FbUA aims at the second scenario. There also have some efficient algorithms in bandwidth consumption in this direction. Deshpande *et al*[9] presents a model-driven data acquisition approach, which suggests using probability models to predict sensor readings. However, it makes the result approximate and is prohibitively expensive in calculation. PROSPECTORPROOF developed by Silberstein *et al*[10] can achieve the exact top-k result, but it needs some proof data, which leads to some extra communications. Wu *et al*[11] exploits the semantics of top-k query and gives a novel filter-based approach, called FILA. It suggests setting a filter at each node. Filter can prevent node from sending out the data with little chance to contribute to top-k result. Unfortunately, FILA may need more than once all-network probing message, which is a significant cost in energy consumption.

FbUA reduces the redundant probing cost by separating nodes into several heaps according to their empirical value ranges and postponing the probing process to the end of the query. Heaps are represented with node boxes, which serve as node containers. Of course, empirical value ranges of nodes may be overlapping or discrete. For example, there are four nodes in distributed networks: A, B, C and D. Their empirical value ranges are [12, 20), [29, 37), [35, 42) and [42, 45), respectively. It is obvious that the overlapping range is [35, 37) and the blank range is [20, 29). How to resolve the problem of overlapping ranges and blank ranges will affect the efficiency of communication. If A reports a reading 36, an optional choice is that the query station probes both B and C for their readings to compare. However, this procedure may be useless for the final result, such that we are running a top-1 query and obviously node D is the final result in this example. Node box may help to leave out the process of probing B and C.

FbUA operates as follows: Initially, query station collects some sampling data and calculates empirical value ranges for participant nodes. Then, it will adjust value ranges to eliminate overlapping ranges and assign node box to each range. The overlapping ranges and the blank ranges also have their own node boxes. The adjusted value range serves as filter and will be installed into participant node. When user poses a top-k query from the query station, the query command will trigger node to collect a real-time phenomenon reading. If the reading is beyond the filter of the node, it will be updated. Otherwise, it will not. After the query station has received all the responses from the nodes, it relocates the nodes according to the data reported. In the end, the query station will probe the node boxes in turn from the highest position for the real nodes and pick out the k firstcomers. They are the elements of exact top-k result that we are searching for.

1 Problem Formulation and Key Definitions

This paper aims at finding the k nodes with the highest values. Consider a distributed network as Fig. 1, the query station is actually an administrator node, and it has the duty to initiate a top-k query, collect data, relocate nodes and produce final result.

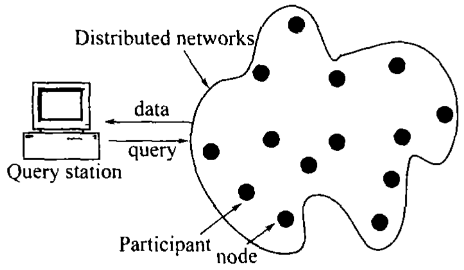


Fig. 1 The topology of distributed networks

Assume that there are n participant nodes in a distributed network N, i. e., $N = \{ p_1, p_2, \cdots, p_n \}$. Each node measures a local physical reading v_{p_i} at a fixed sampling rate, such as the CPU utilization ratio of a server or the on-line user number of a Website. Without loss of generality, we consider that our objective is to find an ordered node set $R = \{ p_1, p_2, \cdots, p_k \}$, where $\forall i < j, v_{p_i} \geq v_{p_j}$ and $\forall p_l \neq p_i (i = 1, 2, \cdots, k), v_{p_l} \leq v_{p_i}$. For discussion, we have the following definitions.

Value Range of Node p_i If having enough samples, we can find that in some applications most readings from a node mainly fall into a given value range. We define r_i for p_i, which means that most readings produced by p_i are within r_i. Let l_i be the lower boundary and u_i be the upper boundary of r_i. Thus, can be represented as $[l_i, u_i)$.

Overlapping Range For $\forall r_i, r_j$ in neighbor, if $l_i < u_j$, then r_i and r_j overlap with each other. See Fig. 2, $[l_i, j_j)$ is the overlapping range and represented as $r_i{}'$.

Blank Range As depicted in Fig. 2, we define $[u_k, l_j)$ as a blank range, which means that none of the nodes take $[u_k, l_j)$ as its empirical value range.

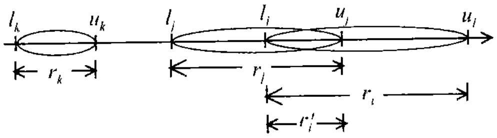


Fig. 2 The overlapping range and the blank range

Filter Filter is set on node and actually a value range. Initially, the query station collects some samples from all nodes and calculates their value ranges. Then the query station adjusts the value ranges and eliminates the overlapping parts. The adjusted value ranges are the filters and installed into the nodes. How to eliminate the overlapping part depends on its size. If two ranges overlap much, then we can merge them into one or assign the overlapping part to one of the two ranges. Otherwise, we can shorten the original two value ranges and regard the overlapping part as an independent range. Let us take the situation in Fig. 2 as an example again, after adjustment, r_i and r_j are shortened from $[l_i, u_i)$ into $[u_j, u_i)$, $[l_j, u_j)$ into $[l_j, l_i)$, respectively, see Fig. 3. $[u_j, u_i)$ and $[l_j, l_i)$ are the filters and will be installed into p_i and p_j.

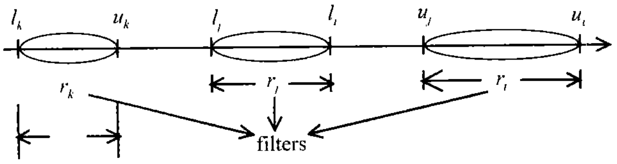


Fig. 3 Value ranges are adjusted into filters

Node Box(nb) Node box may help to reduce the probing cost in some situations. Node Box is a node container and resides only in the memory of the query station. Its data structure is NodeBoxID{(NodeID$_1$, Value), (NodeID$_2$, Value), ⋯, (NodeID$_k$, Value)}. Each value range is assigned with a node box. Node box saves the information of the nodes which have the same value range. Since several nodes may have the same value range, node box must have the ability to contain more than one node. Node boxes assigned to the overlapping and the blank ranges are set to null in the beginning. Fig. 4 is a sketch map of node box. Node p_i is saved in node box nb$_i$, and more than one node are saved in nb$_k$.

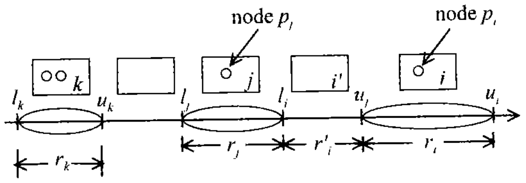


Fig. 4 Node boxes are assigned to the corresponding node

2 The FbUA Algorithmic Framework

The algorithm of FbUA aims at minimizing the communication volume of top-k query in distributed networks. The basic idea is to avoid transmitting data with little chance to enter the top-k result. Fig. 5 is the sequence chart of FbUA Algorithm. The exact top-k result will be achieved if FbUA executes the following steps strictly.

2.1 The Initialization Phrase

There needs some preparations before initiating FbUA. First of all, the query station collects a few sampling data from all the nodes to calculate filters. Filters are sent to all the participant nodes. Since it depends on the application background and has nothing to do with FbUA, we omit it here. Then, the query station assigns and initializes node boxes for each value range. The node boxes for the overlapping and blank ranges are all set to null. Node box is set in order according to their corresponding ranges.

Without any doubt, the initialization work will bring on some costs. However, once for ado, the ordered node boxes sequence may be stable and can be used for a long period of time.

2.2 The Data Acquisition Phrase

Now, user poses a top-k query from the query station. As soon as the participant node p_i receives the query command, it will collect a real-time phenomenon value v_{p_i}. p_i must determine whether to report v_{p_i} back.

If $v_{p_i} \in r_i$, then no updates are needed; Otherwise, the participant node will report (p_i, v_{p_i}) back to the query station.

It means that if p_i makes a response, v_{p_i} must be unusual, and if p_i keeps silent, p_i must be within its filter range. Thus, we can estimate v_{p_i} and shelter a lot of communication useless. In the end of this phrase, the query station will receive all the unusual data reported by nodes.

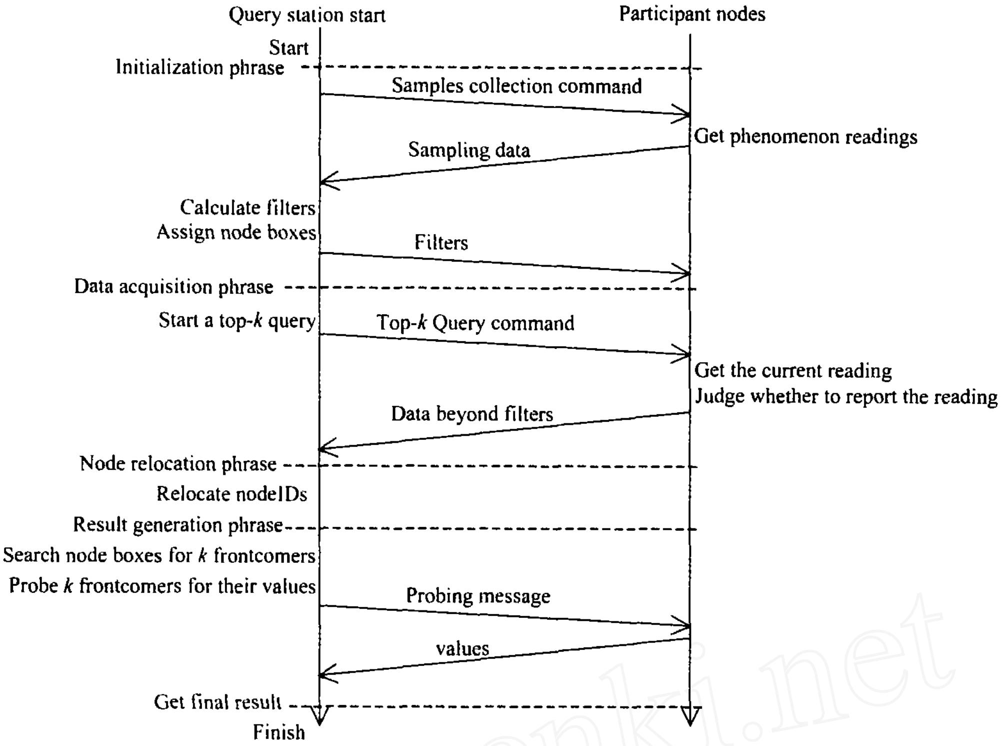


Fig. 5 The sequence chart of FbUA

2.3 The Node Relocation Phrase

After the query station receives all the reported data, it will relocate nodes among the node boxes according to the reported data. For example, if v_{p_i} from node p_i goes beyond r_i and falls into r_j, then the query station will pick p_i out of nb_i and throw it into nb_j, see Fig. 6. Similarly, node p_k is moved from nb_k to $\text{nb}_{i'}$ for that its value falls into $[l_i, u_j)$.

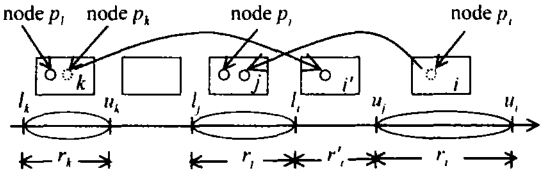


Fig. 6 Node p_i is moved out from nb_i and into nb_j

In the end of this phrase, some boxes will be changed into null, while others will be added more. Let us take Fig. 6 as an example, since node p_i is moved from nb_to nb_j, nb_i is left null, and nb_j has two nodes now. Note that there is no need to probe node p_j immediately for comparison with p_i, because maybe they both have no chance to enter top-k result. Suppose in the example as depicted in Fig. 6 r_i takes up the highest position and we are running a top-1 query. It is obvious that node p_k is the final answer. Thus, which is the larger between p_i and p_j has no effect on the result.

2.4 The Final Top-k Result Generation Phrase

This is the last phrase of FbUA. The remained task is to search the node boxes in turn from the highest position until we get k nodes. As soon as we get a new nodeID, we should probe it for its value. In some applications this may be unnecessary because the top-k nodeIDs are much more concerned than their values.

It is possible that we get more than k nodes while looking for the k frontcomers. For example, we have got k-1 nodes after searching node box nb_{i+1}, but there are two nodes in the box nb_i which is next to nb_i. Thus, we will get $k+1$ nodes. In this situation we need to probe the nodes in nb_{i+1} and resort the nodes according to their values. The larger will be kept and the smaller will be thrown away.

3 Experimental Result

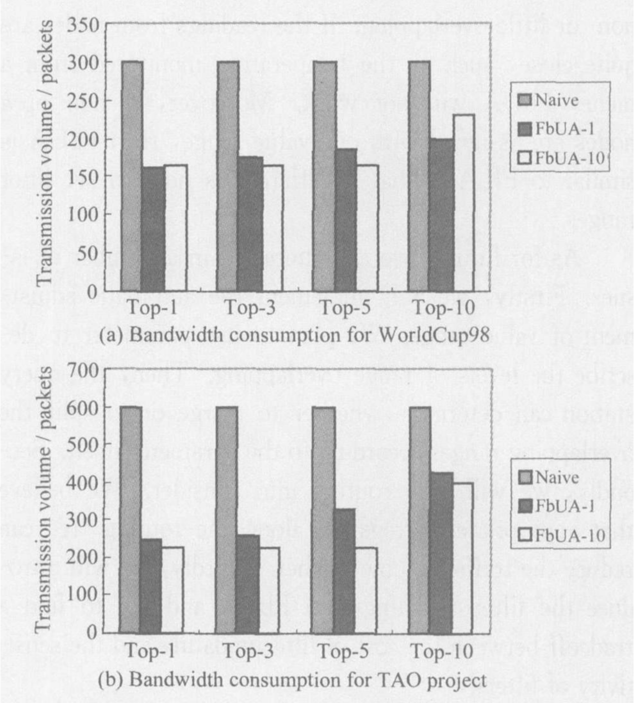
We have made simulative experiments in transmission volume to evaluate the proposed FbUA. We use two real traces as experimental data set. The first comes from FIFA'98 Web sites. It records the hit counts of thirty mirrored servers in a certain period of time, i. e., the load of the servers[12]. The second comes from TAO

(Tropical Atmosphere Ocean) project. The TAO array collects the real-time data from approximately 70 moored ocean buoys in the Tropical Pacific Ocean for improved detection, understanding and prediction of El Nio and La Nia[13]. More details about the characters of data set are described in Table 1.

Table 1 The characters of experiment dataset

Description	n	Sampling data	Testing data	Filters
Hit numbers of FIFA98 servers	30	wc_day65_1 - wc_day66_1	wc_day66_2 - wc_day66_11	4
Sea surface temp. from TAO project	60	1 Dec. 2005 - 10 Dec. 2005	11 Dec. 2005 - 20 Dec. 2005	8

We choose three algorithms for comparison in our experiments. The First is naive algorithm. It asks that all participant nodes send their readings to query station without any filtering on receiving top-k query command, and then the query station calculates the top-k result. The second is FbUA-1, which can be regarded as a variant of FbUA. In FbUA-1, only the latest values reported by the nodes are adopted as the sampling dataset. FbUA-1 is much like FILA[11], but leaving out filter updating, which may lead to a large amount of unexpected transmission cost in some extreme situations where the readings varies a lot as time goes by. The third algorithm is FbUA-10, which means that we select ten couples of readings as the sampling data to determine the filters. In FbUA-10, the filter ranges are relatively stable, but may be some insensitive to the variation of phenomenon reading. Additionally, we leave routing out of consideration, because we are concerned more about the total bytes sent out by all nodes. In order to get rid of the chanciness in testing dataset, we sum up the total number of the packets for ten top-k queries. Fig. 7 (a) is the experiment result based on the dataset of Worldcup98. Fig. 7 (b) is on the dataset of TAO project.



(a) Bandwidth consumption for WorldCup98

(b) Bandwidth consumption for TAO project

Fig. 7 Bandwidth consumption comparison result

Fig. 7 (a) shows the experiment result based on WorldCup98. Since the hit count on Web server is affected much by the place where the server locates, it varies dramatically. For example, there comes a football match involving Morocco team, and then the hit count on the minored server which lies in Africa will increase a lot. The scenarios above lead that there are more values violating their filters and needed reporting back. Generally, in the applications as WorldCup98 FbUA can save about a half of transmission volume than naive algorithm. Fig. 7 (b) is the result of the experiment based on TAO project dataset. The node value varies little because sea surface temperature is much more stable, especially for the ocean around Equator. In this scenario, FbUA can save about 60 percent bandwidth than naive algorithm. We can see in Fig. 7 that the transmission volume of FbUA-1 and FbUA-10 are very close, but FbUA-10 can reduce the cost of filter updating that frequently happens in FbUA-1.

The benefit that we gain from FbUA is affected by many factors, such as n, k and data distribution. Theoretically, the bigger n is and the smaller k is, the more FbUA will benefits. As soon as data distribution is concerned, it determines how to assign filters and how many nodes in each node box on the average. Data distribution depends on background applications. In our experiment, the data what ever it comes from WorldCup98 or TAO project both follow even distribution. However, FbUA will benefit more from the data with normal distribution.

4 Conclusion

This paper introduces a filter-based uniform algorithm for top-k query optimization named FbUA. Its basic idea is to set filters on participant nodes to shelter communication that is useless to top-*k* query. There are two phrases of round-trip communication in all in FbUA. The nodes with the values belongs to the same value range are saved in temporarily node box. FbUA will not probe any node until the top-*k* elements have been identified, which can further reduce the redundant communication. Experimental result shows that FbUA is suitable for the scenario where the value ranges of nodes are stable and with non- or little-overlapping. If the readings from nodes are quite close, such as the temperature monitored from a niche, FbUA will not work. Moreover, if non-top-*k* nodes are assigned with one value range, then FbUA is similar to FILA. What is different is how to set filter ranges.

As for future, we are interested in a number of issues. Firstly, we will implement the automatic adjustment of value range. We plan to use parameter to describe the levels of range overlapping. Then, the query station can determine whether to merge or to split the overlapping ranges according to the parameter itself. Secondly, we will take routing into consider. We believe that appropriate aggregation along the routing tree can reduce the traffic volume further. Thirdly, we will introduce the filter updating into FbUA and try to find a tradeoff between the cost of filter updating and the sensitivity of filters.

References

[1] Hristidis V, Gravano L, Papakonstantinou Y. Efficient IR-Style Keyword Search over Relational Databases [C]// *Proceeding of the 29th Intl Conf on Very Large Data Bases*. Berlin, Germany, Nov, 2003: 850-861.

[2] Babcock B, Olston C. Distributed Top-*k* Monitoring[C]// *Proceeding of the 2003 ACM SIGMOD Intl Conf on Management of Data*. San Diego, California, USA, Jan, 2003:28-39.

[3] Fagin R, Lotem A, Naor M. Optimal Aggregation Algorithms for Middleware [C]// *Proceeding of the Twentieth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*. Santa Barbara, CA, USA, Feb, 2001: 102-113.

[4] Cao P, Wang Z. Efficient Top-*k* Query Calculation in Distributed Networks [C]// *Proceeding of the 23rd Annual ACM Symposium on Principles of Distributed Computing*. Newfoundland, Canada, Jan, 2004: 206-215.

[5] Michel S, Triantafillou P, Weikum G. KLEE:A Framework for Distributed Top-*k* Query Algorithms [C]// *Proceeding of the 31st Conference in the Series of the Very Large Data Bases*. Trondheim, Norway, Oct, 2005: 637-648.

[6] Zeinalipour-Yazti D, Vagena Z, Gunopulos D, *et al*. The Threshold Join Algorithm for Top-*k* Queries in Distributed Sensor Networks [C]// *Proceeding of the 2nd International Workshop on Data Management for Sensor Networks*. Trondheim, Norway, Oct, 2005: 61- 66.

[7] Mainwaring A, Polastre J, Szewczyk R, *et al*. Wireless Sensor Networks for Habitat Monitoring [C]// *Proceeding of ACM International Workshop on Wireless Sensor Networks and Applications*. Atlanta, Georgia, USA, July, 2002.

[8] Akyildiz I F, Su W, Sankarasubramaniam Y, *et al*. A Survey on Sensor Networks [J]. *IEEE Communications Magazine*, 2002, (8): 102-114.

[9] Deshpande A, Guestrin C, Madden S, *et al*. Model-Driven Data Acquisition in Sensor Networks [C]// *Proceeding of the 2004 Intl Conf on Very Large Data Bases*, Toronto, Canada, May, 2004: 588-599.

[10] Silberstein A, Braynard R, Ellis C, *et al*. A Sampling-Based Approach to Optimizing Top-*k* Queries in Sensor Network [C]// *Proceeding of the Intl Conf on Data Engineering*. Atlanta, USA, April, 2006.

[11] Wu M, Tang X, Lee W. Monitoring Top-*k* Query in Wireless Sensor Networks [C]// *Proceeding of the 22nd International Conference on Data Engineering*. Atlanta, Georgia, USA, April, 2006.

[12] ACM SIGCOMM. WorldCup98 [EB/OL]. [2005-06-01]. *http://ita.ee.lbl.gov/html/contrib/WorldCupup.html*.

[13] Pacific Marine Environmental Tropical Laboratory. Atomosphere Ocean Project [EB/OL]. [2005-06-12]. *http://www.pmel.noaa.gov/tao/data_deliv*.

□