International Journal of Grid and Distributed Computing
Vol.6, No.5 (2013), pp.83-92
http://dx.doi.org/10.14257/ijgdc.2013.6.5.08

ECTP: An Enhanced Data Collection Protocol based on CTP

Xingming Sun[1, 2], Hongwei Qian[1], Baowei Wang[1, 2], Qi Liu[1, 2]

School of Computer and Software, Nanjing University of Information Science & Technology, Nanjing 210044, China
Jiangsu Engineering Center of Network Monitoring, Nanjing 210044, China

sunnudt@163.com, oneqhw@163.com, wbw.first@163.com, qrankl@163.com

Abstract

In wireless sensor networks (WSNs), the communication radius of a single sensor node is constrained. Thus, many-to-one and multi-hop routing protocols are designed to relay collected data back to the sink node. One of the challenges handled by present routing protocols is to prevent or reduce traffic congestion, which inevitably causes high packet drop rate, low energy efficiency, and long end-to-end delay. This paper presents an enhanced data collection protocol based on the Collection Tree Protocol (CTP), which introduces the concept of congestion detection and congestion avoidance into CTP. We have implemented test-bed experiments containing 10 TelosB motes, and compared results of our ECTP with the original CTP protocol and also with another enhanced CTP which takes the main mechanisms of ECODA. According to the experimental results, our ECTP improves the efficiency and performance of data collection tasks in WSN compared to the other two.

Keywords: *WSN; ECTP; CTP; Congestion detection; Congestion avoidance*

1. Introduction

The WSN proves to be a powerful tool for prolonged surveillance of large scale phenomena [1, 2, 3]. A WSN network is typically composed of large quantities of tiny wireless sensor nodes, each of which constrains limited computational capability and energy [4]. Due to short transmission range of a sensor node, the WSN usually adopts a multi-hop and many-to-one routing pattern to transmit collected data from surrounding environment to the sink node [5]. To suit the needs of WSN, a number of many-to-one routing protocols proposed; of these protocols, CTP is a commonly used and thoroughly tested one [6]. CTP network maintains an important value, ETX [6] (Expected Transmissions Value) to estimate the link quality of paths forwarding data from nodes to the sink node. However, ETX doesn't address the problem of congestion, where multiple nodes try to transmit data to a node who however cannot forward the received data to its next routing hop in time, due to queue overflow or other situations [5, 7, 8]. Congestion in WSN can be a severe problem, as it causes a plenty of malfunctions such as packet loss, low throughput, energy inefficiency, and consequently short lifetime. Having built a prototype depicting the congestion issue, we have conducted experiments using the CTP protocol in 10 motes. As a typical example, Figure 1 shows the collection tree built by CTP in one of these experiments. Congestion happened at Node 2 during network time. In CTP, the way to solve the problem is to set the C bit, and all the children of Node 2 will look for an alternate next hop to transmit data, thus this will cause a large jitter in the network. In CTP-ECODA, the way to solve the problem is to decrease the data sending rate, thus this will reduce the overall transmission of the network. Other experiments yield similar results, as implies that CTP and CTP-ECODA by themselves cannot provide favorable congestion detection and avoidance services. Similar problems happened in the sensor networks deployed in the real wild environment.

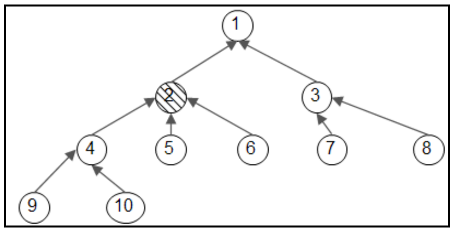


Figure 1. A topology built by CTP, Node 2 is in congestion

ECTP, a data collection protocol based on CTP, aims to solve the problem of congestion. It modifies CTP from two aspects.

First, ECTP adds a congestion detection scheme to measure a node's congestion level and to determine whether the node is in congestion. It will count the transmissions of upstream and downstream of a node in the network during a given period time. If the ratio of them is larger than the threshold for n times continuously, we define it a congestion node.

Second, ECTP adopts a congestion avoidance strategy to migrate the load of the congestion node. It requires the congestion node to send a control message to inform one of its neighbors to choose an alternate route. Then the neighbors of the node will judge whether themselves is the node to accept the message. If so, the specified one will choose an alternate route. As we can see from Figure 2, Node 2 is in congestion, and it wants to reject Node 6, due to its worst ETX of the neighbors, so it broadcasts a control message containing the Id of 6, and then Node 6 migrates to Node 3.

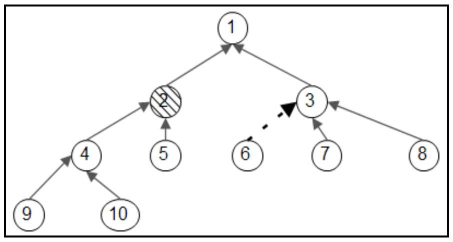


Figure 2. Node 6 migrates from Node 2 to Node 3

Besides, the selection of alternate route in ECTP should insure that no routing loops will occur and the alternate route should not traverse a congestion node itself. ECTP relies solely on the CTP beacon mechanism to deliver necessary information and thus does not introduce any network overhead.

The contributions of this paper are as follows. First, we introduce congestion level as the metric to measure a node's congestion state. Second we design a localized scheme to migrate the load of the congestion nodes to solve congestion. Third, we implement ECTP and evaluate the performance of it in a 10-node test-bed.

The rest of this paper is organized as follows. Section 2 describes the related work about congestion control in WSN. The design of ECTP is discussed in Section 3. Section 4 demonstrates the implementation of ECTP and Section 5 evaluates ECTP through experiments. Finally, Conclusions and perspective to future work are given in Section 6.

2. Related work

In this section, present methods of congestion control in the wireless sensor network have been reviewed and discussed. These methods mainly focus on two techniques that span different layers of the protocol stack, *i.e.*, rate limiting schemes and hop-by-hop flow control metrics.

2.1. Rate limiting schemes

C.Y. Wan *et al.*, have proposed CODA [9], where congestion is detected by sampling the wireless medium and by monitoring the queue occupancy. As soon as a node detects congestion, it broadcasts a back pressure message upstream. The upstream nodes will then throttle the traffic volume to alleviate congestion. In addition, CODA also employs closed-loop source regulation, where long-term end-to-end constant feedback from the sink to source nodes is required to adjust the sending rate through employing the additive increase and multiplicative decrease (AIMD) scheme.

Fusion [10] examines three techniques that span different layers of the traditional protocol stack, *i.e.*, hop-by-hop flow control, rate limiting source traffic when transit traffic is present, and a prioritized medium access control protocol. Basically, it mitigates congestion by throttling the transmissions of the upstream nodes and the source nodes. However, in its rate limiting mechanism, nodes need to continuously watch their parents' sending actions to determine when to generate tokens. This continuous monitoring is costly and consumes more energy.

In [11], Sarkar and Tassiulas propose a fair distributed congestion control system for multi-rate, multi-cast networks. Their system works max-min fair on these networks and induces small overhead for message exchange only.

The CAR [12] is a network layer solution to provide differentiated services in congested sensor networks. It also prevents severe degradation of services to low priority data by utilizing uncongested parts of the network. MCAR [12] is primarily a MAC-layer mechanism used in conjunction with routing to provide mobile and lightweight conzones to address sensor networks with mobile high priority data sources and/or bursty high priority traffic. Compared to CAR, MCAR generates a less overhead but degrades the performance of LP data aggressively.

ECODA [3] comprises three mechanisms, including using dual buffer thresholds and weighted buffer difference for congestion detection, a flexible queue scheduler for packet scheduling, and a bottleneck-node-based source scheme for sending rate control.

2.2. Hop-by-hop flow control metrics

Zawodniok and Jagan Nathan [13] present a decentralized predictive congestion control (DPCC) scheme, which detects the onset of congestion using queue utilization and the embedded channel estimator to predict the channel quality. In DPCC, the adaptive flow control algorithm selects suitable rate enforced by the adaptive back-off interval selection scheme.

In [14], Chen and Yang propose a congestion avoidance scheme based on lightweight buffer management, which follows the basic idea of the credit-based hop-by-hop flow control in ATM networks, and employs an 1-k buffer scheme to prevent hidden terminals from causing congestion. Although the traffic control can effectively alleviate congestion, it could impose a negative impact on fidelity.

The authors in [8] have proposed a distributed congestion feedback algorithm named as DCCA to solve sink bottleneck problems in multi-hop WSNs. It uses a queue based congestion level detection scheme in the MAC layer and a hop-by-hop feedback notification scheme in the transport layer. In this way, forwarding or dropping packet will be decided by each sensor node in the network layer.

In summary, current congestion schemes detect and avoid congestion by adjusting data sending rate and backpressure. In the case of a WSN network where real-time data are generated, however, traditional approaches cannot fulfill the requirements. Our ECTP is therefore proposed to deal with the problem.

3. Design of ECTP

3.1. Design goals and challenges

The design of ECTP aims to detect and avoid congestion, without undermining the reliability and stability of the original CTP or introducing more overheads. The following crucial challenges are met and handled in the ECTP.

First, how to recognize the congestion node in the early stage is the problem. In ECTP, an agile metric is introduced to reflect the congestion level of a node, so as to leave enough time for the protocol to deal with congestion.

Second, after congestion is detected, finding the way to reduce the congestion node's transmission load becomes critical to alleviate the phenomenon. The ECTP therefore provides transmission migration scheme to avoid potential load. Additionally, a congestion avoidance scheme is designed to control the number of congestion nodes.

According to a number of experiments conducted in our lab, the CTP has been proved to be reliable and stable, and reacts quickly to routing inconsistencies with very few beacons. Hence, the ECTP is designed as a supplement of CTP, which deals with the special case that a congestion node emerges. Furthermore, as the sending rate of beacons is highly relative to the frequency of network topology changes, the congestion avoidance scheme in the ECTP can be optimized by keeping the network's topology as much as possible.

Based on the discussion above, the ECTP is designed with consideration of three aspects, *i.e.*, congestion detection, congestion avoidance and integration with original CTP.

3.2. Congestion detection

In ECTP, congestion in a WSN network is recognized by introducing the definition of congestion level (CL), as

$$CL = Pkt_{in} \times N / pkt_{out},$$

Pkt_{in} denotes the transmission inflows of a node, and Pkt_{out} denotes the transmission outflows. N is a constant with the value set to 100 by default. The congestion node is detected by comparing the value of CL with the $THRESHOLD$, of which the default value is 95. A node will consider itself congested if CL is larger than $THRESHOLD$ for continuously n times, where n being set to 3 by default.

When a node is turned on, it counts all packets it gains and issues. Every packet received by the node will make the Pkt_{in} increase by one. Whereas the Pkt_{out} is set by monitoring on the number of packets sent to its next hop. The CL can therefore be calculated periodically to recognize potential congestion happened in a node.

In our congestion detection scheme, more consideration has been given in order to make the detection efficient and accurate. One is to maintain the trade-off between the accuracy of congestion detection and the computation burden spent on calculating the real-time CL value. By default, the CL value is checked every 5 minutes in our ECTP.

Another concern is on the effectiveness of the CL equation since in some cases, congestion cannot be detected relying on the CL value only. For example, when the sending queue of a node is full, the node is actually in congestion, though the Pkt_{in} could be very low. In this case, a timer is used to dynamically coordinate Pkt_{in}, Pkt_{out} and CL.

3.3. Congestion avoidance

The congestion avoidance scheme tries to disconnect the communication of one neighbor node to the congestion node. Other neighbors will be disconnected until the node recovers from congestion. The congestion avoidance is achieved in two phases. First, the congestion node chooses one of its neighbors and stops receiving any packets from it. In order to keep the stability of the network, here the one with worst ETX is selected. Second, the rejected node has to select an alternate node as its next hop. The alternative route should meet the following requirements:

(1) The alternate next hop should be loop-free;

(2) The alternate next hop should not traverse another congestion node;

(3) The alternate route should avoid the extremely lossy links.

Routing Probability Allocation: If the disconnected neighbor of a congestion node finds a suitable next hop, it will forward packets to the alternate destination. There are some considerations in designing the congestion avoidance scheme.

First, the migrated load may cause the nodes on the alternative route to become congested; nonetheless, the problem can be handled as the ECTP can detect it via the CL value.

Second, a congestion node may declare it is free from congestion as soon as its CL is below the threshold, however according to our experiments, it is likely to be congested again as its children get back to their original route, which makes the entire network jitter and unstable. In order to avoid such a situation, a congestion node can only declare itself not in congestion after its CL value is below the threshold for continuous n measurements. The default value of n is 3.

3.4. Integration with CTP

The ECTP is integrated into the CTP protocol for congestion detection and avoidance only. This ensures the ECTP acts as same as the original CTP when there is no congestion node in the network. When a congestion node is detected, the node will broadcast a CTP control message, with a one-byte field appended. It is used to inform the congestion to its neighbors and tell a neighbor to choose an alternate next hop.

4. Implementation

The original CTP mainly consists of three parts: routing engine, forwarding engine and link estimator. CTP uses the Four Bit Link Estimator (4bitle) [5] as its link estimator.

In our design, we still use 4bitle as link estimator, for it is good enough for us to do with ECTP. In addition, to suit the needs of ECTP, a timer is designed to the forwarding engine to help determine whether a node is in congestion. In Figure 3, it can be seen that the main function of the timer is to deal with the relations of Pkt_{in}, Pkt_{out}, CL.

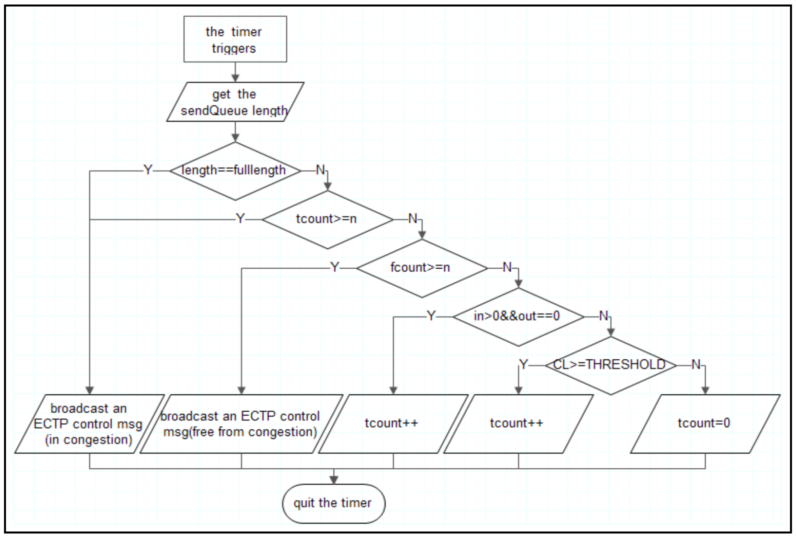


Figure 3. The flow chart describes the measure to determine whether a node is in congestion. This is only a part of the whole timer trigger function. The *tcount* denotes the counts of the node that is detected in congestion and *fcount* denotes the counts of the node that is detected not in congestion, *in* is for Pkt_{in}, and *out* is for Pkt_{out}

The ECTP also modifies the routing engine of the CTP, by utilizing a bit of the CTP control message to mark that the node is in congestion and a one-byte field to indicate the neighbor node to be rejected. Figure 4 shows the actions when a neighbor node receives a control message from its parent node.

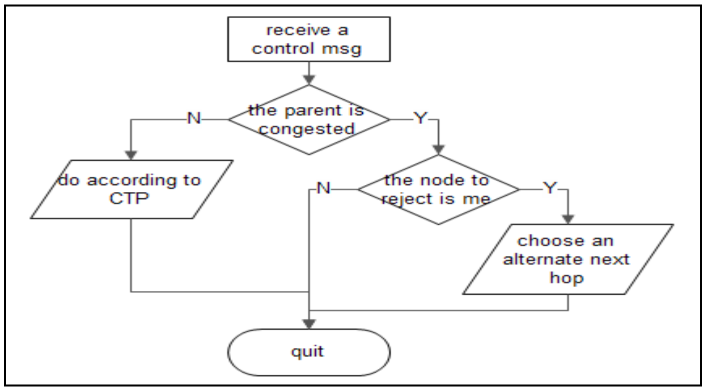


Figure 4. The actions when a neighbor node receives a control message from its parent node. It will check whether its parent is a congestion node

5. Results and Evaluation

We have tested ECTP using 10 TelosB motes, each of them includes a mini-USB port for programming and data transfer, an IEEE802.15.4 radio TI CC2420, a low power MCU MSP430 F1611with 10K RAM, an external flash chip up to 1MB. Figure 5 shows the TelosB motes used in our experiments. The 10 motes are densely deployed within an area of 60 square meters.

Figure 5. TelosB motes used in our experiments

We have run CTP, CTP-ECODA and ECTP consecutively for 3 hours and record all the packets received at the sink node. For ECTP, we set the transmission measure interval x to 5 minutes, the *CL* threshold *THRESHOLD* to 95. In our experiment, Node 1 is selected as the sink node.

Figure 6 depicts the overall transmissions of these nodes using the CTP, CTP-ECODA and ECTP respectively. According to the figure, the CTP and CTP-ECODA have generated similar traffic, which is 19967 and 20003 respectively, but the ECTP raises the amount to 20240, which are more than 200 packets than the other two. This is mainly because the CTP drops packets when it encounters congestions, and CTP-ECODA decreases the sending rate if the queue occupancy is too high; but in the ECTP, only one neighbor is affected without much influence to other ones of the congested node.

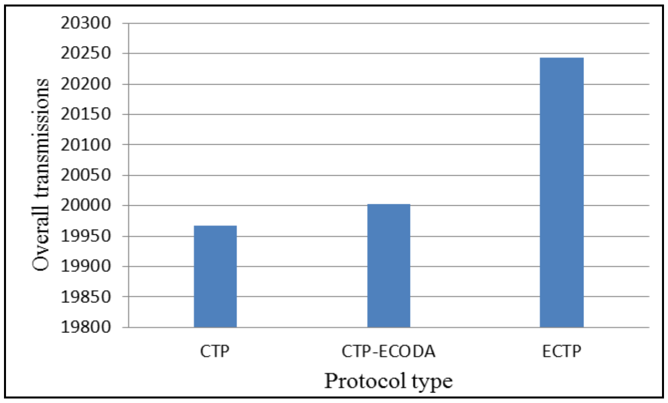


Figure 6. A comparison of the overall transmissions of three protocol types

In Figure 7, the times of a node being chosen as a parent are depicted. Considering the limited communication ability, the more a node is selected as the next hop of others, the more likely it will become congested. The CTP has shown unstable allocation of the parent role, where the times that Node 8 was chosen as the parent are much higher than other nodes. In CTP-ECODA and ECTP, much more balanced allocation has been performed.

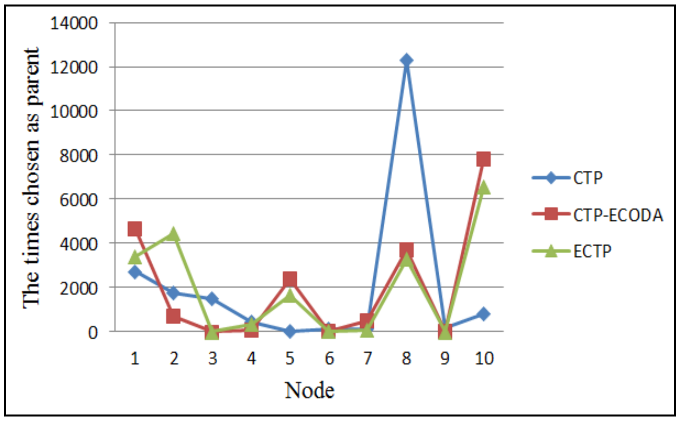


Figure 7. The frequency that each node is chosen as the parent in one hop

Figure 8 illustrates a topology during the network time using the ECTP protocol, where six nodes chose Node 5 as their next hop. In this case, Node 5 would have got congested soon if no appropriate congestion detection and avoidance schemes had been employed. In terms of the CTP protocol, all children of node 5 will choose an alternate node in this case; whereas the ECTP handle the congestion logically. As the result, for about 20 minutes, the topology of the network using the ECTP protocol had changed as depicted in Figure 9, where the child nodes had found their own alternative routes to avoid congestion.

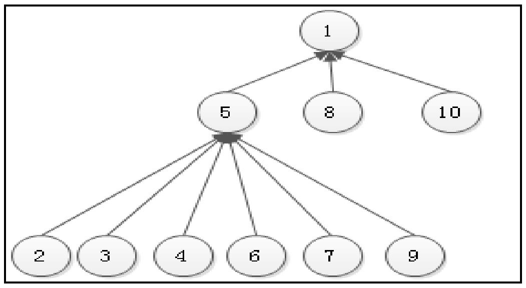


Figure 8. One topology during the network time in ECTP

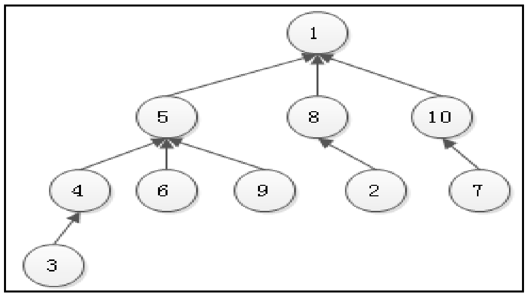


Figure 9. The topology has changed after 20 minutes in ECTP

6. Conclusions and Future Work

This paper describes the design and implementation of ECTP, an enhanced version of the CTP protocol, which considers both congestion detection and avoidance. Experiments in our lab have shown that the ECTP outperforms the original CTP protocol and the CTP-ECODA. This is achieved via optimized congestion detection and avoidance schemes used in the ECTP. At present, the threshold for detecting congestion nodes is set manually in the ECTP. In the future, this threshold is considered to be set dynamically and automatically according to network traffic. Meanwhile, the ETX is used to determine the neighbor to be disconnected; more factors need to be added to make the decision more reasonable.

Acknowledgements

This work is supported by the NSFC (61232016, 61173141, 61173142, 61173136, 61103215, 61070196, 61070195, and 61073191), National Basic Research Program 973 (2011CB311808), 2011GK2009, GYHY201206033, 201301030, 2013DFG12860 and PAPD fund.

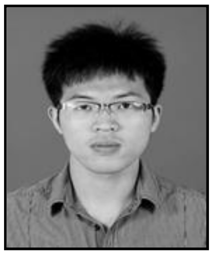
References

[1] A. Mainwaring, J. Polastre, D. Culler and J. Anderson, "Wireless sensor networks for habitat monitoring", ACM, **(2002)**, pp. 88-97.
[2] B. Krishnamachari and S. Iyengar, "Distributed bayesian algorithms for fault-tolerant event region detection in wireless sensor networks", IEEE Trans. on Computers, **(2004)** March, pp. 241-250.
[3] M. Ding, D. Chen, K. Xing and X. Cheng, "Localized fault-tolerant event boundary detection in sensor networks", Proceedings IEEE, vol. 2, **(2005),** pp. 902-913.
[4] I. F. Akyildiz, W. Su and Cayirci, "Wireless sensor networks: a survey", Computer networks, **(2002)**, pp. 393-422**.**
[5] Z. Li, W. Zou and T. Qi, "A cross-layer congestion control strategy in wireless sensor network", IEEE Conference, **(2011)**, pp. 173-177**.**
[6] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss and P. Levis, "Collection tree protocol", ACM Conference, **(2009),** pp. 1-14**.**
[7] L. Tao and F. Yu, "ECODA: enhanced congestion detection and avoidance for multiple class of traffic in sensor networks", IEEE transactions, vol. 56, **(2010)**, pp. 1387–1394**.**
[8] M. Zhang, W. Cai and L. Zhou, "Hop-to-hop congestion feedback mechanism for sink bottleneck problem in WSNs", IEEE Conference, **(2012)**, pp. 142-145**.**
[9] C. Y. Wan, S. B. Eisenman and A. T. Campbell, "CODA: congestion detection and avoidance in sensor networks", ACM Conference, **(2003)**, pp. 266-279**.**
[10] B. Hull, K. Jamieson and H. Balakrishnan, "Mitigating congestion in wireless sensor networks", ACM Conference, **(2004),** pp. 134-147**.**
[11] S. Sarkar and L. Tassiulas, "Fair distributed congestion control in multi rate multi cast networks", IEEE/ACM Transactions, **(2005),** pp. 121-133**.**
[12] R. Kumar, R. Crepaldi, H. Rowaihy and A. Harris, "Mitigating performance degradation in congested sensor networks", IEEE Transactions, **(2008)**, pp. 682-697.
[13] M. Zawodniok and S. Jagannathan, "Predictive congestion control Protocol for wireless sensor networks", IEEE Trans. Wireless Comm., vol. 6, no. 11, **(2007)** November, pp. 3955-3963**.**
[14] S. Chen and N. Yang, "Congestion avoidance based on lightweight buffer management in sensor networks", IEEE Transactions, **(2007),** pp. 3955-3963**.**
[15] R. Fonseca, O. Gnawali, K. Jamieson and P. Levis, "Four-bit wireless link estimation", In HotNets VI, ACM, **(2007).**

Authors

Xingming Sun

He is a professor in the School of Computer and Software, Nanjing University of Information Science and Technology, China from 2011. He received the B.S.degree in Mathematical Science from Hunan Normal University and M.S. degree in Mathematical Science from Dalian University of Technology in 1984 and 1988, respectively. Then, he received the Ph.D. degree in Computer Engineering from Fudan University in 2001. His research interests include information security, network security, cryptography and ubiquitous computing security.



Hongwei Qian

He received his B.S. degree in Computer Science from Nanjing University of Information Science and Technology, China in 2011. Currently he is studying for his M.S degree in Meteorological Information Security at the same university. His research interests include wireless networks and network security.

Baowei Wang

He received his B.S. and Ph.D. degrees in Computer Science from Hunan University in 2005 and 2011, respectively. He is currently working as a lecturer in School of Computer and Software, Nanjing University of Information Science and Technology. His research interests include steganography, wireless networks and securing ad hoc networks.

Qi Liu

He received his BSc degree in Computer Science and Technology from Zhuzhou Institute of Technology, China in 2003, and his MSc and PhD in Data Telecommunications and Networks from the University of Salford, UK in 2006 and 2010. His research interests include context awareness, data communication in MANET and WSN, and smart grid. His recent research work focuses on intelligent agriculture and meteorological observation systems based on WSN.