Query Aggregation in Wireless Sensor Networks

Min Meng[1], Jie Yang[1], Hui Xu[1], Byeong-Soo Jeong[2], Young-Koo Lee[2]
and Sungyoung Lee[1]
Department of Computer Engineering, Kyung Hee University
E-mail: [1]{mengmin,yangjie,xuhui,sylee}@oslab.khu.ac.kr, [2]{jeong,yklee}@khu.ac.kr

Xiaobo Fan
School of information, Renmin University of China
E-mail: xbfan@ruc.edu.cn

Abstract

Wireless sensor networks have been used more and more widely with the developments of related techniques in telecommunication and computer sciences. While sensor nodes in wireless sensor networks have very limited memory spaces and power. In this paper, we propose a new query aggregation method to preprocess the query predicates. The size of the relational table can be further reduced using the aggregated query predicates. The reduced relational table can be stored on the sensor nodes. This may cause false positive but definitely no false negative. When the required data are sent back to the sink, we can do join operation between the real query predicate and the relational table again, and eliminate the redundant data.

Keyword*: Query aggregation, wireless sensor network*

1. Introduction

Wireless sensor networks have been used more and more widely and closely in our everyday life with the developments of related techniques in telecommunication and computer sciences. The application examples include habitat monitoring, environmental monitoring and ubiquitous healthcare system. The main roles of the sensor nodes are sensing, communication and computation. Sensor nodes are usually made up of four basic components, sensing unit, processing unit, transceiver unit, and power unit. Sensor nodes may also have some additional application-dependent components, like location finding system, power generator and mobilizer. While in general the sensor nodes in wireless sensor networks have very limited memory spaces and power. So the relational table which is used to record the filter conditions may be too large to be stored on every sensor nodes. Our idea is that first we eliminate part of the redundant data in the relational table. Then it is suitable to be stored on the sensor nodes. We can preprocess the query 1predicates, using the query predicates to cut the size of the relational table.

[1] This research was supported by the MIC(Ministry of Information and Communication), Korea, Under the ITFSIP(IT Foreign Specialist Inviting Program) supervised by the IITA(Institute of Information Technology Advancement).

In this paper, we propose a new query aggregation method to preprocess the query predicates. The size of the relational table can be further reduced using the aggregated query predicates. The reduced relational table can be stored on the sensor nodes. This may cause false positive but definitely no false negative. False positive means the results may include the redundant data which we don't want but the results include all the possible answers. False negative means we may eliminate some useful data. When the required data are sent back to the sink, we can do join operation between the real query predicate and the relational table again, and eliminate the redundant data.

The remainder of the paper is organized as follows. In section 2 we introduce the related works. In section 3, we propose the method to preprocess the query conditions. In section 4 we conclude the paper and discuss future works**.**

2. Related works

Lots of methods and researches are dealing with energy efficient issues and to prolong the lifetime of the wireless sensor networks [4-15]. Data management in wireless sensor network is an increasing research topic [2], [4]. Saving energy in data transmission is a major topic in wireless sensor network and lots of routing protocols are proposed [8], [9], [10], [14]. In TinyDB[3], the sensor nodes are formed into a routing tree. Each node in the network chooses one node as its parent, which is one network hop closer to the root than it is. When a node answers query, it sends to its parent, parent forward it to its parent, until the data reach the root. When the query comes, it is distributed to the sensor nodes. Every time the node produces data tuples, the data tuples are filtered by expression and the nodes pass results up the routing tree, aggregating with the intermediate nodes if necessary.

In naïve join algorithm, the sensor nodes send all tuples to the root; then perform join at the root, as shown in figure 1. A routing tree is formed first, 1 to 7 are sensor nodes. The relational table is stored at the root. When a new tuple x is created at sensor node 7, it is sent to the parent of node 7, i.e. 5, then passed to the root. At last the new tuple joins with the relational table.

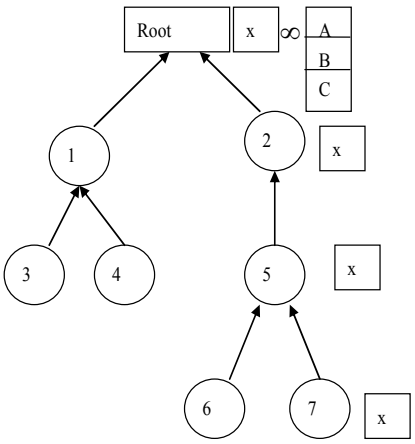


Figure 1. Naïve join algorithm

In ideal join algorithm, the whole join table is sent to all the sensor nodes. When a new data tuple is created, perform join at the sensor node. But the severe node memory constraints limit the size of the relational table. As shown in figure 2. The relational table is stored at every sensor node from 1 to 7. When a new data tuple is created at node 7, it can immediately join with the relational table. Due to the limited memory spaces of the sensor nodes, this method is hard to be realized.

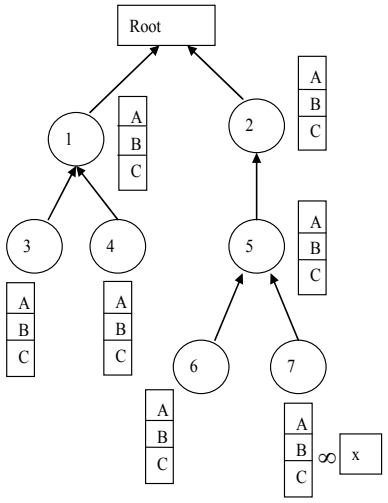


Figure 2. Ideal join algorithm

In REED algorithm [2], three distinct phases are included in this in-network algorithm: group formation, table distribution, and query processing. The sensor nodes are formed into groups. In each group, every sensor node stores portion of the relational table. The sensor data tuples are sent to every member of the group, as shown in figure 3. Two groups are formed. One group has members 1, 3, 4. The other group has members 5, 6, 7. Sensor nodes 1, 3, 4 of group one store part of the relational table respectively. Sensor nodes 5, 6, 7 of group two store part of the relational table respectively. When a new data tuple x of sensor node 7 is created, it do join with every part of the relational table in this group.

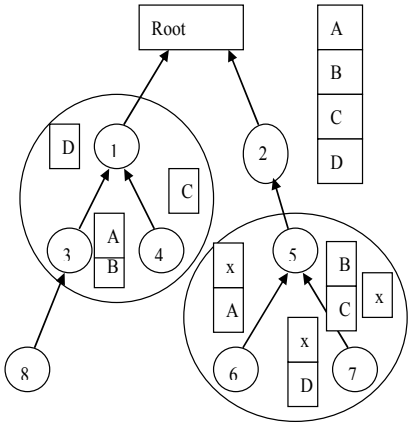


Figure 3. REED algorithm

Based on these former works, we can first eliminate the redundant data so that we can reduce the size of the relational table. We can process the query predicates or the query conditions first.

3. Query aggregation

3.1 Assumptions

In this section, we discuss the detail of how to preprocess the query predicates.
We make the following assumptions about the sensor network:

1) A routing tree is established in the sensor network.
2) The root is the base station, and the base station assembles partial results from nodes in the network, completes query processing and displays results to the user.
3) Multiple applications can simultaneously send a number of queries to the sensor network.

3.2 Query aggregation general model

In this part, we generalize the query aggregation model.

1) Project ($\prod$) the attributes the users are interested in from the relational table.

Combine all the related queries. Here l means lower boundary, u means upper boundary.

Query 1
l1<Table.attribute1<u1

Query 2
l2<Table.attribute1<u2
…
Query n
ln<Table.attribute1<un

Combine all the queries

2) (l1,u1) $\cup$ (l2,u2) $\cup$ … $\cup$ (ln,un)=(l,u)

3) Select (σ) the values by (l,u) including all data from the queries

4) Store the reduced sized relational table on the sensor nodes

The following procedures should be the same as REED algorithm.

3.3 Query aggregation example

Suppose we have a relational table T, as shown in table 1, a, b are attributes of tableT.

Table 1. relational table T

a	b
1	25
2	29
3	23
4	21
5	28
6	17
7	19
8	20
9	24
10	29
11	17
12	26

There are two queries coming at the same time. Query 1 requires to return a's values between 1 and 5. Query 2 requires to return a's values between 4 and 10. Such two queries might look like:

Query 1 Select a
from T
where 1<T.a<5
Query 2 Select a
from T
where 4<T.a<10

We can combine these two queries first: 1<T.a<10. We use relational algebra to reduce the size of the relational table.

$$\sigma_{1<T.a<10}[\Pi_a(T)]$$

Then the relational table T becomes the table T' shown in table 2 with only attribute a and a's value from 2 to 9.

Table 2. relational table T'

a
2
3
4
5
6
7
8
9

Compare the sizes of table T and T', apparently the size of T' is largely reduced and much smaller than T. We can store this table T' on the sensor nodes or we can further split the table T' to store the portions on the sensor nodes according to REED algorithm.

Using this method may cause false positive. When the data is sent to the base station, we should do join operation again, and send the real results to the clients.

4. Simulation

In this section, we describe the flow chart of the query aggregation method.

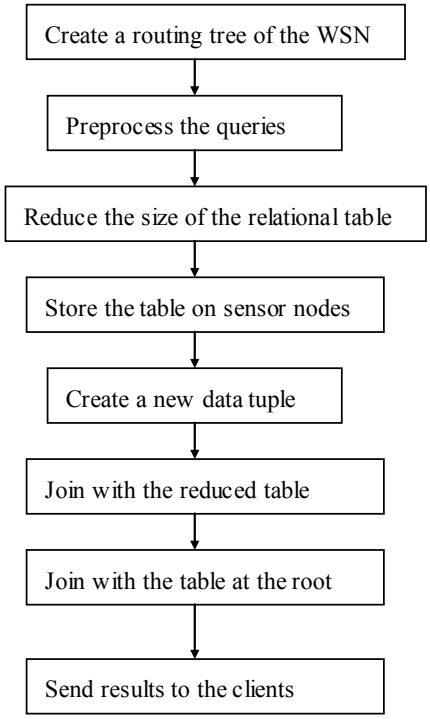


Figure 4. Flow chart of the query aggregation method

5. Conclusions and future works

The former works on query processing in sensor networks only deal with the split of the relational table. Our method query aggregation is to eliminate the redundant data from the relational table by preprocessing the queries. The size of the table can be reduced dramatically. Then we can split the table or store the whole table on the sensor nodes. But this method may cause false positive. After the data is sent to the base station, another join should be done to get the final results.

We will concrete the model of the method in the near future and experiment about the performance.

6. References

[1] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, “A survey on sensor networks,” IEEE Communication Magazine, Vol.40, No.8, 2002, pp.102-114.

[2] D. J. Abadi, S. Madden, and W. Linder, "REED: Robust, Efficient Filtering and Event Detection in Sensor Networks", Proc. of the 31st VLDB conference, Trondheim, Norway, 2005, pp. 769-780.

[3] S. Madden, M. Franklin, J. Hellerstein, and W. Hong. The design of an acquisitional query processor for sensor networks. In SIGMOD, 2003.

[4] D. Abadi, et al. REED: Robust, Efficient Filtering and Event Detection in Sensor Networks. In technical report, MIT-LCS-TR-939, 2004.

[5] Daniel Abadi, et al. An Integration Framework for Sensor Networks and Data Stream Management Systems..In Proc. of VLDB, 2004.

[6] Daniel Abadi, et al. The Design of the Borealis Stream Processing Engine.In Proc. of CIDR, 2005.

[7] Philip A. Bernstein, Dah-Ming W. Chiu, Using Semi-Joins to Solve Relational Queries. Journal of the ACM, 28(1):25-40, 1981.

[8] W. Yu, T. Le, Dong Xuan and W. Zhao.: Query Aggregation for Providing Efficient Data Services in Sensor Networks, in Proc. of IEEE Mobile Sensor and Ad-hoc and Sensor Systems (MASS), October 2004

[9] Ian D. Chakeres and Elizabeth M. Belding-Royer.: AODV Routing Protocol Implementation Design. WWAN, 2004

[10] C. Intanagonwisat, R. Govindan, and D. Estrin.: Directed Diffusion: A Scalable and Robust Communication, In Proceedings of ACM MobileCom’00, August 2000

[11] X. Li, Y. J. Kim, R. Govindan, and W. Hong.: Multi-dimensional Range Queries in Sensor Network, In Proceedings of ACM SenSys’03, Nov., 2003

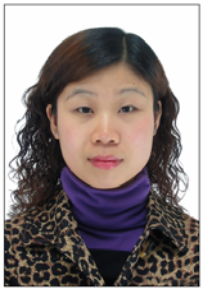
[12] H. O. Tan, and I. Korpeoglu.: Power Efficient Data Gathering and Aggregation in Wireless Senor Network, In Proceedings of ACM SIGMOD’03, Special Section on Sensor Network Technology and Sensor Data Management, 2003

[13] N.Sadagopan, B.Krishamachari, and A.Helmy.: Active Query Forwarding in Sensor Networks, Accepted to Journal of Ad-hoc Networks, ELSEVIER, August, 2003

[14] Samuel Madden, Michael J. Franklin, Joseph M.Hellerstein, and Wei Hong.: TAG: a Tiny Aggregation Service for Ad-Hoc Sensor Networks. OSDI, 2002

[15] Yong Yao and Johannes Gehrke.: Query Processing for Sensor Networks. CIDR, 2003

Authors



Min Meng

Received a B.S. degree in Computer Management from Shandong Economic University, China, 2001 and M.S. degree in Technological Economics and Management from North China Electric Power University, China, 2004. A Ph D. Candidate in Computer Engineering from Kyung Hee University, Korea since 2005.

Byeong-Soo Jeong

Received a BS degree in Computer Engineering from Seoul National University, Korea, 1983 and MS degree in Computer Science from Korea Advanced Institute of Science & Technology, Korea, 1985 and Ph.D in Computer Science, Georgia Institute of Technology, USA,1995. From 1996, he is an associate Professor in Computer Engineering Major, Kyung Hee University. From 1995 - 1996, he was a research scientist in College of Computing, Georgia Institute of Technology. From 1985 - 1989, he was a senior researcher in Computer Communications Lab, Data Communication (DACOM).