www.s2is.org
International Journal on Smart Sensing and Intelligent Systems
ISSN 1178-5608

BI-FILTERED FORWARDING: A QUASI-OPTIMAL ROUTING ALGORITHM FOR QUERY DELIVERY IN WIRELESS SENSOR NETWORKS

Junhu Zhang[1], Xiujuan Zhu[1], Hui Peng[2]
1 College of Computer Engineering, Qingdao Technological University
Qingdao, China, Emails: zhangjunhu@gmail.com; zhuxiujuan789@126.com
2 School of Information, Renmin University of China
Beijing, China, Email: pphh666@163.com

Submitted: Sep. 25, 2012 ***Accepted: May 3, 2013*** ***Published: June 5, 2013***

Abstract- A quasi-optimal query propagation algorithm Bi-Filtered Forwarding (BFF) for quickly routing a query throughout a wireless sensor network is proposed in this paper. BFF is implemented in a limited flooding manner for guaranteeing quick query propagation and low message consumption in wireless sensor networks. The experimental results show that in comparison with the flooding algorithm, BFF can greatly reduce the redundant message consumption during the procedure of real time query propagation throughout a wireless sensor network.

Index terms*:* Real time routing, distributed query processing, query propagation, constrained flooding, wireless sensor networks.

I. INTRODUCTION

A wireless sensor network is a distributed system consisting of mobile routers connected by wireless links, where the sensor nodes have the characteristics of limited energy and confined communication bandwidth. Thus it is significant for a wireless sensor network to have the capability of quickly routing the query submitted on a node to all the other nodes (which possibly know or partly know the results of the query) while keeping the message consumption caused by the routing as little as possible for the purpose of the energy saving.

Nowadays, the flooding algorithm is widely used for the query routing described above because of its distinctive advantages such as high reliability, high node coverage rate and low response delay. However, the flooding algorithm always introduces too much message consumption due to its redundant message forwarding between nodes in the network.

To save the message consumption of the routing while still maintaining the time constraints to the routing, we manage to propose the Bi-Filtered Forwarding (BFF), a kind of limited flooding algorithm for routing the query submitted by any node to all the other nodes in a wireless sensor network, with real time routing and less message forwarding traffic caused.

The organization of the paper is organized as follows. After a general introduction of state-of-the-art solutions on how to effectively and efficiently deliver a query throughout a wireless sensor network in section II, the design inspiration of the Bi-Filtered Forwarding (BFF) is introduced and discussed in section III. In section IV, the BFF algorithm implemented by the necessary local data structures, the related message structures and the main event-driven procedures is proposed. The various kinds of experiments are designed and the results are analyzed in section V. The paper is concluded in section VI.

II. RELATED WORK

Nowadays, applications built on wireless sensor networks always need distributed query processing [1, 2], while the distributed query processing demands real time routing and low energy consumption [3, 4]. In other words, there are two new challenges for the route protocol design: one is routing with low response delay, the other is routing with low energy consumption.

On the one hand, for the challenge of routing with low response delay, tree-based routing methods are proposed. For example, Liang et al. [5] proposed a top-k query evaluation algorithm with low routing response delay for wireless sensor networks, where a routing tree is firstly found and then the query evaluation is done on the tree for answering a top-k query committed on a sensor node. Pervin et al. [6] proposed a spanning tree construction method that reduces the routing response delay as well as energy consumption in query execution. By virtue of the optimal operator placement with minimum communication cost overhead, Chatzimilioudis et al. [7] constructed and optimized a query routing tree for aggregate query evaluation. Similarly as the above tree-based routing methods, Kang et al. [8] proposed an in-network query method, IQM, which divides a query region into several cells according to the distribution of sensor nodes and builds a quad-tree, and then processes aggregate queries in parallel for each cell region according to routing.

For real time routing without the construction of any kinds of query routing tree, Ahmed et al. [9] proposed a real time routing protocol where each node made the forwarding decision by taking into account of the link quality, packet delay time and the remaining power of next hop sensor nodes. Gao et al. [10] introduced the routing optimization algorithm with the help of the ant colony algorithm. Similarly, Kim et al. [11] proposed a multi-path routing for QoS aware mobile ad-hoc networks based on the ant colony optimization technique. Through the routing protocol, data packets are adaptively distributed through the established paths while maintaining an acceptable level of QoS requirement. Specially, the protocol has the adaptability to real time traffic conditions of an ad-hoc network.

As another main class of real time routing schemes, the flooding-based routing is good at delivering the query message throughout the network with little routing response delay. For example, Sausen et al. [12] made use of a broadcast backbone for real time routing. To saves the message consumption for query propagation, the backbone is carefully chosen for energy efficiency. Yen et al. [13] proposed a routing algorithm with adaptive path and limited flooding for mobile ad hoc networks. Similarly, Zhang et al. [14] proposed another flooding-based kind of routing algorithm, ripple routing, for saving the routing response time and the message consumption for route discovery.

On the other hand, for the challenge of low energy consumption for route discovery in wireless sensor networks, energy-efficient routing protocols have also been widely studied under the environment of wireless sensor networks [15, 16]. For example, to find an efficient way to reduce the energy consumption for route discovery and consequently prolong the network lifetime, Yin et al. [17] proposed a centralized routing method called LEACH-DE which outperformed LEACH and LEACH-C in the aspects of reducing the overall energy consumption and prolong the network lifetime.

Different from the idea in [17], Mohapatra et al. [18] developed a joint policy involving routing and node replacement decisions for the energy saving. Riva and Finochietto [19] proposed a probabilistic pheromone-based in-network processing scheme, PhINP, to monitor information on wireless sensor networks in an energy-efficient way. Actually PhINP took advantage of both query-based in-network filtering to select nodes to answer, and a pheromone-based strategy to direct queries towards nodes with relevant readings. Similarly, Ye et al. [20] developed an algorithm for distributed processing of probabilistic top-k queries in cluster-based wireless sensor networks. By virtue of inter-cluster query processing with bounded rounds of communications, the algorithm achieved the goal of reducing the data transmissions and consequently the energy consumption for the query routing.

For energy efficient routing which overcomes the hole problem to routing in wireless sensor networks, Chen et al. [21] proposed a so-called Mobicast routing protocol for data collection in underwater sensor networks having the characteristics of low communication bandwidth, large propagation delay, and ocean current. Preethi and Sumathi [22] proposed an on-demand routing with void avoidance (ODVA) routing protocol to overcome the hole problem and ensure the load balancing. ODVA achieved the goal of a high packet delivery ratio, a long network lifetime and efficient usage of energy in comparison with other prevalent schemes. As another example for overcoming the hole problem, Cheng et al. [23] proposed a highly topology adaptable ad hoc routing protocol with complementary preemptive link breaking avoidance and path shortening mechanisms. To be noticed, in comparison with the above hole-avoidance routing protocol, broadcast-based routing can handle the hole problem without introducing extra message consumption in wireless sensor networks.

Based on the above stat-of-the-art research on the routing algorithms for query propagation in wireless sensor networks, we manage to propose the Bi-Filtered Forwarding (BFF), for quickly routing the query submitted by any node to all the other nodes in a wireless sensor network, with low routing response delay and low message consumption as well.

III. THE INSPIRATION OF THE BI-FILTERED FORWARDING ALGORITHM

Most of the applications built on a wireless sensor network demand an effective and efficient query propagation algorithm for transmitting a query message hop by hop from one specified node to all the other nodes so that each node of the network can be queried at least once in the network. To achieve the above goal, flooding-based routing algorithms are always applied because of their distinctive advantages such as high routing reliability, high node coverage rate and low delay of query delivery. Actually, under flooding-based routing algorithms, nodes forward the query message unconditionally in a parallel manner, and the query message can be delivered quickly hop by hop throughout the whole network.

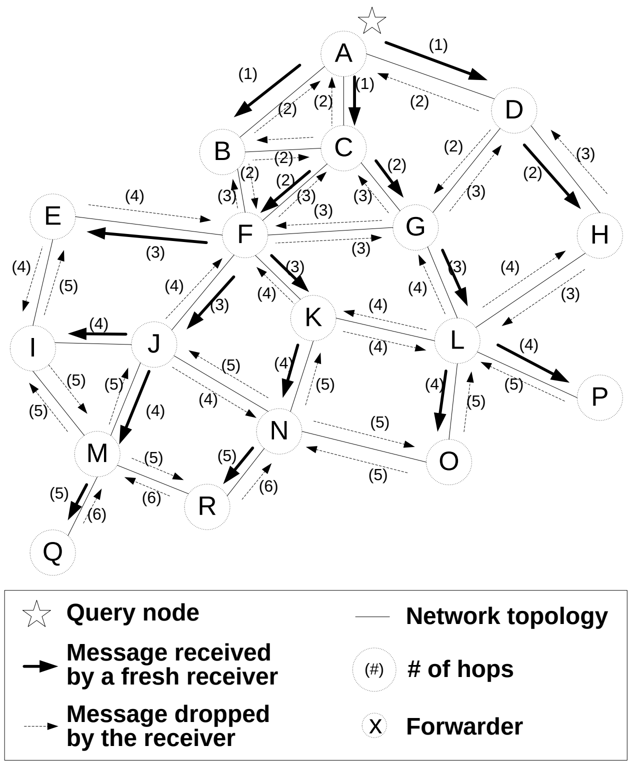
A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P
Q
R
Query node
Message received by a fresh receiver
Message dropped by the receiver
Network topology
# of hops
Forwarder

Figure 1. The procedure of a query propagation directed by a flooding-based routing algorithm

However, flooding-based routing algorithms always introduce redundant message consumption for the query propagation throughout the whole network. As an example, figure 1 shows how a flooding-based routing algorithm helps a query node A propagate a query to all the other nodes in a wireless sensor network. Actually, under the control of the flooding-based routing algorithm, A starts the query propagation by broadcasting the query message to A’s one-hop neighbors. Then the node who receives the query message forwards the message to all its one-hop neighbors. The above message forwarding continues until all the nodes have received the query message once. As seen from the figure 1, the message forwarding covers the whole network topology within just 6 hops of message forwarding and the messages consumption for the query propagation is 18 (because all the 18 nodes are designated as forwarders for message broadcasting).

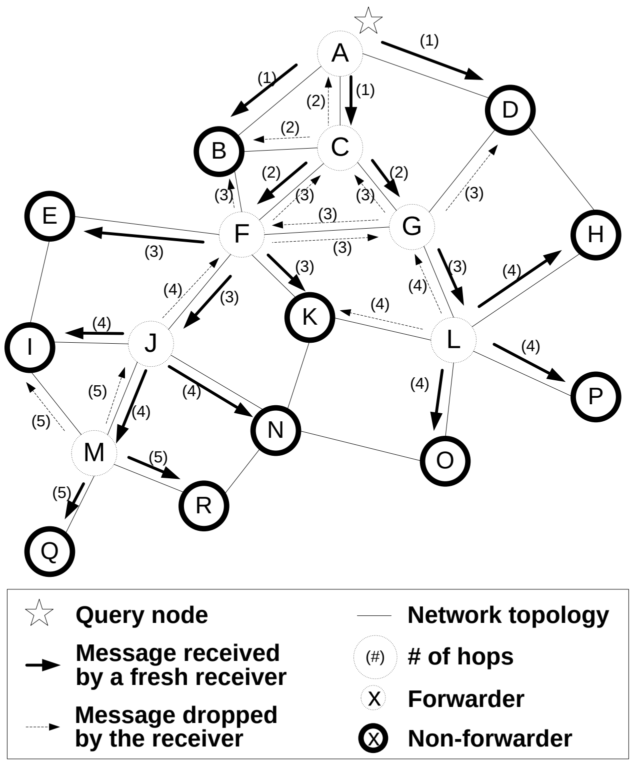

Figure 2. The procedure of a query propagation directed by an optimal routing algorithm

For the optimization of the query propagation shown in figure 1, some of the nodes must be designated to be non-forwarders who will never forward any query messages. As an example shown in figure 2, the optimal routing algorithm only needs 7 forwarders for forwarding the query message to all the other nodes in a wireless sensor network. Actually, for the optimal routing algorithm executed in the network scenario shown in figure 2, the node coverage rate is 100% after just 5 hops of message forwarding, and the message consumption is 7 because only 7 nodes are designated as forwarders for message broadcasting. To be noticed, here the redundant message consumption is 0 (because any 6 forwarders chosen from the network are not able to fulfill the query propagation throughout the whole network), that is to say, at least 7 forwarders are required for the query propagation throughout the whole network.

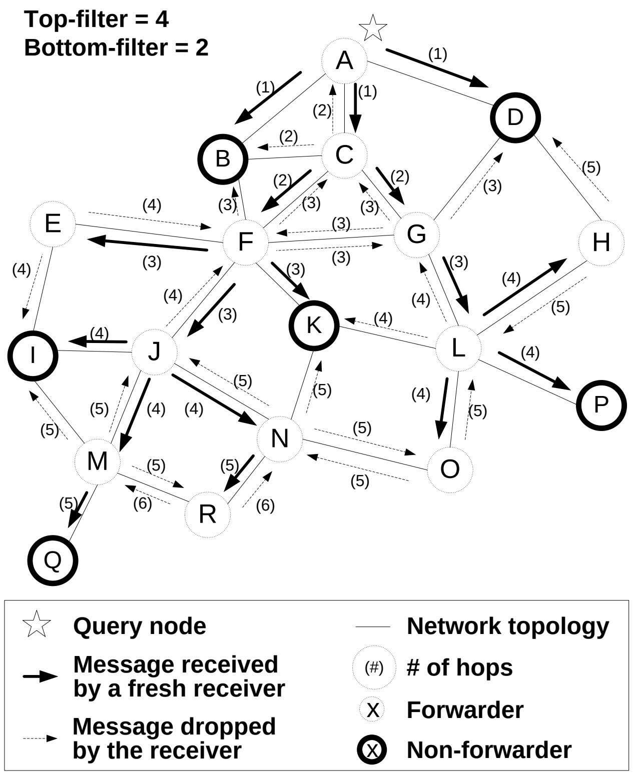

Figure 3. The procedure of a multi-hop query propagation directed by the BFF routing algorithm

Inspired by the optimal routing algorithm addressed above, we propose a constrained-flooding algorithm, called Bi-Filtered Forwarding (BFF), for the distributed resource query processing in a wireless sensor network.

As illustrated in figure 3, BFF begins with the query node A broadcasting a query message. At the first hop of message forwarding when the node B, C, D receives the query message respectively, only the node C is authorized by the two filters and designated as a forwarder (the description of the two filters and the authorization procedure is described in the following section). The node C then, as a forwarder, broadcasts the received message to its one-hop neighbors. Similarly, at the second hop of message forwarding when the “fresh” node F and G (who never receive the message before) receive the query message, they all get the authorization of the two filters and forward the message. At the last (sixth) hop of message forwarding, all the nodes have received the query message once.

To be noticed, in contrast with the optimal routing algorithm mentioned in figure 2, controlled by the proposed BFF routing algorithm, the node coverage rate is 100% after 6 hops of message forwarding, and the message consumption is 12 because 12 forwarders are designated as forwarders for message broadcasting. According to the analysis in figure 2, the redundant message consumption is thus 12 – 7 = 5. That is to say, the proposed BFF routing algorithm consumes 5 extra unnecessary message broadcastings for the query propagation throughout the network. In comparison, the redundant message consumption caused by the flooding-based routing algorithm mentioned in figure 1 is 18 – 7 = 11 which is larger than that caused by the proposed BFF routing algorithm.

IV. THE BI-FILTERED FORWARDING ALGORITHM

a. Introduction of the two filters: the top-filter and the bottom-filter

As mentioned in figure 2, to meet the goal of eliminating redundant message consumption, the optimal routing algorithm has to increase the amount of non-forwarders as much as possible. Considering the non-forwarders in the network scenario shown in figure 2, 63.6% of the non-forwarders' connectivity degree is 3, while only 9.1%, 18.2%, and 9.1% of the non-forwarders' connectivity degree is 1, 2, and 4 respectively.

Therefore, a good routing algorithm for the query propagation can by found by converting most of the forwarders whose connectivity degree is not too high or too low into non-forwarders.

As analyzed above, the bi-filtered forwarding algorithm (BFF) is proposed by introducing two filters, a "top-filter" and a "bottom-filter", for propagating the query message throughout the network. Actually, on the one hand, the so-called "top-filter" is used to guarantee that only the node with high connectivity degree could become a forwarder to route the query message to all the other nodes so as to reduce the message redundancy.

On the other hand, if the query message is only forwarded by the nodes with high connectivity degree, the query message may not be forwarded to all the nodes in a sparse wireless sensor network. Thus to promise 100% node coverage rate, the so-called "bottom-filter" is also introduced into BFF to guarantee that the node with low connectivity degree (at least 2) could also be a forwarder.

Based on the two filters addressed above, BFF can accomplish not only low message redundancy but also 100% node coverage rate in a small amount of message forwarding traffic.

Before implementing the Bi-Filtered Forwarding algorithm (BFF), the message structure and the necessary variables installed on each node must be specified in the following.

b. The message structure used by BFF and the variables installed on each node

Actually, the message used by BFF contains the following five components.

(1)TYPE: the type of the message. “0” means a “Hi” message; “1” means a “Query” message to be propagated throughout the network;

(2) src: the node who sends the message.

(3) sQuery: the query string expressed in SQL. It is invalid if TYPE = 0;

(4) bFilter: the bottom-filter. It is invalid if TYPE = 0;

(5) tFilter: the top-filter. It is invalid if TYPE = 0;

Table 1: The “nbr-hi” procedure is responsible for making the connectivity degree up-to-date

```
Procedure nbr-hi()
Begin
X ← the node who receives the message;
msg ← the message received;
if (msg.TYPE = 0) /* it’s a “Hi” message sent by a neighbor */
   if msg.src ∉ X.nbr[]
            T_NODE* one = new T_NODE;
            one.nid ← msg.src;
            one.timestamp ← now();
            X.nbr ← X.nbr[] + one;
            X.conn ← X.conn + 1;
   else     /* Assume msg.src = X.nbr[i] */
            X.nbr[i].timestamp ← now();
   endif
endif
End
```

Also, the following three local variables have to be maintained up-to-date on each node X.

(1) X.visited, a boolean variable indicating whether or not the node X has been visited by a “Query” message. This variable is set to 0 by default in the boot-up time of each node.

(2) X.conn, an integer variable representing the connectivity degree of the node X. This variable is set to 0 by default in the boot-up time of each node.

(3) X.nbr[], a set contains the current neighbor list. The scheme of the list “nbr” conforms to the “T_NODE” structure <nid, timestamp>, where “nid” refers to the identification of one of X’s neighbors, “timestamp” refers to the last time when the neighbor broadcasts a “Hi” message.
Having the above prerequisites, the BFF routing algorithm can be implemented on each node in a message-driven mode.

c. Event-driven “nbr-hi” procedure installed on each node
Anytime when a node manages to join the ad hoc network, it initiates a “Hi” message with its TYPE = 0, and then broadcasts the message to its one-hop neighbors.
On receiving the “Hi” message, the node “X” executes the “nbr-hi” procedure which is specified in table 1 for updating its neighbor list and its connectivity degree.

d. Event-driven “nbr-check” procedure installed on each node
By calling the “nbr-check” procedure specified in table 2, each node checks its neighbor list at every *t* seconds to see if some of its neighbors should be eliminated.

Table 2: Every *t* seconds, the “nbr-check” procedure is called to eliminate the dead neighbors

```
Procedure nbr-check
Begin
k ← X.conn;
for (i=1 to k)
    if (X.nbr[i].timestamp – now() > t)
        X.nbr[] ← X.nbr[] – X.nbr[i];
        X.conn ← X.conn – 1;
    endif
endfor
End
```

e. Event-driven “bi-filtered-forwarding procedure” installed on each node
Having the connectivity degree maintained by the “hi procedure”, each node can then perform the distributed bi-filtered forwarding procedure at any time. Actually when the query node A initiates a resource query, it broadcasts a resource query message “msg” to its one-hop neighbors. Then

when a node X receives the message "msg", it performs the following "bi-filtered forwarding" procedure as implemented in table 3.

Table 3: Once a node X receives a message, the "bi-filtered forwarding" procedure is called

Procedure bi-filtered-forwarding

```
Begin
X ← the node who receives the message;
msg ← the message received;
if (msg.TYPE = 1) /* it's a "Query" message sent by a neighbor */
   if (X.visited = false)
       X.visited ← TRUE;
       if ( isCandidate(msg.bFilter, msg.tFilter, X) )
           int d1 ← distance( X, msg.src ); int d2  ← 0;
           for ( int i = 1 to X.conn )
               if ( isCandidate(msg.bFilter, msg.tFilter, X.nbr[i])
                    && distance1 < distance(X.nbr[i], msg.src) )
                   d2 ← distance( nbr[i], msg.from );
                   break;     /* break from the loop */
               endif
           endfor
           if (d2 == 0 ) /* The distance of the node X to the node msg.src is the largest */
               X.isForwarder ← TRUE;
               X broadcasts the query message to all its one-hop neighbors;
           else
               X.isForwarder ← FALSE;
               Drops the query message;
           endif
       else /* X is not a candidate node for message forwarding*/
           Drops the query message;
       endif
   else  /* X has received the message more than once*/
       Drops the query message;
   endif
endif
End
```

Table 4: The function "isCandidates" called by the "bi-filtered forwarding" procedure

```
BOOL isCandidate(int bFilter, int tFilter, T_NODE* n)
Begin
        if (n.conn < bFilter and n.conn >1) || (n.conn > tFilter) then
                n.isCandi  ← TRUE;
                return  TRUE;
        else
                return FALSE;
        endif
End
```

Through the above "bi-filtered forwarding" procedure described in table 3, with just a small amount of redundant message forwarding, the query message submitted by any node can be propagated quickly throughout the whole ad hoc network.

V. EXPERIMENTAL RESULTS

Before testing the performance of the BFF algorithm in comparison with the flooding-based routing algorithm, we construct two scenarios of wireless sensor networks with different node connectivity degrees. The first scenario has 25 nodes and the average connectivity degree of each node is 3.5; the second scenario has 45 nodes and the average connectivity degree of each node is 5.

Then the following experiments are designed and the experimental results are analyzed respectively.

a. Message consumption caused by the flooding algorithm and the BFF algorithm

Considering the message consumption after each hop of message forwarding for the query propagation, we can know that BFF works better than the flooding algorithm. As shown in figure 4, when the message delivery traffic covers 100% of the nodes in the network after 7 hops of message forwarding, the flooding-based routing algorithm consumes 6090 messages which are 1.38 times of that consumed by BFF. That is to say, BFF can save more than 27.5% of messages consumed by the flooding-based routing algorithm while still keeping the 100% node coverage rate.

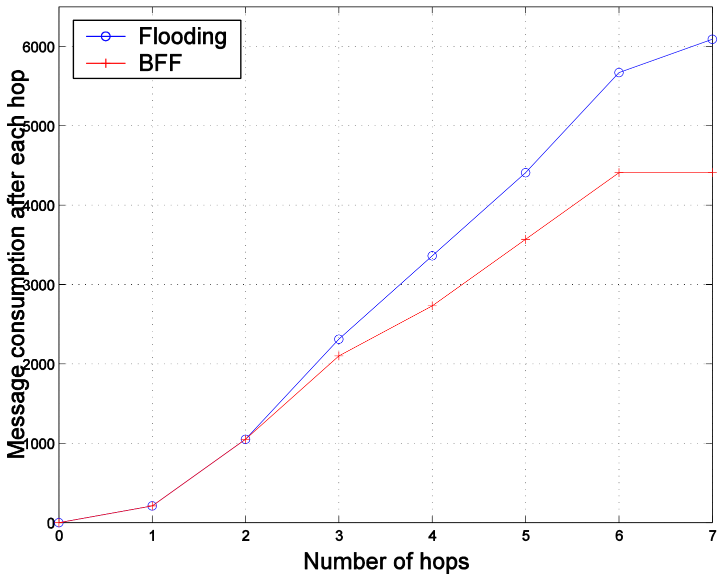

Figure 4. The message consumption accumulated in hops of message forwarding under the flooding algorithm and the BFF algorithm respectively

b. Redundant message consumption caused by the flooding-based routing algorithm and the BFF routing algorithm

As shown in figure 5, the experiment begins with a specified node broadcasting a message to its neighbors, and the query propagation is controlled by the flooding algorithm. The result shows that in both of the network scenarios the flooding-based routing algorithm takes no more than 7 hops of message forwarding for delivering the query message to all the nodes in the network.

To see the redundant message consumption accumulated after each hop of message forwarding in the flooding-based routing algorithm, figure 6 illustrates that in the first hop of message forwarding only a small amount of redundant messages is accumulated, but after that the redundant message consumption increases rapidly with the number of hops increases. That is because in the flooding-based routing algorithm some unnecessary forwarders are involved in the message delivery. Actually, as shown in figure 6, after the seventh hop of message transmission when the message has arrived at every one of the nodes in the network, the redundant message consumption accumulated is above 50% of the overall message consumption, which means more than a half number of messages caused by the flooding-based routing algorithm are unnecessary for the query propagation.

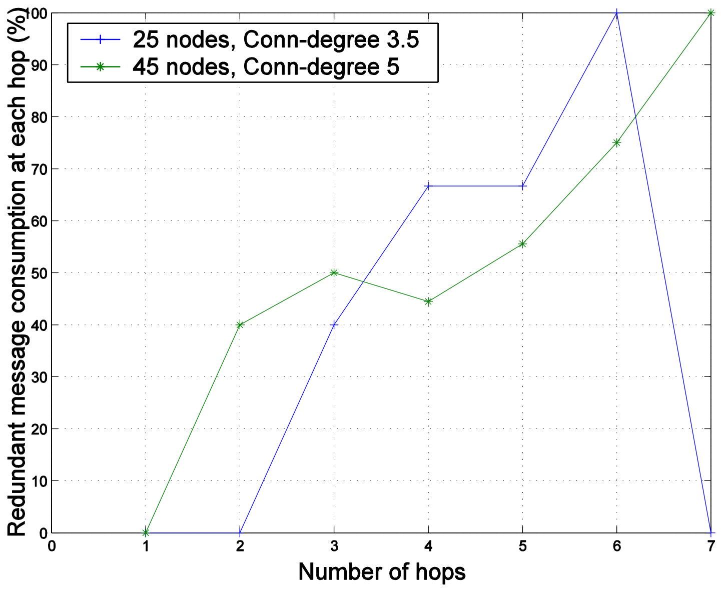

Figure 5. The redundant message consumption at each hop of message forwarding in the flooding-based routing algorithm

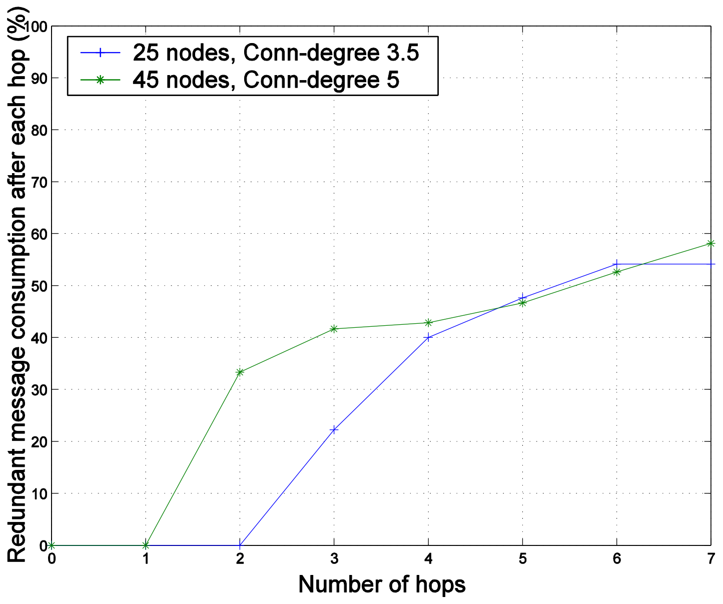

Figure 6. The redundant message consumption accumulated after each hop of message forwarding in the flooding-based routing algorithm

To compare the routing performance between the flooding-based routing algorithm and the BFF algorithm, the redundant message consumption after each hop of message forwarding caused by the flooding-based routing algorithm and the BFF algorithm respectively are traced in figure 7.

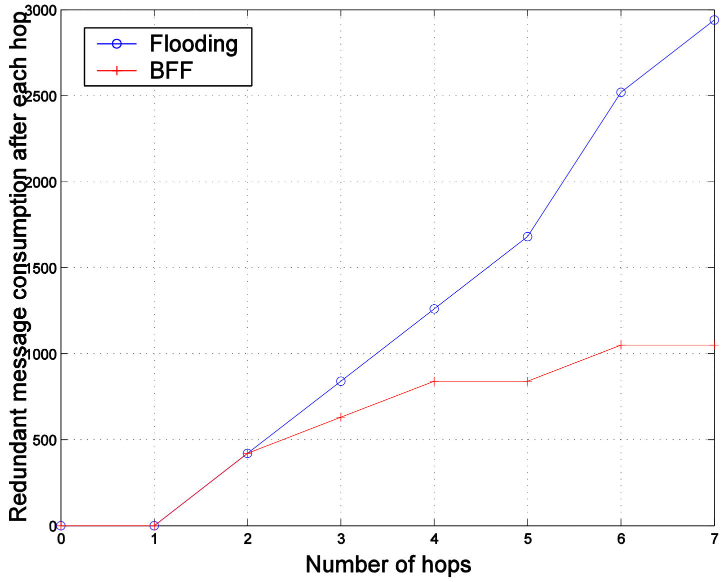

Figure 7. The redundant message consumption accumulated in hops under the flooding algorithm and the BFF algorithm respectively

It is seen from the figure 7 that from the third to the sixth hops of message forwarding the redundant message consumption caused by the flooding-based routing algorithm increases sharply as the number of hops of message forwarding increases.

In comparison, BFF doesn't introduce any redundant messages during the first hop, and the number of redundant messages grows slowly with the increase of the amount of hops, especially after the forth hop of the message forwarding.

In summarization, BFF consumes only a small amount of redundant messages and on the contrary the flooding-based routing algorithm consumes 2.8 times of the redundant messages generated by BFF. In other words, compared with the flooding-based routing algorithm, BFF can save more than 64.29% of the redundant messages consumption for the query propagation throughout the network.

To be in more detail, as shown in figure 4 and figure 7, the message consumption and the redundant message consumption of the BFF algorithm is 27.5% and 64.29% less than that of the flooding-based routing algorithm respectively.

To be noticed, the above experiments also show that the configuration of the top-filter and the bottom-filter can greatly influence the performance of BFF. Actually, when the top-filter and the bottom-filter are reconfigured before each of the above experiments, the result shows that a higher top-filter or a lower bottom-filter may raise the redundant message consumption, while a higher bottom-filter or a lower top-filter may decrease the node coverage rate of the routing when the node is distributed sparsely in the network.

Fortunately, when the top-filter equals to the average node connectivity degree of the network and the bottom-filter equals to 2, BFF shows a 100% node coverage rate and the least amount of redundant message consumption in various scenarios of wireless sensor networks.

Furthermore, with that good performance, the BFF algorithm also shows a good node coverage rate in comparison with the flooding-based routing algorithm, which is proved in the following experiment.

c. Comparison of the node coverage rate under the flooding-based routing algorithm and the BFF routing algorithm

As shown in figure 8, after the forth or the fifth hop of message forwarding, the node coverage rate of BFF is a bit less than that of the flooding-based routing algorithm, but both of them become 100% after the sixth hop of message forwarding, which shows that the delivery speed of BFF is just a bit slower than that of the flooding-based routing algorithm.

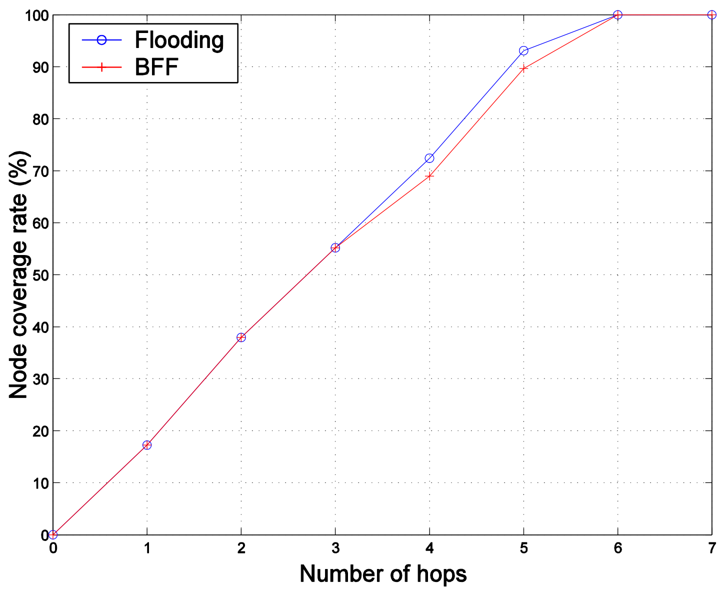

Figure 8. The percentage of the nodes (the node coverage rate) after each hop of message forwarding caused by the flooding algorithm and the BFF algorithm respectively

VI. CONCLUSIONS

In this paper, we compare and analyze the routing characteristics of the flooding-based routing algorithm and that of the optimal routing algorithm for the query propagation in a wireless sensor network respectively. Then we find that eliminating the node whose connectivity degree is too high or too low for the query propagation can greatly reduce the redundant message consumption while still promising a good query propagation speed.

Thus, we introduce the two filters, "top-filter" and "bottom-filter", into the proposed query propagation algorithm (BFF), for selecting appropriate forwarders for propagating the query message throughout the network. Through the two filters implemented on each node, only parts of nodes are designated as the message forwarder for the query message delivery so that the unnecessary message forwarding traffic can be eliminated as much as possible. At the same time, the BFF algorithm promises that the query be propagated throughout the network as quickly as is done by the flooding-based routing algorithm.

The experimental results show that the BFF algorithm guarantees a 100% node coverage rate in a small amount of message delivery traffic while introducing just a little redundant message consumption throughout the wireless sensor network.

REFERENCES

[1] I. Galpin, C. Y. A. Brenninkmeijer, A. J. G. Gray, F. Jabeen, A. A. A. Fernandes and N. W. Paton, "SNEE: A query processor for wireless sensor networks", Distributed and Parallel Databases, vol. 29, No. 1-2, 2011, pp. 31-85.

[2] H. Ehsan and F. A. Khan, "Query Processing Systems for Wireless Sensor Networks", Communications in Computer and Information Science, vol. 150, No.1, 2011, pp. 273-282.

[3] F. Jabeen, S. Nawaz, S. Tanveer and M. Iqbal, "Spatio-Temporal Query Processing Over Sensor Networks: Challenges, State Of The Art And Future Directions", KSII Transactions on Internet and Information Systems, vol. 6, No. 7, 2012, pp. 1756-1776.

[4] O. Diallo, J. J. P. C. Rodrigues and M. Sene, "Real-time Data Management on Wireless Sensor Networks: A survey", Journal of Network and Computer Applications, vol. 35, No. 3, 2012, pp. 1013-1021.

[5] W. F. Liang, B. C. Chen and J. X. Yu, "Top-k Query Evaluation in Sensor Networks under Query Response Time Constraint", Information Sciences, vol. 181, No.4, 2013, pp. 869-882.

[6] S. Pervin, J. Kamruzzaman and G. Karmakar, "Delay-Aware Query Routing Tree for Wireless Sensor Networks", Proceeding of 2012 11th IEEE International Symposium on Network Computing and Applications (NCA), pp. 105-110, United States, Aug. 23-25, 2012.

[7] G. Chatzimilioudis, A. Cuzzocrea, D. Gunopulos and N. Mamoulis, "A Novel Distributed Framework for Optimizing Query Routing Trees in Wireless Sensor Networks via Optimal Operator Placement", Journal of Computer and System Sciences, Journal of Computer and System Sciences, vol. 79, No. 3, 2013, pp. 349-368.

[8] J. J. Kang, K. Y. Lee, J. J. Kim, G. S. Choi, Y. S. Im and E. Y. Kang, "In-network Query for Wireless Sensor Networks", International Journal of Multimedia and Ubiquitous Engineering, vol. 7, No. 2, 2012, pp. 377-382.

[9] A. A. Ahmed and N. Fisal, "A Real-Time Routing Protocol with Load Distribution in Wireless Sensor Networks", Computer Communications, vol. 31, No. 14, 2008, pp. 3190-3203.

[10] J. Gao, L. M. Wei, Y. L. Zhu and L. F. Li, "Routing Optimization Based on Ant Colony Algorithm for Wireless Sensor Networks with Long-Chain Structure", Proc. *IOT Workshop 2012*, pp. 91-97, China, Aug. 17-19, 2012.

[11] S. Kim, "An Ant-based Multipath Routing Algorithm for QoS Aware Mobile Ad-hoc Networks", Wireless Personal Communications, vol. 66, No. 4, 2012, pp. 739-749.

[12] P. S. Sausen, M. A. Spohn and A. Perkusich, "Broadcast Routing in Wireless Sensor Networks with Dynamic Power Management and Multi-coverage Backbones", Information Sciences, vol. 180, No. 5, 2010, pp. 653-663.

[13] Y. S. Yen, H. C. Chang and R. Chang, "Routing with Adaptive Path and Limited Flooding for Mobile Ad hoc Networks", Computers and Electrical Engineering, Vol. 36, 2010, pp. 280-290.

[14] J. H. Zhang, S. Grumbach, D. Q. Yang and A. M. Anaya, "Ripple Routing: An On-demand Routing Protocol for In-network Query Processing on Wireless Sensor Networks", Proc. *ICMA 2010*, pp. 1964-1970, China, Aug. 4-7, 2010.

[15] M. Eslaminejad, S. Abd Razak and M. Sookhak, “Classification of Energy-Efficient Routing Protocols for Wireless Sensor Networks”, Ad hoc & Sensor Wireless Networks, vol. 17, No. 1-2, 2013, pp. 103-129.

[16] R. Khoury, T. Dawborn, B. Gafurov, G. Pink, E. Tse, Q. Tse, K. Almi'Ani, M. Gaber, U. Röhm and B. Scholz, “Corona: Energy-efficient Multi-query Processing in Wireless Sensor Networks”, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 5982, No. 2, 2010, pp. 416-419.

[17] X. Y. Yin, Z. H. Ling and L. P. Guan, “Low Energy Adaptive Routing Hierarchy Based on Differential Evolution,” International Journal on Smart Sensing and Intelligent Systems, vol. 5, No. 2, 2013, pp. 523-547.

[18] A. K. Mohapatra, N. Gautam and R. L. Gibson, “Combined Routing and Node Replacement in Energy-Efficient Underwater Sensor Networks for Seismic Monitoring”, IEEE Journal of Oceanic Engineering, vol. 38, No.1, 2013, pp. 80-90.

[19] G. G. Riva, and J. M. Finochietto, “Pheromone-based In-network Processing for Wireless Sensor Network Monitoring Systems”, Proc. *ICC 2012,* vol. 132, pp. 6560-6564, Canada, June 10-15, 2012.

[20] M. Ye, W. C. Lee, D. L. Lee and X. J. Liu, “Distributed Processing of Probabilistic Top-k Queries in Wireless Sensor Networks”, IEEE Transactions on Knowledge and Data Engineering, vol. 25, No. 1, 2013, pp. 76-91.

[21] Y. S. Chen and Y. W. Lin, “Mobicast Routing Protocol for Underwater Sensor Networks”, IEEE Sensors Journal, vol. 13, No. 2, 2013, pp. 737-749.

[22] J. D. Preethi and R. Sumathi, “An Energy Efficient On-Demand Routing by Avoiding Voids in Wireless Sensor Network”, Proc. *INDIA 2012*, vol. 132, pp. 255-263, India, Jan. 5-7, 2012.

[23] R. H. Cheng, T. K. Wu and C. W. Yu, “A Highly Topology Adaptable Ad hoc Routing Protocol with Complementary Preemptive Link Breaking Avoidance and Path Shortening Mechanisms”, Wireless Networks, vol. 16, No. 5, 2010, pp. 1289-1311.