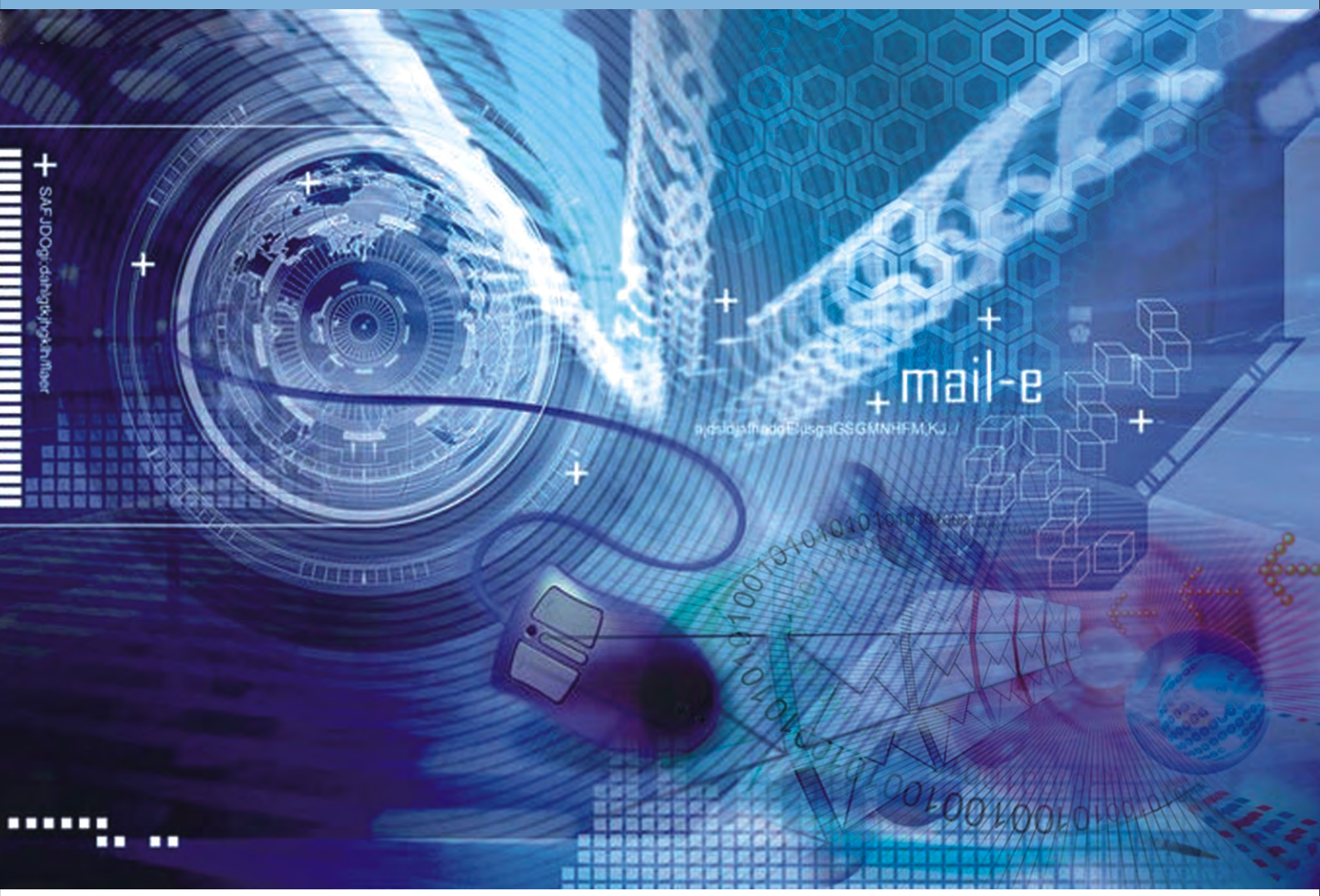
ISSN: 1945-3078 Volume 2, Number 9, September 2010

Scientific Research

Wireless Sensor Network

Chief Editor : Kosai Raoof



ISSN: 1945-3078

www.scirp.org/journal/wsn/

Journal Editorial Board

ISSN 1945-3078 (Print) ISSN 1945-3086 (Online)

http://www.scirp.org/journal/wsn/

Announcement: Special Issue of Wireless Sensor Network

It is our great pleasure to announce that all papers published in this issue are recommended by the Organization Committee of CWSN'10 which is organized by the Sensor Network Technical Committee of China Computer Association.

We hope this special issue can attract more scholars to submit their research papers to WSN, the journal that publishes the good quality research and review articles in all important aspects of wireless sensor network and applications.

WSN Editorial Office

Wireless Sensor Network, **2010, 2, 655-723**
Published Online September 2010 in SciRes (http://www.SciRP.org/journal/wsn/)

TABLE OF CONTENTS

Volume 2 Number 9 September 2010

Wireless Sensor Network (WSN)

Journal Information

SUBSCRIPTIONS

The *Wireless Sensor Network* (Online at Scientific Research Publishing, www.SciRP.org) is published monthly by Scientific Research Publishing, Inc., USA.

Subscription rates:

Print: $50 per issue.
To subscribe, please contact Journals Subscriptions Department, E-mail: sub@scirp.org

SERVICES

Advertisements
Advertisement Sales Department, E-mail: service@scirp.org

Reprints (minimum quantity 100 copies)
Reprints Co-ordinator, Scientific Research Publishing, Inc., USA.
E-mail: sub@scirp.org

COPYRIGHT

PRODUCTION INFORMATION

For manuscripts that have been accepted for publication, please contact:
E-mail: wsn@scirp.org

Wireless Sensor Network, **2010, 2, 655-660**
doi:10.4236/wsn.2010.29078 Published Online September 2010 (http://www.SciRP.org/journal/wsn)

Scientific Research

L^{3}SN: A Level-Based, Large-Scale, Longevous Sensor Network System for Agriculture Information Monitoring*

Yongcai Wang, Yuexuan Wang, Xiao Qi, Liwen Xu, Jinbiao Chen, Guanyu Wang
Institute for Theoretical Computer Science, Tsinghua University, Beijing, China
E-mail: *wangyc@tsinghua.edu.cn, wangyuexuan@tsinghua.edu.cn, qix*08*@mails.tsinghua.edu.cn, kyoi.cn@gmail.com, cjb*06*@mails.tsinghua.edu.cn, wgiveny@gmail.com*
Received June 1, 2010; *revised July* 5, 2010; *accepted August* 17, 2010

Abstract

We developed L^{3}SN, a scalable, longevous, adaptive, and internet accessible wireless sensor network system for agriculture information monitoring, which is meticulously designed to meet the requirement of thousands hectares coverage, years of time monitoring and the adverse environment. The system architecture, the agriculture sensor device, the mesh protocol, and the web-based information processing platform are introduced. We also presented some implementation experience. The mesh protocol (LayerMesh) is highlighted, in which "stair scheduling" and "distributed dynamic load-balancing" are proposed to response the scalability, longevity and adaptivity requirements. We believe the design of L^{3}SN is useful to many other large-scale, longevous applications such as hydrologic monitoring, geological monitoring etc.

Keywords: Wireless Sensor Network, Agriculture Information Monitoring, Large-scale, Longevity, Adaptivity

1. Introduction

Agriculture Informatization is an important area, related to people life and national interest, is proposed to be empowered by wireless sensor network technology. For agriculture information monitoring, the water, soil conditions, the crops, fruits conditions, as well as the conditions of livestock are required to be monitored in real-time by many spatially distributed wireless sensors. These wireless sensors are battery or solar energy powered, equipped with wireless radio, storage unit, data processing unit and various sensing units. They are desired to be easily deployed into the large-scale farmland, to self-organize to a functional distributed multi-hop network via wireless communication, to work for years of time to collect data, and to be self-maintenance against the environmental dynamics of the four seasons.

If such systems are available, they can bring many economic and social benefits. However, although the wireless sensor network technology has been fast developed in the past ten years [1], it is still very challenging to realize such an agriculture information monitoring system. The difficulties come from three aspects.

1) Extremely large in scale. The farmland is often in the scale of millions hectares. The complexity of network organization and routing will turn to a qualitative change when the network scale becomes very large.

2) Extremely long lifetime. Agriculture applications need the network be functional for years of time, but the scarce energy on the sensor node can hardly afford this, especially when the network is large and the sensor has many data to forward.

3) Adverse environment. The adverse weather will challenge the durability of the hardware. The growth of crops will block or affect the wireless links, causing the network working in a highly dynamic radio environment.

Existing studies have presented some implementation trails, but seldom results have been presented to thoroughly investigate and solve above challenges. In this paper, we propose and demonstrate L^{3}SN: A Level-Based, Large-Scale, Longevous Sensor Network for Agriculture Information Monitoring. We focus on the design and implementation issues to show how the above challenges are handled. Our main contributions are two

*PAPER CLASSIFICATION: Application System of Wireless Sensor Network.

This research is supported by the national 863 high tech R&D program of the Ministry of Science and Technology of China under grant No. 2006AA10Z216, the National Science Foundation of China under grant No. 60604033, and the National Basic Research Program of China Grant 2007CB807900, 2007CB807901.

folds.

1) The design and development of the L^3SN system, including system architecture, interfaces, hardware, mesh protocol, and the web-based information processing platform.

2) The level-based mesh protocol, which is named as LayerMesh is highlighted. We specially designed "stair scheduling", and "distributed dynamic load-balancing" to handle the above challenges.

To our best knowledge, it is the first time that the following key points are developed and demonstrated for the large scale sensor network.

1) Load-balancing problem in large scale network is solved and implemented by a fully distributed algorithm.

2) The two-tiered architecture and the open interface design provides infinite scalability.

3) The "stair scheduling" and "adaptive routing" provides novel solutions for energy efficient and adaptive longtime maintenance.

With these key issues, we designed and developed the L^3SN system. The following sections of this paper are organized as follows. In Section 2, related work will be introduced. In Section3, the architecture and interfaces of L^3SN system will be presented. The LayerMesh protocol will be introduced in Section 4. Some implementation results and experience are discussed in Section 5. We conclude the paper in Section 6.

2. Related Works

2.1. Agricultural Information Monitoring Using Wireless Sensor Network

Wireless sensor network system for agriculture information monitoring has attracted great research attentions. Some preliminary experimental systems have been reported. Jenna Burrell, *et al*., investigated different sensor network configurations in vineyard application [2], and summarized some design guidelines for agricultural monitoring system. Murat Demirbas, *et al*. [3] deployed a small scale sensor monitoring system in a green house and drew suggestions that single-hop cluster should be a better solution for easy-to-use and network longevity. A mobile field data acquisition system was developed by Gomide *et al*. [4] to collect data for crop management and spatial-variability studies. Mahan and Wanjura [5] cooperated with a private company to develop a wireless, infrared thermometer system for infield data collection. Cross-bow developed eKo system [6], which involves solar-powered "eKo node" with zigbee radio and "eKo view" for real-time data rendering. It uses Xmesh protocol to make the eKo system easily setup.

2.2. Large-Scale Longevious Sensor Network

Studies of Large-scale longevious sensor network can be categorized into two classes: 1) application-oriented and 2) theoretical-oriented. In the first class, an early study of WSNs for habitat monitoring is reported in [7]. In [8], Werner-Allen *et al.* deployed a WSN at Ecuador to monitoring the activity of an active volcano. But the system scale is only 16 nodes. GreenOrbs [9] is reported as a large-scale system for canopy closure estimates, which had 120 sensors in 2009. However they used only the built in low power listening mode to conserve energy, which is not very energy efficient.

In theoretical aspect, many results can be found in the literature. Li *et al*. [1] studied the message, energy and time complexities for data collection, query and aggregation in large scale network. Yick *et al*. [10] provided an extensive survey for the main results of this area. However, up to now, few results are found to thoroughly investigate and solve the scalability, longevity and environment dynamics of agricultural wireless sensor network.

3. L^3SN System: Architecture and Interfaces

We designed and developed L^3SN. The system architecture and interfaces are introduced in this section.

3.1 System Architecture

L^3SN is designed with the aim of open architecture and standardized interfaces. **Figure 1** shows the system architecture and the standardized interfaces.

Particularly, the architecture of L^3SN includes a Two-tiered Sensor Network for data collection and an Information Service Platform for data rendering and processing.

1) Two-tiered Sensor Network.

The sensor network in L^3SN is composed by many low-tier energy-limited nodes (LNs), and some high-tier gateways. The LNs in low-tier capture environmental data and report the raw data to the gateway. The gateways organize the neighboring LNs into clusters and work as the cluster head. It collects information in its cluster, aggregates the information, and reports the results to the Internet. Through the Internet, the data captured by the wireless sensors are finally reported to the service platform and utilized by end users.

2) Information Service Platform.

The information service platform contains data logging daemon, database, and web server. It provides data storage, data processing, data querying and other agricultural information services to users.

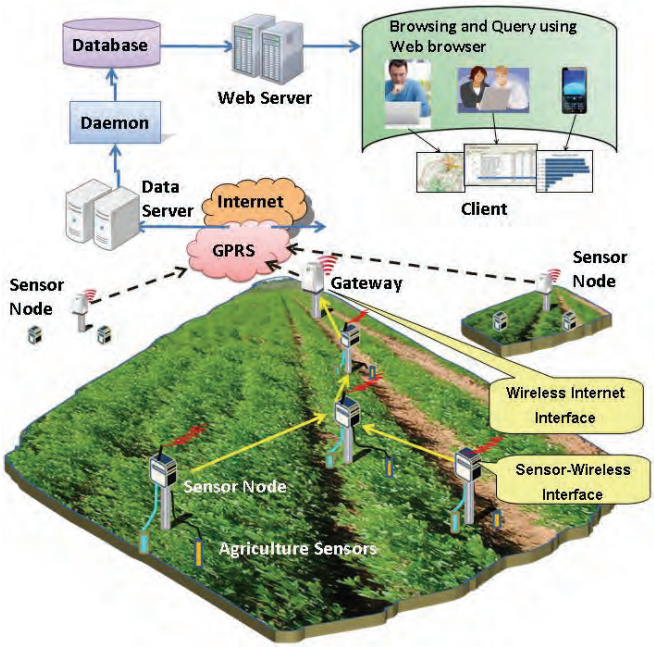


Figure 1. System architecture and the standardized interfaces in L^{3}SN system.

3.2 Standardized Interfaces

In order to easily integrate more kinds of agriculture sensors into the system, and in order to standardize the connection from the sensor network to the internet service platform, we propose two standard interfaces in L^{3}SN:

1) Sensor-Wireless Interface

The "Sensor-Wireless Interface" defines mapping functions, which transform the sensory data that is captured by the sensor devices to the meaningful data which is easily interpreted by the LN nodes. For a sensor type S_i, which provide raw data X, the interface is defined as:

$$S_i(X)=\{Y = f(X); (Y_{min}; Y_{max}); Y_{precision}\} \quad (1)$$

where f(X) is a mapping function which transforms the raw data X to meaningful value Y. The variables (Y_{min}; Y_{max}) characterize the measuring range of the sensor S_i, and $Y_{precision}$ is the measurement accuracy. With this interface, the raw sensory data of diverse format can be transformed to the regularized format. Therefore, data from different kinds of sensors can be easily interpreted by the LN nodes.

2) WSN-Internet Interface

The "WSN-Internet Interface" performs the task of *data aggregation*. Since the gateway collects data from all the LNs in its cluster, the data at the gateway is large in amount and has some redundancy. In addition, the message collected from a LN contains both the sensory information and the routing information. The sensory information is the agriculture data, and the routing information represents the multi-hop route that the message has traveled. The data aggregation at the gateway is performed in the following way:

1) *Non-compressive aggregation to the sensory data.* The agriculture data from different sensor is accumulated and forwarded to Internet in bulk for avoiding of information loss.

2) *Compressive aggregation to the routing data.* The routing information is mainly used for topology construction and maintenance at the service platform. If all the routing information from all the messages is transmitted to the service platform, it will be very redundant. A light weight compressive aggregation algorithm is developed. If the received routing information at the gateway is: $R = [R_1, R_2, \ldots, R_n]$, where n is the number of sensors in the cluster. R_i is the route from the node i to the gateway. We find that *the routes of leaves are enough to construct the routing tree*, so the routes from the intermediate node will be filtered out during the aggregation. A white list based searching algorithm is developed to carry out data aggregation. The compression rate is n_l/n, where n_l is the number of leaf nodes.

Once finishing the aggregation, the sensory information and the routing information are formatted by the Wireless-Internet interface into standardized message types by adding preamble and ending marks, so that the information platform can easily interpret the received information.

Using the above two standardized interfaces, the L^{3}SN system can integrate different kinds of hardware, including sensors and gateways. So user can focus on their application-oriented development and hardware, and easily access the wireless sensor network protocols and the information platform of L^{3}SN, which support the scalability and easy-to-use features.

3.3. Mesh Protocol

For the two-tiered sensor network, sensors in the different clusters use different radio channel and are assigned different GroupIDs, so that sensors in one cluster will not affect the sensors in the other clusters. A LayerMesh protocol is meticulously designed within one cluster to organize the large amount low-tier sensors to work in energy efficient way, to smartly select multi-hop route and to be adaptive to the environmental dynamics. The key feature of LayerMesh is that routing and scheduling are based on the level information.

The basic functions of the LayerMesh can be separated into two phases:

1) Setup phase. When the sensors are initially deployed, they have no knowledge about the neighbor sensors and environment condition. All sensors are initially

active. The mesh protocol is designed to let the cluster head broadcast self-organization messages periodically, and let the LNs forward the messages. Via this CH broadcasting and LN forwarding process, all the LNs finish the tasks of:

a) *time synchronization to the cluster head;*

b) *neighborhood survey;*

c) *local-level determination;*

d) *transmission route selection.*

2) Runtime phase. After the setting up phase, LNs are turned to online data collection status by receiving a command broadcasted from the CH. Once entering this status, each LN calculates and schedules its wakeup time based on its local-level, and turns to sleep mode immediately. Each LN sleeps for most of time to save energy and wakes up in the assigned slot to collect data and to transmit data to its parents along the transmission route. Its parents must have waked up to receive the message to work in cooperation. Therefore, the tasks in working phase include:

a) *Energy efficient scheduling;*

b) *Cooperative multi-hop communication;*

c) *Longtime route maintenance.*

To efficiently carry out above tasks to meet the requirements of scalability, longevity and adaptivity, two key innovative designs were proposed and developed in LayerMesh.

3.3.1. Distributed Dynamic Load-Balancing

Load-balancing is one of the keys to deal with the scalability and longevity challenges. **Figure 2** illustrates the load unbalance problem in multi-hop sensor network. The load of node A is six times of the load of node B, caused by the multi-hop load accumulation. Node A will die much quickly than the node B, although they are in the same level. This problem is general and serious when the network scale is large. Previous research has proved that the load-balance tree construction problem with deterministic link is NP-complete [11].

Our solution is to propose a distributed dynamic load-balancing algorithm. In this algorithm, the links are no longer deterministic, but probabilistic. At time *t*, a node *i* updates the traffic assignment probabilities to its parent candidates based on the following neighborhood information:

1) $L(t) = [L^t_{k,1}, L^t_{k,2}, \ldots, L^t_{k,n}]$, the current loads of parent candidates.

2) $P(t) = [P^t_1, P^t_2, \ldots, P^t_n]$, the traffic assignment probability from *i* to its parent candidates at time *t*. where *n* is the number of parent candidates. k means the level number. We have developed an algorithm using Equation (2) to update the transmission probabilities of the node *i*. Its convergence and optimality in load balance has been proved [12].

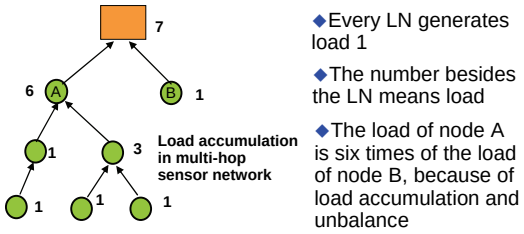


Figure 2. Load unbalance problem caused by load accumulation in multi-hop sensor network.

$$P_i^{t+1} = \frac{P_i^t / L^t_{k,i}}{\sum_{j=1}^{n} P_j^t / L^t_{k,j}} \tag{2}$$

Before transmission, the node *i* generates a random number *x* for route selection. It transmits data to parent candidate *j* if:

$$\sum_{c=1}^{j-1} P_c^{t+1} < x \le \sum_{c=1}^{j} P_c^{t+1} \tag{3}$$

3.3.2. Stair Scheduling

We have developed a level-based stair scheduling scheme to carry out energy efficient data collection and cooperative multi-hop communication. The key idea is to schedule the sensors based on their levels. As shown in **Figure 3**, sensors in each level spend most of time in sleeping. The sensor in level *k* will wake up one slot earlier than the sensors in level *k*-1. After waking up, each sensor will be active for only three time slots:

1) R-Slot, each sensor in level *k* listens to channel to receive data from its children (in level *k* + 1)

2) T-Slot, the sensor transmits the locally collected data and the forwarding data to its parents (in level k-1).

3) Syn-Slot, the sensor listen to the channel to receive the message from its parent and do time synchronization.

From the network point of view, the active slots of the sensors form a stair like scheduling scheme. The key point for stair scheduling implementation is the multi-hop time synchronization. In setup phase of LayerMesh, we use PulseSync[13] MAC stamping method to do multi-hop time synchronization. In stair scheduling, we also use a Syn-Slot to do online time-synchronization. In our experiments, the time synchronization error within 10 hops is less than 1ms. So the stair scheduling can be efficiently implemented. More details of collision avoidance and implementation of stair scheduling can be referred to [14].

4. L^{3}SN System Development

We developed prototypes of the L^{3}SN system, including the LNs, the gateways, and the information service plat-

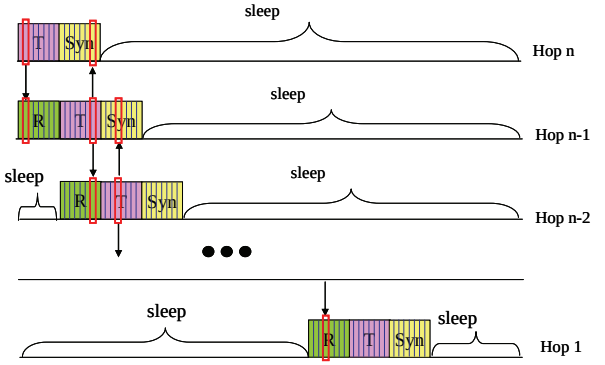


Figure 3. Level-based Stair Scheduling for energy efficient data collection and cooperative multi-hop communication.

form, and we have developed and tested a L^3SN system with 60 LNs and one gateway.

4.1. LN Node

Our LN is shown in **Figure 4**. It is composed by:

1) A solar panel for energy recharge, which can provide up to 200mw/hour energy supply in a sunny day.

2) Waterproof shell and four waterproof interfaces for easily plugging in of different agriculture sensors. The four interfaces are RS232, I^2C, RS485, Analog respectively.

3) A RF230 radio companied with 12dB antenna. The measured maximum transmission range is 300m, and the data rate is 250k bps.

4) The speed of MCU is up to 16MHz when power supply is enough. The MCU has 128K ROM, 8K RAM, and 10bit ADC.

5) We use air-temperature, air-humidity, soil-temperature (RS485), soil-humidity (RS485), light, and CO2 sensors which can be easily plugged into the LN.

The sensor-wireless interface converts the voltage values from sensors to meaningful temperature and humidity data. For example, the scope of voltage reading of the humidity sensor is 0- 1500mv; the measurement accuracy is 3%, and its transformation function is y = f(v) = 0.05071(v -106.9), therefore the interface of the humidity sensor is:

$$S(v) = \{0.05071^* (v-106:9); \{0; 1500\}; 3\% \} \quad (4)$$

4.2. Gateway

The gateway is composed by a sink sensor, a serial converter and a GPRS DTU [14]. The composition of the gateway is shown in **Figure 4**. The sink sensor use the same hardware as the LN, it is connected to GPRS DTU. The GPRS DTU provides transparent data forwarding from the sink sensor to the Internet. It supports standard TCP/IP protocol, and can be always online by heartbeat signal. The WSN-Internet interface is applied to the sink sensor, as explained in Subsection 3.2.

4.3. Weather Station

We have also developed weather station to monitor the weather information including wind speed, temperature, light etc. The weather station is built based on Davis's product and we

4.4 Information Service Platform

We developed the information service platform. The platform consists of a data collection daemon, a data base and a web server.

1) The daemon processes data reception from the GPRS DTUs, and stores the received data into the database.

2) We build the database using mySQL[14]. It maintains the sheets of LNs and gateways to store the historical and the real-time data.

3) The web server is implemented by JavaScript and Tomcat. It uses the Google Earth API to render the topology information of the sensor network, and supports graphical rendering of the real-time and historical data of the LNs and the gateways. The web server also supports various query forms for spatial and historical data analysis.

The snapshots of the web pages are shown in **Figure 5** and **Figure 6**.

5. Conclusions

We have design and developed L^3SN, a level-based, large-scale, longevous sensor network for precision agriculture information monitoring. It is composed by a two-tiered sensor network and an information service platform. In the low tier of the sensor network, a large amount of energy-limited sensor nodes (LNs) are deployed to capture and report information of their designated vicinity. In the high tier of the sensor network, some powerful GPRS gateways organize the LNs to form clusters and report the aggregated information to the Internet. The information service platform is de-

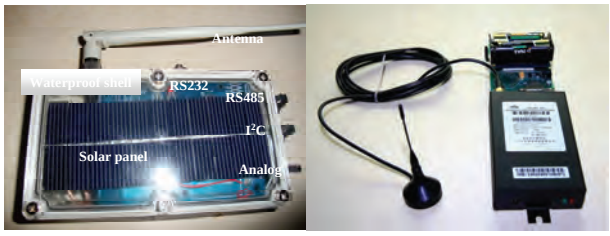


Figure 4. The prototype of LN node and the gateway.

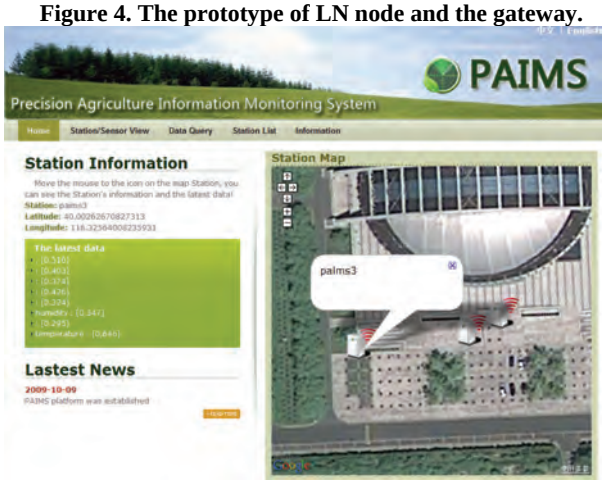


Figure 5. The prototype of LN node and the gateway.

Figure 6. The prototype of LN node and the gateway.

signed to log, render and analyze the temporal and spatial agriculture information to provide value-created services. We have presented the key design issues of L^3SN, including the system architecture, two standardized interfaces and the mesh protocol. Distributed dynamic load balancing and stair scheduling and adaptive routing are proposed to handle the large scale, longevous and adaptivity requirements.

In future work, we will 1) further improve stair scheduling to enhance the energy efficiency performance. The scheduling scheme will be made adaptive to the load of the sensors. So that redundant time slots can be saved to improve energy efficiency. 2) In the second stage, we plan to deploy more 300 LN nodes to Huantai City, Shandong Province to do larger scale field test.

5. References

[1] J. Yick, *et al*., "Wireless Sensor Network Survey," *Computing Network*, Vol. 52, No. 12, 2008, pp. 2292-2330.

[2] J. Burrell, T. Brooke, *et al*., "Vineyard Computing: Sensor Networks in Agricultural Production," *IEEE Pervasive Computing*, Vol. 3, No. 1, 2004, pp. 38-45.

[3] M. Demirbas and K. Y. Chow, *et al*., "INSIGHT: Internet-Sensor Integration for Habitat Monitoring," *Proceedings of International Symposium on a World of Wireless, Mobile and Multimedia Networks*, Buffalo, 2006, p. 558.

[4] R. L. Gomide, R. Y. Inamasu, *et al*., "An Automatic Data Acquisition and Control Mobile Laboratory Network for Crop Production Systems Data Management and Spatial Variability Studies in the Brazilian Center-West Region," ASAE Paper No. 01-1046, The American Society of Agriculture Engineers, Michigan, USA, 2001.

[5] J. Mahan, D. Wanjura, *et al*., "Design and Construction of a Wireless Infrared Thermometry System," *The USDA Annual Report*, Project Number: 6208-21000-012-03, 2004.

[6] "EKo Wireless Sensor Vineyard, Crop and Environment Monitoring System". http://www.xbow.com/eko/index.aspx.

[7] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler and J. Anderson, "Wireless Sensor Networks for Habitat Monitoring," *Proceedings of ACM International Workshop on Wireless Sensor Networks and Applications*, Atlanta, 2002, pp. 88-97.

[8] L. Selavo, A. Wood, Q. Cao, T. Sookoor, H. Liu, A. Srinivasan, Y. Wu, W. Kang, J. Stankovic, D. Young and J. Porter, "LUSTER: Wireless Sensor Network for Environmental Research," *Proceedings of ACM Senses*, Sydney, 2007.

[9] L. F. Mo, Y. He, Y. H. Liu, J. Z. Zhao, S. J. Tang, X. Y. Li and G. J. Dai, "Canopy Closure Estimates with Green Orbs: Sustainable Sensing in the Forest," *ACM Senses*, Berkeley, 2009.

[10] X. Y. Li, Y. J. Wang and Y. Wang, "Complexity of Data Collection, Aggregation, and Selection for Wireless Sensor Networks," *IEEE Transactions on Computers*, February 2010.

[11] C. Buragohain and D. Agrawal, *et al*. "Power Aware Routing for Sensor Databases," *Proceedings of International Conference on Computer Communications*, Miami, 2005, pp. 1747-1757.

[12] Y. C. Wang, Y. X. Wang and X. Qi, "Optimal Distributed Load Balancing in Multi-Hop Sensor Networks," *Technique Report*, 2010-06.

[13] C. Lenzen, P. Sommer and R. Wattenhofer, "Optimal Clock Synchronization in Networks," *Proceedings of Conference on Embedded Networked Sensor Systems*, Zurich, 2009, pp. 225-238.

[14] Y. X. Wang, Y. C. Wang, X. Qi and L. W. Xu, "OPAIMS: Open Architecture Precision Agriculture Information Monitoring System," *ACM CASES*'09, Paris, October 2009, pp. 233-239.

Wireless Sensor Network, **2010, 2, 661-667**
doi:10.4236/wsn.2010.29079 Published Online September 2010 (http://www.SciRP.org/journal/wsn)

The Safe Navigation of Partial Motion Planning Based on "Cooperation" with Roadside Fixed Sensors in VANET

Rong Ding[1,2], Xiaoguang Li[1,2]
[1]*State Key Laboratory of Software Development Environment, Beihang University, Beijing, China*
[2]*School of Computer Science and Engineering, Beihang University, Beijing, China*
E-mail: *dingr@buaa.edu.cn, lixiaoguang@cse.buaa.edu.cn*
Received July 3, 2010; *revised August* 5, 2010; *accepted September* 7, 2010

Abstract

In recent years, many methods of safe vehicle navigation and partial motion planning (PMP) have been proposed in vehicular ad-hoc network (VANET) field. In order to improve the limitation of traditional PMP, this paper presents a novel effective way to plan motion with cooperation of roadside fixed sensors (RFSs). With their cooperation, the vehicles can get the surrounding information quickly and effectively, and give highly accurate projections about the near future conditions on road. After proposing our algorithm, the worst case is analyzed and methods are found to solve the problem. Finally we conduct one elemental contrast experiment, driver situation awareness, with or without the "cooperation" of RFSs in highway scenarios. The result shows that the vehicles can make a better PMP based on the forward conditions received from RFSs, and extend the warning distance obviously when emergency happens.

Keywords: Partial Motion Planning, Safe Vehicle Navigation, Roadside Fixed Sensors, Cooperation

1. Introduction

It is a dream of human beings to achieve autonomous mobile vehicles since the invention of cars. With the rapid development of science and technology, and through the persistent and unremitting efforts by scientists, many novel tools which are equipped with modern technologies are used in cars such as GPS navigators and PSD sensors. By using those tools, unnecessary accidents can be avoided and destination can be achieved easily as well [1].

However, there is still a long way to make the dream come true. The tools are only used to human in order to make the driving safer, but it cannot take the place of human's driving so far. The 2007 DARPA Urban Challenge presented the fact that there are plenty of works to do if achieving fully autonomous driving especially in urban environment with the extremely complicated conditions, such as traffic lights and pedestrians, and safe navigation is also a huge problem.

In safe navigation field, reaching a goal and avoiding collisions are two main purposes and the analysis is carried out in detail [2]. So far, a proven method is the layer structure. In that structure, the safe navigation architecture can be divided into five layers as follows, mission manager, route planning, partial motion planning (PMP), low level components, and hardware layer [3]. Nowadays, with the GPS's development, the research emphasis has been changed to partial motion planning.

There are two main paradigms in partial motion planning [4]. They are the plans based on prior information and real time information. And the latter seems to be safer and more reliable. Several schemes have been presented firstly in mobile robots [5,6]. In real time model, the systems are obliged to make a relatively correct decision within a restricted time. If not, the vehicles may be in danger passively.

Nowadays, in vehicular ad-hoc networks (VANETs), cars can be equipped with communication device so as to exchange information about position, speed, traffic jams or other road conditions. Safe navigation is one of the VANETs applications. Vehicle to vehicle communications provide a new way of collision warning forwarding and intersection collision warning.

This paper is organized as follows: Section 2 briefly reviews the previous work and the limitation that is not resolved completely. And in Section 3, an effective improvement is provided: the "cooperation" with RFSs, with details of the algorithms to achieve "cooperation" in Section 4. Then contrast experiments are carried out in

Section 5 and Section 6 concludes the paper.

2. Previous Work and the Limitation

2.1 Previous Work

In this section, we discuss the related work on safe navigation of Partial Motion Planning.

On hazardous terrain, a fuzzy logic approach is advanced in [7]. In that paper, navigation strategy is comprised of three simple and independent behaviors: seek-goal, traverse-terrain and avoid-obstacle. Those three behaviors are combined through appropriate weighting factors in order to generate the final steering and speed commands. The weighting factors are produced by fuzzy logical rules that are taken the current status into account. Also in [8], they employ fuzzy reasoning to allow for aggregation decisions based on a flexible and extensible set of criteria. These criteria can be application specific and enable a dynamic fragmentation of the road according to the applications' requirements.

In order to acquire safe motion planning, Inevitable Collision State (ICS) has been characterized by [9]. The most important contribution of ICS is that it promotes a dynamic relationship properly between collision states and safety states. Thus we can guarantee stronger safety and giving a safe partial motion planning.

In [4], ICS takes into account the dynamics of both the system and the moving obstacles. The core of safety issues is computing ICS-free partial motion in the shortest period of time. The paper [4] presents a property in order to reduce the complexity of the PMP algorithm, which simplifies the safety checking of a trajectory. If one partial trajectory is an ICS-free partial motion, the trajectory is collision free.

Another safe navigation technology is using virtual stick [10]. They propose a reliable method for navigating the robot to the target point, and draw an environment map using infrared sensors. However, it is difficult to distinguish distance information using only one-dimensional image information. So they use a virtual stick that represents different distances with different colors to improve the perception of the position and distance of the obstacle.

2.2 Limitation

All above methods only concern the one-car situation which is myopia. We cannot get enough information of the driving environment only by sensors or measure infrastructure equipped on a single car, especially in the complex and volatile urban scenarios. Each car needs to be kept in contact with others nearby. However, the number of cars is huge and they are all in motion. Those external factors have made it more difficult to ensure the reliable connections.

One significant reason of "partial" lies in the real-time limitation in PMP. In an environment featuring moving objects, you have to decide upon the further action within limited time, otherwise you would take the risk of being hit by another moving object. Getting the conditions of front road sections in time can short the PMP processing time.

In addition, a prior information updater does not exist. Even if there was one, the prior information would not update in time. Prior information would be collected by single car with the whole system in a lack of sink, as a result, cars would fail to send what they have already collected and therefore could not acquire front road condition.

3. An Effective Improved Method: the "Cooperation" with RFSs

In this part, we propose a novel approach to solve problems mentioned above. We introduce "cooperation" with RFSs into PMP. **Figure 1** shows the structure of "cooperation". Cars send and receive messages from RFSs, and RFSs can transmit these messages to each other. The latest technology is applied to enable every single car to detect its own possible collision, but ignore the transmission among cars. In a word, messages are transmitted from car to RFS, from RFS to car, and RFS to RFS. In Section 4, the three types of messages will be expounded in detail.

Traditional safe navigation of partial motion planning is based on single cars. In their opinion, if one car manages to do well in collision avoidance, the whole work would be finished. However, that is very shortsighted. One successful navigation even one successful partial motion planning depends on the accuracy grade of real-time information and front-road condition. In this case, RFSs are used as sink, and they can aggregate messages from cars and transmit warning messages or other significant messages from each other. In this way, as is shown in **Figure 1**, for example, Car5 can hardly get known about the condition of Car1 by itself when Car1 is collided, because Car3 has covered Car5's sight. But collision messages can be transmitted from RFS1 to RFS2, and then to RFS3, and at the same time both RFS2 and RFS3 send warning messages to cars nearby. In this way, Car5 manage to acquire the front condition.

4. Algorithms to Achieve "Cooperation"

In this section, we give an intimate design of three types

of messages mentioned in Section 3, then put forward our novel algorithms in dynamic road scenarios, and at the end advantages of this cooperative methods will be summarized.

4.1 Design of Three Types of Messages

As is shown in Section 3, the three types of messages are from car to RFS (C-R), from RFS to car (R-C), and RFS to RFS (R-R). The basic message formats are shown in **Figure 2**.

As is shown in **Figure 2**, there are three types in this paper, 0 means C-R, 1 means R-C, and 2 means R-R. Len means the length of data segment. CRC is cyclic redundancy check, and it is used to check whether data transmission is true or not.

C-R messages are the source knowledge of the system. Cars can get information from sensors which are attached to them. In this paper, attention is paid to the danger cars may meet. Each C-R message's source address is car's own ID, when moving into a new road section, cars will receive RFS's broadcast message. Immediately after cars get RFS's ID in source address of RFS's broadcast message, then cars begin to send C-R messages. Data segment of C-R messages contains the following information: car's velocity, acceleration, distance from forward obstacle, and some flag bit for RFS's judging safety, such as danger flag, ICS flag and emergency flag.

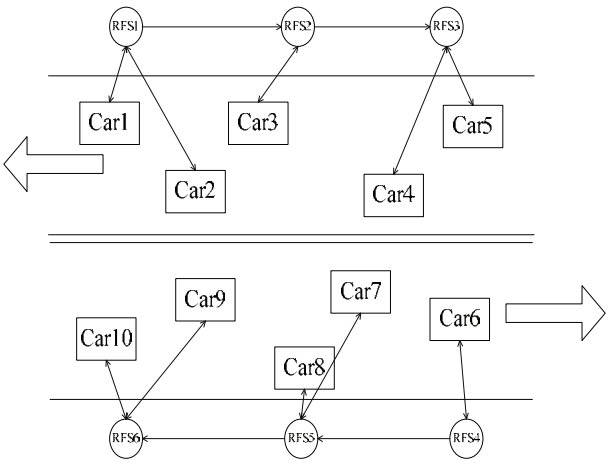


Figure 1. The structure of "cooperation" with RFSs.

SRC ADDRESS	DST ADDRESS	Type	Len
Data			CRC

Figure 2. Basic message format.

R-C messages are always regarded as prior information, as RFSs sends warning alarm or motion suggestions based on conditions of front section. Therefore, source address of R-C is RFS's own ID, messages are sending in flooding way thus any cars which is in this section can get R-C messages. Therefore, destination address should be defined as "1" sequence. Data segment of R-C messages consist of following information: forward section cars (if in the middle of the road), forward intersection queue length (if in urban scenarios), forward highway exit queue length (if in highway scenarios). The most important data is flags of warning alarm and motion proposal. These flags are concerned of safety of drivers and passengers' lives and property.

R-R messages are the information exchanged between each RFS on roadside, using cable transmissions in order to make sure the reliability while transmitting. Both source address and destination address are RFS's ID. We specially point out that source address is RFS's own ID, and destination address is upriver RFS's ID. That is because in traffic flow, if one car has passed the scene of accident, that car has no relationship to this accident. Data segment of R-R messages consist of following information: emergency state and RFS's ID of accident site (if has), numbers of passing cars in last 10s.

Based on the elementary design of message formats above, the algorithm of "cooperation" in dynamic scenarios will be discussed next.

4.2 Algorithm of "Cooperation"

"Cooperation" can be used similarly between highway scenarios and urban scenarios, although the two scenarios are very different.

In highway scenarios, all the obstacles are cars, thus the participators in the whole system are simpler and it is easier to monitor data and administer them. However, we cannot ignore the accidents happening on the highway. The velocity of each car reaches 120 kilometers per hour, that is to say, an ordinary car may run 10 meters or more in the time of a blink of an eye. Without a reliable warning system, it is very dangerous for those following cars while one car may suddenly have an accident. While in urban scenarios, obstacles can be cars, motorcycles, bicycles and pedestrians. Thus system must be very complex and it is more difficult to monitor a very exact real-time data. However, we should supply a reliable service to decrease the number of accidents, and cooperation with RFSs can provide all the cars around front road conditions and cars can analyze messages they have received and control their speed itself, and once they meet an accident forward they can take measures immediately. **Figures 3(a) (b) (c)** shows the brief flowcharts of algo-

rithm for the RFSs at the intersections or highway exit, on the middle of the road and for the cars respectively.

As is shown in **Figure 3(a)**, there is no prior RFS for the sensors who are settled at the intersections or highway exit. So they can only send interaction or exit messages to cars nearby in flooding way, thus they can receiving C-R messages as feedbacks. Then analyze the received message and judge whether there exists an emergency flag. If so, then send emergency messages to the cars nearby and send danger message to the next RFS; if not, just record this C-R message in the buffer and regularly (10s, *e.g.*) send road condition messages to next RFS based on the information recorded last period of time.

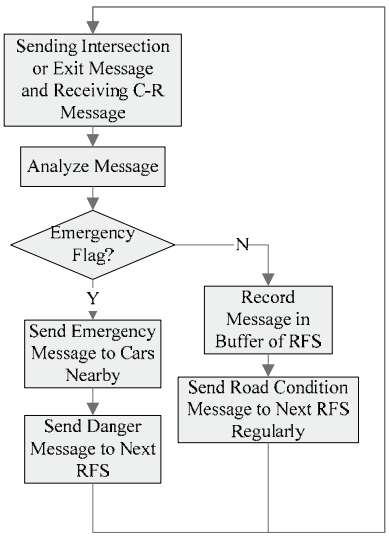
In **Figure 3(b)**, we can observe that there are more assignments for those RFSs in the middle of roads. First they have to listen to receive messages so as to get information from prior RFS and once they have received R-R messages they can make a judgment whether danger flag is included. If so, they send warning message to cars nearby and send danger messages to next RFS in addition if original danger message is from prior RFS to make sure danger messages can send at least three RFSs to extend warning area. Their other assignments such as receiving R-R messages exclude danger flag; receiving C-R messages and dealing with C-R messages are same as RFSs at the intersections or highway exit.

At last, we analyze **Figure 3(c)**, which is cars' behavior. When drivers are driving into a new road section, what they can do is driving steadily and safely. But they have no idea of front road condition; as a result, they have to receive information passively. In our algorithm, just like a seasoned driver, cars receive messages from RFS around, and when cars get R-C message, analyze it and record RFS's ID so as to make contact with the nearest RFS, then check whether there exists emergency flag, if so, brake the car immediately, and if just exists warning flag, that means in the front section of the road, there happens an accident, so decelerate the car immediately, and else just control car's velocity as the case may be. It is worth notice that once the car meets an accident, an emergency brake should be carried out by the automatic control immediately, send collision message at the same time. Thus we can make sure the RFS can get road condition and notice other cars in time, and then the car have to wait for aids.

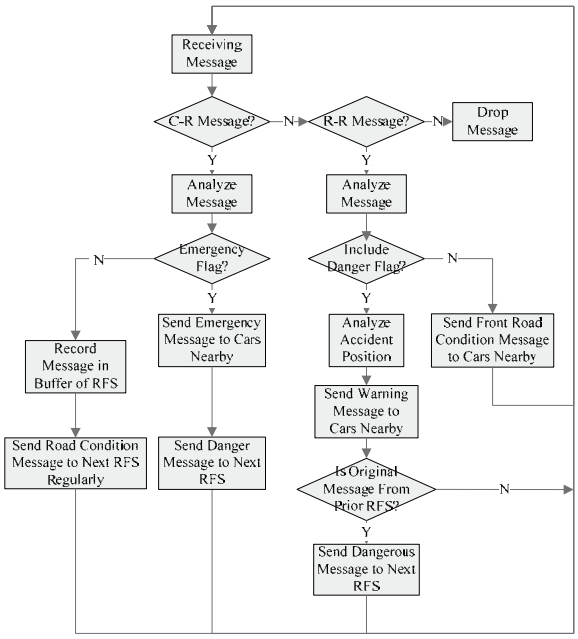
4.3. Worst Case

We must realize what the worst case is, and we should make measures to overcome the trouble. In this system, we enumerate three bad cases as follows, the failure of RFS, the failure of Cars and substantial cars on one section. In this part, we discuss the method to solve them.

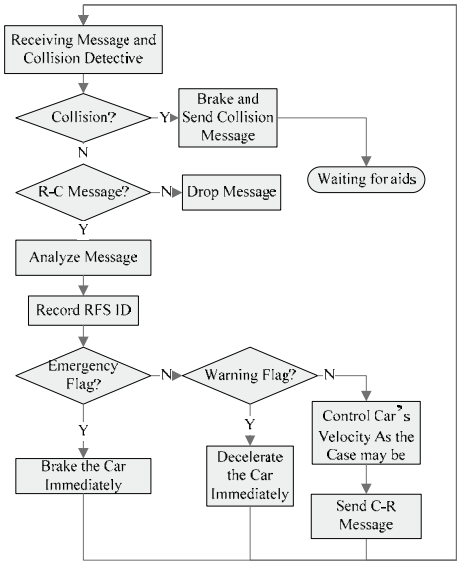
Once the RFS lost efficacy and the supervision on the road section may be missed. While one car runs into a new section, and RFS of that section is right out of work, it cannot get R-C messages and once it gets a traffic accident, it cannot be helped to send emergency messages and as a result, "cooperation" does not work and cars behind the traffic accident scene will not receive any warning message.



(a) Brief flowchart for the RFSs at the intersections or highway exit.



(b) Brief flowchart for the RFSs on the middle of the road.



(c) Brief flowchart for the cars.

Figure 3. Brief flowcharts of "Cooperation" with RFSs.

Also, it is the same to cars, once cars cannot send their emergency messages or they cannot receive R-C messages from RFS, it will be very dangerous for either those failure cars or cars behind them.

Another criticality factor of this system is vehicles' number, we have to take into account that the bandwidth and the considered RFS's capability. While cars' number is very huge and thus bandwidth is not enough, and emergency messages may be drop in RFS's queue, which means it is blind of RFS if traffic accident happens around, and that case is not allowed absolutely.

And our solutions are listed as follows. First, we prepare spare RFSs and Car sensors in order to eliminate the error coursed by failures happen on sensors. We have to take the cost if the whole system into account, if we prepare too much sensors, it is a waste of money, so there is a dynamic balance between safeties and costs. Here we choose the scheme that each car has a spare sensor, and every four RFSs have a spare sensor. Second, we use priority queue to make sure emergency messages are not dropped by RFS while the condition of communication is in a dickey state. In [11], priority concept is brought forward firstly in vehicle ad-hoc network field. And in this system, we apply this comparatively mature idea, and we give some improvement to satisfy the need of safe navigation.

5. Contrast Experiments

In this section, we introduce our contrast experiments. Firstly we introduce the experimental platform, and then our contrast trial in highway scenarios will be shown in the second part, at the end of the second part, we give our analysis of this experiment.

5.1. Experimental platform

As is shown in **Figure 4**, we use HBEROBOCAR as our experimental platform. **Figure 4(a)** shown the car and **Figure 4(b)** shows the RFS. All the messages of this system are transported by ATmegal128L just as **Figure 4(b)** shows. Our car equips some sensors, such as PSD sensors and ultrasonic sensors so that it can detect forward obstacles, and also it quips one ATmegal128L interface so that it can communicate with RFSs while connecting with ATmegal128L node.

This car is 23 cm long and 15 cm wide, and 1:18 to the true car. The highest speed of the car is 47 cm/s. In this experiment, we pay our attention to the different speed after 60 cm of the traffic accident happens and warning distance between with and without RFSs.

5.2. Experiments in Highway Scenarios

We apply four cars and two RFSs to simulate the highway scenarios as is shown in **Figure 5**. Car1 is the first car which would have an artificial accident thus we can measure car2 to car4's reaction time and stop distance.

(a) One experimental car with PSD sensors and ultrasonic sensors.

(b) RFS sensor used for communication

Figure 4. HBE-ROBOCAR experimental platform.

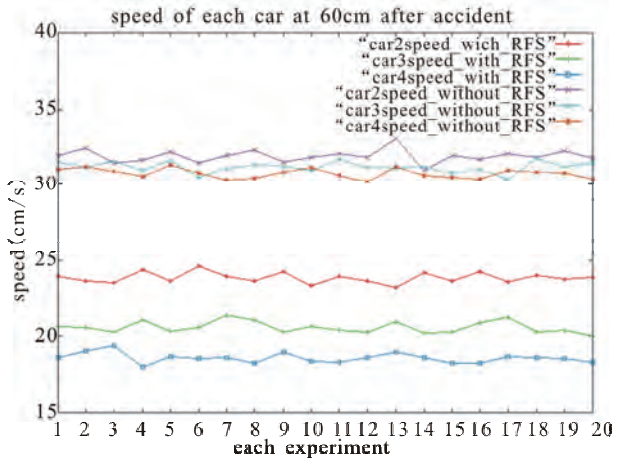
Figure 5. The simulative scenarios of our contrast experiments.

At First, we do not use RFSs and only four cars driving on the simulative scenarios. Suddenly the car runs upfront meet an accident, and we measure each car's speed when car runs into the 60 cm of the traffic accident and warning distance when accident happens. We will repeat for 20 times.

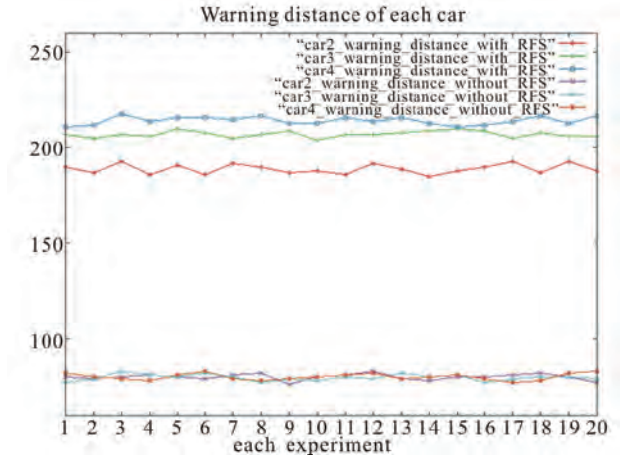
Then we add RFSs and with three RFSs and four cars, then we let the first car to have an accident just like the experiments above, then measure each car's speed and warning distance too. Also, we make the experiments for 20 times.

The results are shown in **Figure 6** and consolidated measured data are shown in **Table 1.**

As is shown in **Figure 6** and **Table 1** above, while adopting "cooperation" with RFS strategy, each car's speed at 60 cm after accident is much slower and the warning distance is much wider than the case without RFSs. Car2 is in front of Car3 and Car4, thus Car2's speed at 60 cm after accident is a little higher than Car3 and Car4. However, we can clearly find that the performance is improved obviously.



(a) Speed of each car at 60cm after accident.



(b) Each car's warning distance

Figure 6. The result of contrast experiments of with and without RFSs.

Table 1. Average data of the contrast experiments.

cars	Average data of measured parameter	without RFSs	with RFSs
car2	speed (60cm)(cm/s)	31.93	23.86
	warning distance(cm)	79.95	189.05
car3	speed (60cm) (cm/s)	31.22	20.59
	warning distance(cm)	79.65	207.15
car4	speed (60cm) (cm/s)	30.77	18.58
	warning distance(cm)	80.05	214.40

6. Conclusions and Future Work

In this paper, it is shown how to develop a novel safe navigation algorithm based on "cooperation" with RFSs, so as to improve the safety factor for the drivers while driving cars. Our experimental results show our algorithm effectively reduces the chain collision and extends the range of divers' situation awareness.

In the future, we focus our research priority on how to create a novel transport protocol so as to overcome wicked condition such as very high speed, too many cars and atrocious weather.

7. Acknowledgements

Supported by the Research Fund for the Doctoral Program of Higher Education of China(RFDP), the fund of the State Key Laboratory of Software Development En-

vironment (Grant No. SKLSDE-2009ZX-04) and the National High-Tech R&D Program of China (863 Program) (Grant No. 2009AA043303, 2008AA12A216, 2009AA043305).

8. References

[1] G. Taylor, G. Blewitt, D. Steup, S. Corbett and A. Car, "Road Reduction Filtering for GPS-GIS Navigation," *Transactions in GIS*, Vol. 5, No. 3, 2001, pp. 193-207.

[2] R. Siegwart and I. R. Nourbakhsh. "Introduction to Autonomous Mobile Robots," MIT Press, Massachusetts, 2004.

[3] K. Macek, D. Vasquez, T. Fraichard and R. Siegwart, "Safe Vehicle Navigation in Dynamic Urban Scenarios," *Proceeding of the* 11*th International IEEE Conference on Intelligent Transportation Systems*, Beijing, October 2008, pp. 482-489.

[4] S. Petty and T. Fraichard, "Safe Motion Planning in Dynamic Environments," *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, Edmonton, 2005, pp. 2210-2215.

[5] P. Fiorini and Z. Shiller, "Motion Planning in Dynamic Environments Using Velocity Obstacles," *International Journal of Robotics Research*, Vol. 17, No. 7, July 1998, pp. 760-772.

[6] R. Simmons, "The Curvature Velocity Method for Local Obstacle Avoidance," *International Conference on Robotics and Automation*, Minneapolis, April 1996, pp. 3375-3382.

[7] H. Seraji, A. Howard and E. Tunstel, "Safe Navigation on Hazardous Terrain," *Proceedings of the* 2001 *IEEE International Conference on Robotics & Automation*, Seoul, May 2001, pp. 3084-3091.

[8] S. Dietzel, B. Bako, E. Schoch and F. Kargl, "A Fuzzy Logic Based Approach for Structure-free Aggregation in Vehicular Ad-Hoc Networks," *Proceedings of the* 6*th ACM international workshop on Vehicular Internetworking*, Beijing, September 2009, pp. 79-88.

[9] T. Fraichard and H. Asama, "Inevitable Collision States —A Step towards Safer Robots?" *Advanced Robotics*, Vol. 18, No. 10, 2004, pp. 1001-1024.

[10] S. M. Jung, T. H. Song, J. H. Park, J. H. Park and J. W. Jeon, "The Safe Navigation of Remote Mobile Robot Using Virtual Stick," 2008 *IEEE International Conference on Industrial Technology*, Chengdu, April 2008, pp. 1295-1734.

[11] T. M. Marc, D. Jiang and H. Hartenstein. "Broadcast Reception Rates and Effects of Priority Access in 802.11—Based Vehicular Ad-Hoc Networks," *Proceedings of the* 1*st ACM International Workshop on Vehicular Ad Hoc Networks*, Philadelphia, October 2004, pp. 10-18.

Wireless Sensor Network, 2010, 2, 668-674
doi:10.4236/wsn.2010.29080 Published Online September 2010 (http://www.SciRP.org/journal/wsn)

A Real-Time Urban Traffic Detection Algorithm Based on Spatio-Temporal OD Matrix in Vehicular Sensor Network*

Ke Zhang, Guangtao Xue
Department of Compute Science & Technology, Shanghai Jiao Tong University, Shanghai, China
E-mail: *xue-gt@cs.sjtu.edu.cn*
Received July 4, 2010; *revised August* 7, 2010; *accepted September* 10, 2010

Abstract

Currently, there are kinds of algorithms in order to detect real-time urban traffic condition. Most of these approaches consider speed of vehicles as a main metric to describe traffic situation. In this paper, we find out two important observations through several experiments. 1) In urban city, the speed of vehicles is influenced significantly by some factors such as traffic lights delay and road condition. The actual situation rarely satisfy hypothesis required for these solutions. Therefore, these traditional algorithms do not work well in practical environment. 2) Traffic volume on a road segment shows strong pattern and changes smoothly at adjacent time. This feature of traffic volume inspires us to define a metric: traffic-rate, which is used to detect traffic condition in real time. In our solution, we develop a novel traffic-detection algorithm based on original-destination (OD) matrix. We illustrate our approach and measure its performance in real environment. The performance evaluations confirm the effectiveness of our algorithm.

Keywords: KNN; Linear Least Square; OD Matrix; Traffic Status Detection

1. Introduction

Nowadays, traffic congestion has been a serious problem in many urban cities. In China, it caused about 5%-8% GDP wasted each year. Therefore, governments cost million dollars to build Intelligent Transportation System (ITS). In response to these challenges, Original-Destination (OD) matrix is often required as application input to provide service for transportation. OD matrix is a non-negative matrix $f(i, j)$ which represents volumes of traffic from a source i to a destination j. In a road network, element $f(i, j)$ in the OD matrix means the number of vehicles from road segment i to road segment j.However, traditional algorithms to construct OD matrix do not work well in real world, since their solutions do not consider features of real traffic condition. Furthermore, the traditional algorithms of traffic detection have limitations when runs in real environment. In their solutions, speed of vehicles is considered as a main metric to identify traffic states. Nevertheless, we find that speed is influenced significantly by traffic lights.

In this paper, we study this problem using GPS devices which have been equipped in taxis and buses in Shanghai. All GPS vehicular data are transferred to the local APs which are deployed in intersections, and GPS reports would be sent to a data center, where we run our algorithm to construct OD matrix and analyze traffic condition. We receive a great support from Shanghai government and have deployed a cost-effective system to collect traffic information. However, there are still lots of challenges to process information in a metropolis like Shanghai.

First of all, due to vehicular GPS signal is always varying, all GPS location data we collected are not only large and noisy but not uniformly distribute. Some areas near downtown produce most of traffic volumes. **Figure 1** shows millions of GPS reports on each road from January 5 to January 20 in 2007. In addition, according to our statistics, almost 30% vehicles lost their GPS reports. Besides, the solution must consider the fact that not every vehicle has installed GPS device.

Although taxies and buses have installed vehicular wireless communication devices whose communication range is about 200 meters, it is costly if we deploy AP every 200 meter. Thus, we need to interpolate traffic volumes on road segments that do not have AP. Fur-

*Supported by the National Natural Science Foundation of China under Grant No. 60970106, the National Grand Fundamental Research 973 Program of China under Grant No.2006CB303000, and Key Program of National MIIT under Grant 2009ZX03006-004.

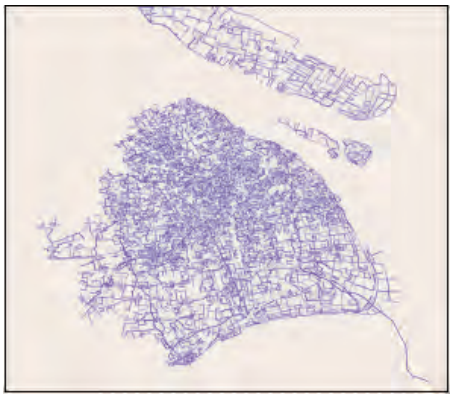


Figure 1. GIS map of Shanghai with spatial distribution of GPS reports from January 5 to January 20, 2007.

thermore, because of errors of GPS data (the range of the error is about 20 meters to 100 meters), we have to firstly apply our own GPS-correcting algorithms to abandon "dirty" data and amend incorrect ones.

The last but not the least, it is difficult to capture every factor which has effect on traffic condition. For instance, traffic lights delay and complicated condition of road have significant impact on traffic condition. Meanwhile, these factors are difficult to be estimated using a simple model. This is why most proposed algorithms do not work well in real life.

Fortunately, we find out a spatial-temporal model that is able to overcome these limitations to reconstruct OD matrix. In addition, we define a metric: Traffic-Rate, which is used to describe current traffic condition. We are going to discuss them in Subsection 2.5.

The remainder of this paper is organized as follows: Subsection 1.1 provides background and compares our approach with related works. We then introduce our spatial-temporal algorithm and traffic-detection solution in detail in Section 2. We demonstrate our evaluation result in Section 3. Finally, we present conclusion in Section 4.

1.1. Related Work

Due to importance of OD matrix, there are a number of works focus on how to estimate OD matrix accurately in recent years [1-3]. However, most of their solutions work on highways or freeways, where traffic lights timing and behavior of drivers are not issues because there are no intersections and vehicles' routes followed a certain way. On the other hand, traffic situation of an urban is so complicate that the technical conditions which required for these solutions are difficult to be satisfied. Yin Zhang *et al.* [4] illustrate a novel algorithm to estimate a traffic matrix in Internet whereas our solution runs in the ***V**ehicular **S**ensor **N**etwork* (***VSN***), more importantly, they do not give the fact to prove their observation.

With rapid evolvement of intelligent traffic system (ITS), a lot of efforts focus on monitoring traffic and incidents detections installing sensors like camera on roads [5]. From a practical perspective, these approaches need large sums of money to deploy numbers of sensors. Moreover, Lin *et al.* [6] and Coifman *et al.* [7] propose schemes to detect traffic condition by installing traffic detectors. Meanwhile, their solutions work on freeway not an urban area. How to define an appropriate metric to describe traffic condition is also a hot topic for researchers. For example, Jungkeun Yoon *et al.* [8] analyzed main factors which influence traffic states. Furthermore, some researchers [9-11] propose interesting schemes to estimate velocity of vehicles to reflect traffic condition. However, they do not take condition of roads into account. Therefore, their solutions have limitations when runs in real environment.

2. Spatio-Temporal Original-Destination Matrix

In this section, we give a brief introduction to OD matrix first. Then we present our algorithm in detail. This construction algorithm consists of two important rules that are observed by experiments. We take advantage of OD matrix to implement traffic-detection solution.

2.1. Original-Destination Matrix

For a road network with N intersections which connect M road segments, the OD matrix is a square $M \times M$ matrix. In this paper, the element $f(i, j, t)$ in OD matrix represents traffic volumes from $road_i$ to $road_i$ during $[t, t + \Delta t)$. In order to get more convenience for calculation, we convert OD matrix as a 2-dimensional array, where columns represent OD matrix at different time while the rows represent the evolution of traffic volumes in each route.

2.2. The Spatial Component

After extracting and analyzing millions of real vehicles GPS reports, we find the first important fact that there are strong correlations of traffic volumes among road segments. In practice, road i to some extent has similar traffic flows with road j.

As is shown in **Figure 2**, we choose K (K = 4) road segments which have the most similar traffic-rate (the rate represents throughput of traffic in a road) evolution for each road segment. Finally, we find out that most neighbors consist of a linear combination of the target road segment. In our solution, we develop this discovery by using an approach: KNN (k-Nearest Neighbors).

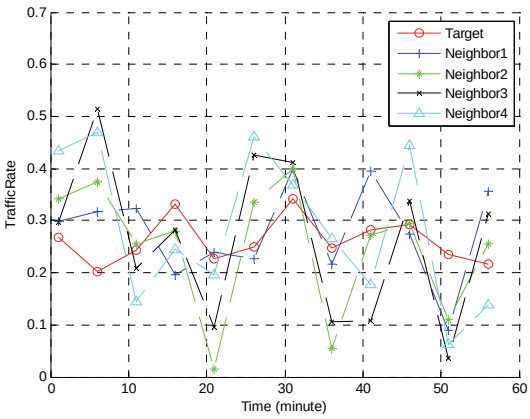


Figure 2. Traffic-rate comparisons between 4 neighbors and target road.

The KNN algorithm to calculate correlation is given as follows:

1) A GPS dataset is used as the train dataset to construct a complete OD matrix.

2) Any row in the OD matrix could be considered as a vector. We take advantage of (1) to calculate similarity of vectors. Always, we set K = 4 that a road segment has four neighbors.

3) We use linear regression to find a vector of weights $\omega(k)(k=1,2...K)$ so that $road_i$ is expressed by combination of $road_k$. In (1), $road_i$ represents the ith row of OD matrix.

$$Correlation(roadi, roadk) = \cos\theta = \frac{roadi \bullet roadk}{\| roadi \| \times \| roadk \|} \quad (1)$$

$$road_i = \sum_{k=1}^{K=4} \omega(k) road_k \quad (2)$$

Even though we use just one train dataset to find neighbors for each road segment, the neighbor relations of the road segment can still be hold for other dataset.

2.3. The Temporal Component

As is show in the **Figure 3**, the left picture shows a distribution of traffic-rate from 11:00 p.m to 13:00 p.m during one day. We interpolate these values and present them in the right picture, which implies that traffic rate changes in a small range (less than 0.1). Thus there is the second important truth: The evolution of traffic volumes on a road segment can keep "stable" during a time interval, which is based on the fact that a number of vehicles do not appear suddenly in the same time at same place, namely $f(i,j,t) \approx f(i,j,t+\Delta t)$.

Meanwhile, from the second fact, we can construct a

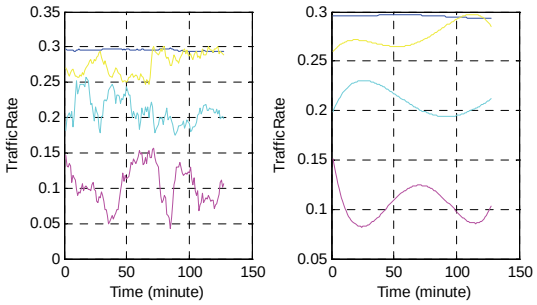


Figure 3. Traffic-rate in 4roads within 2hours.

temporal constraint matrix TM to show the smoothly changes of traffic volume on each road.

The construction of *TM* matrix is simple, and the form of *TM* matrix is given as follows:

$$TM = \begin{pmatrix} 1 & -1 & 0 & \cdots \\ 0 & 1 & -1 & \ddots \\ 0 & 0 & 1 & \ddots \\ \vdots & \ddots & \ddots & \ddots \end{pmatrix}$$

2.4. Construction of OD Matrix

Linear Least Square procedure is often used when constructing OD matrix in industry. Constructing OD matrix is modeled as follows:

$$Q = BX \quad (3)$$

where Q represents total traffic volumes on the road segment and can be easily measured by device like traffic detector [6,7]. In (3), where B is a topological matrix which represents the connectivity of each road segment in an area. Our goal is to find X which is the OD matrix. In practice, B is often a low-rank matrix. Hence, there are probably several solutions to this equation so we have to answer two questions as follows:

1) Which solution is the best?

2) What constraints should we add to estimate X more accurately?

As we mentioned above, B and X are sparse matrices. In implementation, we set a rank value as input and pick the solution whose rank is less than our input as final result.

Through analyzing several experimental results, we are inspired from the spatial component that each road can be linear expressed by its neighbors, so we can take advantage of KNN to build a spatial matrix SP to express which rows in the OD matrix are neighbors.

The construction of SP is given as follows:

1) Use KNN method to find the set of weights for each

road segment.

2) For k = 1, 2, 3,…K, we let $SP(i,i)=1$ and $SP(i, j_k)=-\omega(k)$

Then, a spatial constraint equation is given as follows:

$$SP * X = 0 \quad (4)$$

Meanwhile, according to the temporal feature, the temporal constraint equation is given as follows:

$$TM * X = 0 \quad (5)$$

Now, we have acquired two constraint equations through capturing features of traffic volume. In our system, we use standard *Linear Least Square* (we develop this procedure in Matlab) which is based on (3), (4), (5) to estimate X .

2.5. Real-Time Traffic Condition Description and Alarm

How to describe traffic condition is a meaningful but difficult problem which always fascinate researchers [9-11], who proposed variety of algorithms to try to predict speed of vehicles. Intuitively, a fast transit velocity implies a good traffic condition. However, another non-negligible is that speed can be influenced by many factors which are difficult to measure, such as behavior of drivers, traffic lights delay and condition of road.

As is shown in **Figure 4**, we select a taxi randomly and display its changes of speed during an hour. We can see that the taxi stop when t = 17 min and runs slow at some time. These are normal phenomenon because of traffic lights which have a significant impact on speed. On the other hand, as is described in **Figure 4**, speed does not have a regular pattern to predict in reality.

Instead, we propose a solution to capture the traffic status and broadcast warning message to avoid traffic congestion. We do not just use traffic volumes as a metric to identify traffic status. Because different road has different number of lanes and length, it is unreasonable to use traffic volume to identify traffic jam in such cases.

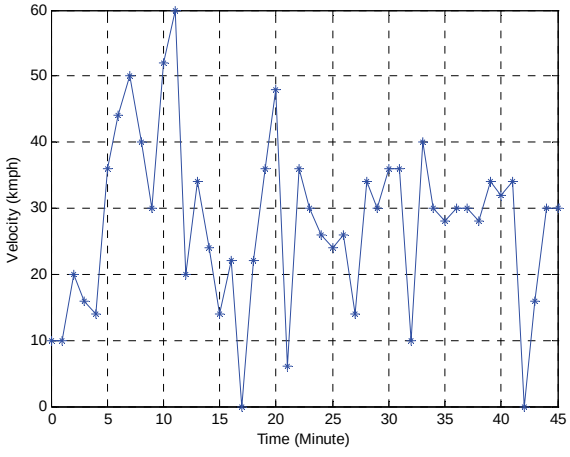


Figure 4. Speed changes during an hour.

Considering different condition of roads, in this paper, we define "traffic-rate" R_t as a metric to describe current traffic condition. The number of lanes and length of road are taken into account in our design.

We calculate R_t using following:

$$R_t = \frac{S_{in}+1}{S_{in}+S_{out}+1} \quad (6)$$

where S_{in} expresses the number of vehicles that enter into the road during $[t, t+\Delta t)$, S_{out} expresses the number of vehicles that leave the road during $[t, t+\Delta t)$.

Meanwhile, we assign a threshold λ for each road segment. From a simple perspective, if we find $R_t > \lambda$, we think current traffic status is changing. Before we discuss how to classify traffic status in detail, we give a solution to calculate λ dynamically.

There is a traditional solution to calculate λ : we can analyze the past GPS reports and take advantage of a maximum a posteriori (MAP) classification with both likelihood functions and a priori probabilities to calculate λ for each road segment. But this solution cost too much time because of thousands of road segments. More importantly, it does not have enough flexibility.

Now, we propose an experimental formula below to calculate λ dynamically.

$$\lambda = \frac{L \times N}{V(T+D)} \quad (7)$$

where V expresses the mean speed of vehicles on the road and T is an estimation value which can be calculated by using GPS information to show how long a taxi may stay in the road. In (7), where D is the traffic light duration, L represents length of the road and N represents number of the lanes. Since on a road, the bigger $L \times N$ is, the more vehicles the road can contain.

However, we observe that traffic light duration can be calculated as follows:

$$D \approx \frac{L}{V} \times S_{out} \quad (8)$$

where L , V and S_{out} have the same mean we mentioned above. Equation (8) has an intuitive mean that vehicles should leave the road in traffic light duration.

Then, (7) is equivalent to

$$\lambda = \frac{N}{1+S_{out}} \quad (9)$$

Rather than previous studies, we do not simply identify two traffic states: good and bad. In our design, we divide current traffic status into five states: FREE,

NORMAL, ALERT, BUSY and OVERLOAD. Since we must send warning messages of traffic condition to vehicles before traffic condition becomes worse. Another benefit is that we can make flexible policies when we are processing traffic information. The rules to identify traffic status on a road are given as follows:

1) $0 \le R_t < \lambda$, the traffic state is FREE;
2) $\lambda \le R_t < 2\lambda$, the traffic state is NORMAL;
3) $2\lambda \le R_t < 3\lambda$, the traffic state is ALERT,
4) $3\lambda \le R_t < 4\lambda$, the traffic state is BUSY;
5) $4\lambda \le R_t < 1$, the traffic state is OVERLOAD.

In our experiments, $\lambda \approx 0.23$ for most cases. When we detect that current traffic state is ALERT, we start to send warning message to those related vehicles around the road.

3. Experiments

3.1. Experimental Setup

We have collected a great number of real traffic data which are generated in real time by installing GPS devices in taxis and buses from data center in Shanghai. Even we do not collect all vehicular GPS reports since some vehicles do not install GPS devices, the evaluations also show that our solution can work well.

The fields (id, longitude, latitude, velocity, angle, timestamp) denote the GPS reports, where id identifies a taxi, the pair (longitude, latitude) shows current coordinates of the taxi, timestamp is the time report, the pair(velocity, angle) shows current speed and driving direction of the taxi, which is used with Shanghai map to calculate the short-term destination in our algorithm.

The basic experimental parameters are given as follows:

3.2. Methodology

In order to demonstrate the effectiveness of our solution, we have developed a mobility model to generate vehicular GPS reports in a target area. The mobility model consists of three parts:

1) We load GIS map of Shanghai to generate topological matrix.

2) We set source and destination coordinates for each node. Traffic light duration is set according to surveys.

3) The nodes send GPS reports randomly and some nodes do not send. Moreover, for making our simulation close to real life, we also insert real GPS traces into our test dataset.

Our initial OD matrix lost elements at some positions and some elements have error in their value.

At last, we measure performance using the Normalized Mean Absolute Error [4].

$$NMAE = \frac{\sum_{i,j:M(i,j)=0} |X(i,j) - \hat{X}(i,j)|}{\sum_{i,j:M(i,j)=0} |X(i,j)|} \quad (10)$$

3.3. Performance

Figure 5 shows the performance of our algorithm ST and comparison between SRSVD Base [4], KNN, and ST when GPS reports lost randomly. The default parameters have been listed in **Table 1**. We can clearly realize that ST outperforms the other interpolation algorithms in our experiment. Since we capture spatial and temporal features of traffic flows, ST has more rules and techniques to interpolate missing values than KNN and SRSVD Base. Although there is high GPS data loss, ST can still keep a good performance. From the comparison, we find that there is significant difference between ST and other algorithms when GPS-LOSS-PROBABILITY = 0.6. KNN works well if enough GPS reports have been collected in the datacenter. However, the performance of KNN decreases sharply when more and more GPS data lost. Because even we have chosen the best K neighbors for target route, we still do not have another rule to interpolate traffic flows for target road segment.

SRSVD Base does not work well, since it is to some extent a pure technique of matrix calculation. Therefore, it is not able to be adapted to complicated traffic condition.

3.4. Simulation

The basic experimental parameters are displayed in **Table 1**. In followed experiments, we assume that 30% vehicles do not install GPS devices. **Figure 6** shows evolutions of R_t on different road segments. At first, we select three different road segments randomly in center of Shanghai. Secondly, we collect the GPS reports on these roads from 11:00 to 12:00 on January 7 2007. At last, we recon-

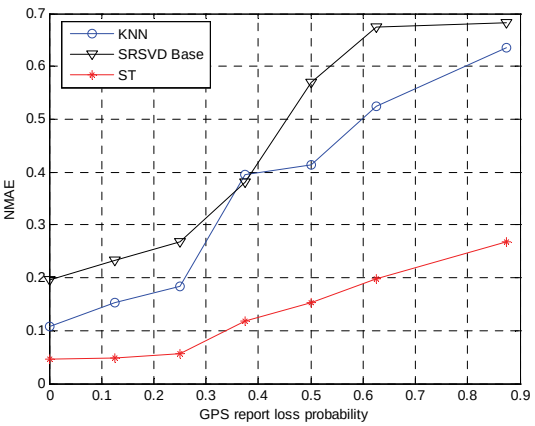


Figure 5. Performance for GPS report loss.

Table 1: Experimental parameters.

Target Area	$3.5km^2$
Time duration	2hours (11:00-13:00)
Time granularity	5min
Number of road segments	146
Weather	Sunny

structed OD matrix using ST to calculate R_t. In this experiment, we find that traffic lights have little effect on Traffic-Rate and Traffic-Rate changes smoothly.

In order to demonstrate performance of equations of λ: We select 10 road segments randomly and monitor the traffic volume from 9:00-22:00 on January 10 2007. Then we calculate R_t and λ every 5 minutes. Meanwhile, we use mean speed of these vehicles to detect current traffic status. Finally, we compare our result with pre-defined traffic state.

Figure 7 shows that the accuracy of our solution and comparison with traditional policy. Apparently, from this experiment, speed is not a good metric to detect traffic condition. Since it is often influenced by many factors, and there are always some vehicles have "strange" driving behaviors because of habits of drivers. On the other side, we can clear see that even 90% of accuracy can be achieved at sometimes. There are also two least values which are about 0.712 at 12:00 and 0.734 at 17:00. Since they are two important time points when people come off duty, and traffic condition at that time is difficult to be captured. But in general, the performance of threshold-based Traffic-Rate is good and adapted to the real environment.

In **Figure 8**, the axis of abscissas represents proportion of vehicles that installed GPS devices. This experiment proves that the more vehicles send GPS reports, the better performance of our algorithm is.

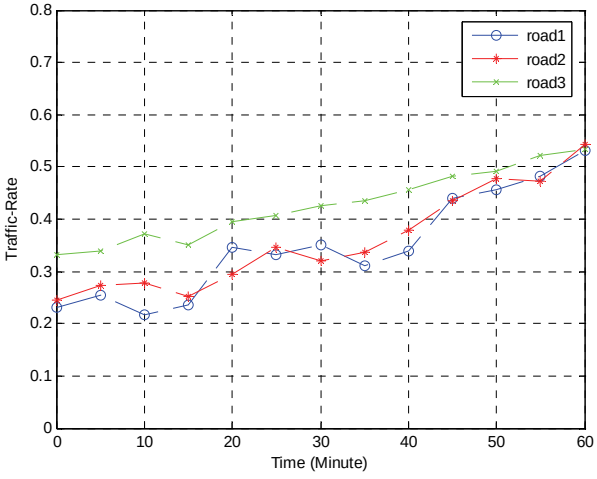


Figure 6. Traffic-Rate changes during an hour on 3 roads.

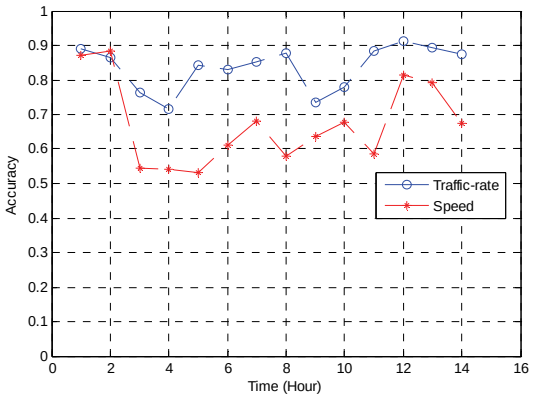


Figure 7. Accuracy of threshold-based Traffic-Rate and mean speed of vehicles.

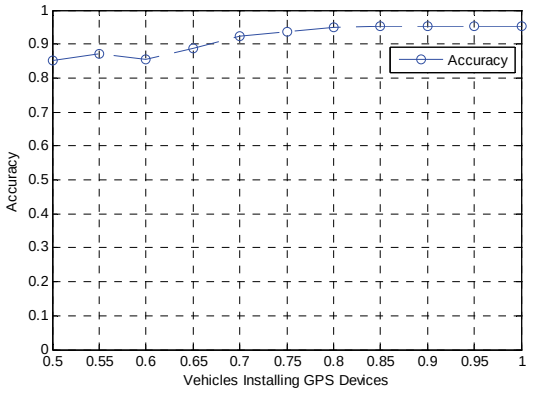


Figure 8. Accuracy of threshold-based Traffic-Rate.

4. Conclusions

In this paper, we have presented a practical solution to detect traffic condition which is based on OD matrix in an urban city by using a cost-effective system of vehicle GPS sensors. Our algorithm consists of spatial and temporal components which are based on neighbors of roads and stability change of traffic volumes during a time. Meanwhile, we use traffic-rate to capture traffic status rather that speed. As a result, our solution overcomes many limitations in existing approaches and yields a good performance.

5. References

[1] D. Bhattacharjee, K. C. Sinha and J. V. Krogmeier, "Modeling the Effects of Traveler Information on Freeway Origin-Destination Demand Prediction," *Transportation Research*, Vol. 6, No. 9, 2001, pp. 381-398.

[2] G. L. Chang and J. F. Wu, "Recursive Estimation of Time-Varying Origin-Destination Flows from Traffic Counts in Freeway Corridors," *Transportation Research Part B*, Vol. 28, No. 2, 1994, pp. 141-160.

[3] A. Medina, N. Taft, K. Salamatian, S. Bhattacharyya and C. Diet, "Traffic Matrix Estimation: Existing Techniques and New Directions," *Proceedings of ACM SIGCOMM*, 2002.

[4] Y. Zhang, M. Roughan, W. Willinger and L. L. Qiu, "Spatio-Temporal Compressive Sensing and Internet Traffic Matrices," *Proceedings of ACM SIGCOMM*, 2009.

[5] Y. Cho, "Estimating Velocity Fields on a Freeway from Low-Resolution Videos," *IEEE Transactions on Intelligent Transportation Systems*, Vol.7, No. 4, 2007, pp. 463-469.

[6] W. Lin and C. Daganzo, "A Simple Detection Scheme for Delay-Inducing Freeway Incidents," *Transportation Research*, Vol. 31A of Part A, 1997, pp. 141-155.

[7] B. Coifman, "Identifying the Onset of Congestion Rapidly with Existing Traffic Detector," *Transportation Research*, Vol. 37 of Part A, 2003, pp. 277-291.

[8] J. Yoon, B. Noble and M. Liu, "Surface Street Traffic Estimation," *Proceedings of ACM Mobicom*, 2007

[9] H. Z. Zhu, Y. M. Zhu, M. L., Li and L. M. Ni, "SEER: Metropolitan-Scale Traffic Perception Based on Lossy Sensory Data," *Proceedings of ACM INFOCOM*, 2009.

[10] M. McNally, J. Marca, C. Rindty and A. Koos, "Tracer: In-Vehicle, Gps-Based, Wireless Technology for Traffic Surveillance and Management," Technical Report UCB-ITS-PRR-2003-23, California Partners for Advanced Transit and Highways (PATH), July 2003.

[11] J. Ygnace, C. Drane, Y. Yim and R. Lacvivier, "Travel Time Estimation on the San Francisco Bay Area Network Using Cellular Phones as Probes," Technical Report UCB-ITS-PWB-2000-18, California Partners for Advanced Transit and Highways (PATH), September 2000.

Wireless Sensor Network, **2010, 2, 675-682**
doi:10.4236/wsn.2010.29081 Published Online September 2010 (http://www.SciRP.org/journal/wsn)

Approximate Continuous Aggregation via Time Window Based Compression and Sampling in WSNs

Lei Yu, Jianzhong Li, Siyao Cheng
Department of Computer Science and Technology, Harbin Institute of Technology, Harbin, China
E-mail: {*yulei*2008,*lijzh*}*@hit.edu.cn, csyhit@*126*.com*
Received July 4, 2010; *revised August* 7, 2010; *accepted September* 10, 2010

Abstract

In many applications continuous aggregation of sensed data is usually required. The existing aggregation schemes usually compute every aggregation result in a continuous aggregation either by a complete aggregation procedure or by partial data update at each epoch. To further reduce the energy cost, we propose a sampling-based approach with time window based linear regression for approximate continuous aggregation. We analyze the approximation error of the aggregation results and discuss the determinations of parameters in our approach. Simulation results verify the effectiveness of our approach.

Keywords: Approximate Aggregation, Continuous Aggregation, Sampling, Sensor Network

1. Introduction

Wireless sensor networks (WSNs) offer a powerful and efficient approach for monitoring and collecting information in a physical environment. To extract the summary information about the monitored environment, the aggregations of sensed data, such as sum and average, are common interesting queries for users. Therefore, a lot of algorithms and protocols for aggregate query processing in WSNs are proposed [1-8].

The existing works addressed two types of aggregate queries which include exact and approximate aggregate queries. The exact aggregate query requires all the sensed data to be involved in aggregation computation to obtain the exact aggregation results [1,2]. However, the exact aggregate query processing often incurs great energy consumption and is also very sensitive to the packet loss and node failure during the data aggregation. Considering the approximate aggregation results would be enough to reflect the information of the environment, approximate aggregate query processing is addressed to save energy and achieve robustness against the failure of the links and nodes [3-8]. In the research of the approximate aggregate query processing in WSNs, sampling is widely used as a powerful and energy-efficient technique to obtain the statistical information of the environment. A number of sampling based schemes have been proposed for approximate query processing in WSNs [8-10].

In the applications of WSNs such as monitoring air pollution and water quality, the users are often interested in understanding how the environment changes over time and observing data trend in a time window. In such cases, continuous aggregation of sensed data is usually required. In a continuous aggregation, the query aggregation period is divided into epochs and one aggregate answer is provided at each epoch. The existing aggregation schemes usually compute every aggregation result in a continuous aggregation either by a complete aggregation procedure [1,2-4,7] or by partial data update [8] at each epoch. However, the users, who are interested in the time-evolving characteristic of aggregation results, are more concerned about the data trend rather than each individual accurate aggregation result. On the other hand, the communication cost of the existing schemes could be substantial, especially for continuous query with a short epoch and a long period. Motivated by such circumstances, we propose a sampling-based approach with time window based compression for approximate continuous aggregation.

Our approach leverages the batch-based design to compute a period of aggregation results at one time. While giving a series of good approximate aggregation results to provide accurate data trend information, it achieves greater energy-savings than the existing approaches by avoiding individual computation cost of every epoch. In our approach, the combination of data compression and sampling techniques is exploited. A small portion of sensor nodes transmit to the base station

(BS) a compact description of their sensor readings during a time window. The BS computes approximation aggregation results of every epoch in this time window. In this paper, linear regression modeling is adopted by sensor nodes to compress their sensor data in a time window. We analyze the approximation error of the aggregation results and discuss the determinations of parameters in our approach.

The rest of the paper is organized as follows. We present our approach and approximation error analysis in Section 2. We discuss the determination of parameters in our approach in Section 3. Simulation results are presented in Section 4. Finally, we conclude this paper in Section 5.

2 Approximate Continuous Aggregations

2.1 System Model and Time Window Based Framework

We assume a multi-hop sensor network with *N* number of sensor nodes. The BS knows *N*. All the sensor nodes and the base station are loosely time synchronized. Each node has the same communication radius R_c. We assume a continuous querying environment for sensor networks. For a continuous aggregation query, the base station initially disseminates a query into the network, consisting of the epoch duration, the lifetime of the query evaluation and a sampling ratio ϱ.

During the period of a continuous aggregation query, aggregation computation is conducted at time intervals. Each time interval consists of l number of successive epochs. The BS computes the aggregation result of every epoch in a time interval at one time. Such a time interval is referred to as time window and represented by $[t+1, t+l]$. l is the time window size. Let $Ag_{t+1}, \ldots, Ag_{t+l}$ denote the aggregation results from l successive epochs $t+1, \ldots, t+l$.

In the network, the aggregation computation involves sampling sensor nodes that participate in answering the aggregation query, and collecting a compressed representation of sensor readings within a time window from each sampled node.

After receiving the query from the BS, each sensor node *u* generates a random number rn_u in the range of [0, 1). If $rn_u \le \varrho$, *u* is sampled for the aggregation query, otherwise *u* is not sampled. Let $S = \{s_i \mid 1 \le i \le m\}$ (m is the sample size) be the set of sampled nodes. At the end of a time window $[t+1, t+l]$, each node $s_i \in S$ generates a compressed representation M_i of its sensing readings $\{r_{i,t+1}, r_{i,t+2} \ldots, r_{i,t+l}\}$ that contributes to the aggregation in the time window $[t+1, t+l]$. The generation of M_i depends on the specific data compression method we adopted. After that, s_i transmits M_i to the BS. The BS reconstructs the sensor readings of every sampled node s_i by M_i, denoted by $\{\hat{r}_{i,t+1}, \hat{r}_{i,t+2}, \ldots, \hat{r}_{i,t+l}\}$, and computes an approximation answer $\widehat{Ag}_k (t+1 \le k \le t+l)$ for a specific aggregation query.

Definition 1. ((ε, δ)-*approximation aggregation*): Let A_k be a true aggregation result of epoch k, $\widehat{Ag}_k$ is called as (ε, δ)-approximation of A_k, if $\Pr(|\widehat{Ag}_k - A_k| \ge \varepsilon) \le \delta$.

2.2. Modeling Sensor Data with Error Constraint

In our framework, a sample is not a single sensor reading but a compressed representation of the sensor readings, which enables a sensor node to transmit its sensing readings in a time window with less communication cost. It can be built by either lossy or lossless compression methods.

Considering the inherent redundancy of sensor data and the fundamental limit of lossless compression in information theory, we use a data modeling approach, linear regression, to achieve a lossy compression of sensor readings. Linear regression has been widely used to characterize data in sensor networks and answer aggregation queries [11-13]. On this basis, lossless compression methods always can be used for any possible further size reduction. Nevertheless, we note that our framework does not depend on any particular compression method. However, data compression with linear regression modeling would introduce errors in the reconstructed data. Therefore, we put error constraints on the modeling process in our approach. If sampled nodes find that the variance of error incurred by modeling exceeds some threshold σ_T^2, referred to as *error constraint*, they choose to transmit their original data. Otherwise, model parameters including error variance are transmitted.

2.2.1. Linear Regression Model

Regarding the sensor readings $r_{t+1}, \ldots, r_{t+l}$ of a node in each time window $[t+1, t+l]$ as a function of the sequence number from 1 to l, a linear regression model [14] for these sensor readings is built in the following form

$$\mathbf{R} = \mathbf{X}\Theta + \xi \tag{1}$$

where $\mathbf{R} = (r_{t+1}, \ldots, r_{t+l})^T$, $\Theta = (\theta_0, \theta_1, \ldots, \theta_p)^T$,

$$\mathbf{X} = \begin{pmatrix} h_0(1) & h_1(1) & \ldots & h_p(1) \\ h_0(2) & h_1(2) & \ldots & h_p(2) \\ \vdots & \vdots & \vdots & \vdots \\ h_0(l) & h_1(l) & \ldots & h_p(l) \end{pmatrix} = \begin{pmatrix} 1 & 1 & \ldots & 1 \\ 1 & 2 & \ldots & 2^p \\ \vdots & \vdots & \vdots & \vdots \\ 1 & l & \ldots & l^p \end{pmatrix},$$

$\xi = (\varepsilon_{t+1}, \ldots, \varepsilon_{t+l})^T$.

In the model, $\{h_i(x) \mid h_i(x) = x^i, 0 \le i \le p\}$ are the set of basis functions, $\theta_0, \theta_1, \ldots, \theta_p$ are regression coefficients, and ξ is a random error vector. Besides, the time window size l is larger than $p+1$. According to Gauss-Markov conditions [14], we also have $E(\varepsilon_i) = 0$, $Var(\varepsilon_i) = \sigma^2$ and $Cov(\varepsilon_i, \varepsilon_j) = 0$ where $i \neq j$, $i, j \in \{t+1, \ldots, t+l\}$.

By the least square estimate, the estimation of regression coefficients, denoted by $\widehat{\Theta} = (\hat{\theta}_0, \hat{\theta}_1, \ldots, \hat{\theta}_p)$, can be computed by solving the following matrix equation, using, for example, Gaussian elimination:

$$\mathbf{A}\widehat{\Theta} = \mathbf{b} \tag{2}$$

where $\mathbf{A} = \mathbf{X}^T\mathbf{X}$, $\mathbf{b} = \mathbf{X}^T\mathbf{R}$.

Once determining l and p, we can see that the matrices $\mathbf{X}$ and $\mathbf{A}$ do not change with $\mathbf{R}$, so they just need to be computed only once for an aggregation query.

2.2.2. Error Variance and Data Reconstruction

Besides computing regression coefficients $\widehat{\Theta}$, each sampled node also needs to estimate the variance of the errors, denoted by σ^2, to decide whether to transmit original data or regression coefficients.

Under Gauss-Markov conditions [14], an unbiased estimator of error variance σ^2 can be computed by

$$\widehat{\sigma}^2 = \frac{(\mathbf{R} - \mathbf{X}\widehat{\Theta})^T(\mathbf{R} - \mathbf{X}\widehat{\Theta})}{l - p - 1} \tag{3}$$

Given an error constraint σ_T^2, if $\widehat{\sigma}^2 \le \sigma_T^2$, the node transmits $p+1$ number of regression coefficients $\widehat{\Theta} = (\hat{\theta}_0, \hat{\theta}_1, \ldots, \hat{\theta}_p)$ and $\widehat{\sigma}^2$ to the base station. Otherwise, it transmits l number of original sensor readings.

By the regression coefficients of $\widehat{\Theta}$ received from a sampled node, the BS can reconstruct its sensor readings $\widehat{\mathbf{R}} = (\hat{r}_{t+1}, \ldots, \hat{r}_{t+l})$ in the time window by

$$\widehat{\mathbf{R}} = \mathbf{X}\widehat{\Theta} \tag{4}$$

where $\mathbf{X}$ can be pre-computed by the BS with l and p.

In the rest of this paper, we regard both the original readings and the regression coefficients as model parameters and do not distinguish them. A sample transmitted by a sampled node s_i is denoted by $M_i = (\widehat{\Theta}_i, \widehat{\sigma}_i^2)$. When $M_i = (\widehat{\Theta}_i, 0)$, M_i represents the original sensor readings.

2.3. Approximate Aggregation

2.3.1. Aggregation Estimation

At the end of each time window, the BS waits for the arrivals of all samples for some time t_w. The waiting time t_w should be larger than the maximum time needed for the message delivery from the samples node to the BS.

After reconstructing sensor readings $\{\hat{r}_{i,k} \mid 1 \le i \le m\}$ of sampled nodes $\{s_i \mid 1 \le i \le m\}$ at epoch k in a time window $[t+1, t+l]$ by Formula (4), the approximation aggregation result $\widehat{Ag}_k$ of epoch k $(t+1 \le k \le t+l)$ can be obtained by

$$\widehat{Ag}_k = F(\hat{r}_{1,k}, \hat{r}_{2,k}, \ldots, \hat{r}_{m,k}) \tag{5}$$

where F is the estimator function of aggregation results. Now we specifically discuss how to estimate the results of aggregation queries including Average and Sum respectively.

Average Average aggregation is estimated by

$$\hat{A}_k^a = \frac{1}{m}\sum_{i=1}^{m} \hat{r}_{i,k} \tag{6}$$

Sum Sum aggregation is estimated by

$$\hat{A}_k^a = N\hat{A}_k^a = \frac{N}{m}\sum_{i=1}^{m} \hat{r}_{i,k} \tag{7}$$

2.3.2. Approximation Error Analysis

Let $\hat{\varepsilon}_{i,k} = r_{i,k} - \hat{r}_{i,k}$. If the estimator function F is a linear function, Formula (5) can be rewrote as

$$\begin{aligned} \widehat{Ag}_k &= F(r_{1,k} - \hat{\varepsilon}_{1,k}, r_{2,k} - \hat{\varepsilon}_{2,k}, \ldots, r_{m,k} - \hat{\varepsilon}_{m,k}) \\ &= F(r_{1,k}, r_{2,k}, \ldots, r_{m,k}) - F(\hat{\varepsilon}_{1,k}, \hat{\varepsilon}_{2,k}, \ldots, \hat{\varepsilon}_{m,k}) \end{aligned} \tag{8}$$

where $r_{i,k}$ is the original data of epoch k and $\hat{\varepsilon}_{i,k}$ is the residual in the linear regression model (1) of node s_i.

Then, the approximation error of $\widehat{Ag}_k$ to the exact aggregation result A_k of epoch k is

$$\begin{aligned} &|\widehat{Ag}_k - A_k| \\ &= |F(r_{1,k}, r_{2,k}, \ldots, r_{m,k}) - A_k - F(\hat{\varepsilon}_{1,k}, \hat{\varepsilon}_{2,k}, \ldots, \hat{\varepsilon}_{m,k})| \\ &\le \underbrace{|F(r_{1,k}, r_{2,k}, \ldots, r_{m,k}) - A_k|}_{\text{sampling estimation error}} + \underbrace{|F(\hat{\varepsilon}_{1,k}, \hat{\varepsilon}_{2,k}, \ldots, \hat{\varepsilon}_{m,k})|}_{\text{modeling estimation error}} \end{aligned} \tag{9}$$

where $|F(r_{1,k}, \ldots, r_{m,k}) - A_k|$ is the estimation error with original data samples, referred to as *sampling estimation error*, and $|F(\hat{\varepsilon}_{1,k}, \ldots, \hat{\varepsilon}_{m,k})|$ is referred to as *modeling estimation error*. The above result indicates the approximation error consists of two types of errors including sampling estimation error and modeling estimation error. Because these two errors separately rely on different factors such as the sample size or the number of regression coefficients, we regard them as two independent random variables.

Now we specifically analyze the approximate error of Average and Sum. Let A_k^a and A_k^s be the exact average and sum result of epoch k $(t+1 \le k \le t+l)$ respectively, *i.e.*, $A_k^a = \frac{1}{N}\sum_{i=1}^{N} r_{i,k}$ and $A_k^s = \sum_{i=1}^{N} r_{i,k}$. By Formula (8), we have

$$\hat{A}_k^a = \frac{1}{m}\sum_{i=1}^{m} r_{i,k} - \frac{1}{m}\sum_{i=1}^{m} \hat{\varepsilon}_{i,k}$$

As we can see, $\hat{A}_k^a$ is a linear combination of two random variables R_k and Z_k, $R_k = \frac{1}{m}\sum_{i=1}^{m} r_{i,k}$ and $Z_k = \frac{1}{m}\sum_{i=1}^{m} \hat{\varepsilon}_{i,k}$.

According to the linear regression theory, under Gauss-Markov conditions, the residual $\hat{\varepsilon}_{i,k}$ follows a normal distribution $N(0, \sigma_i^2(1-p_{k'k'}))$ where σ_i^2 is the error variance in the linear model at node s_i, $k' = k - t$ and $p_{k'k'}$ is the k'-th element on the principal diagonal of matrix $P_X = X(X^T X)^{-1} X^T$. Considering that $\hat{\sigma}_i^2$ in M_i is an unbiased estimator of σ_i^2, we have

$$Z_k \sim N(0, \frac{1-p_{k'k'}}{m^2}\sum_{i=1}^{m} \hat{\sigma}_i^2) \quad (10)$$

Since R_k is the mean of original data samples, according to the general results in the sampling theory [15], we have the following results:

$$E(R_k) = A_k^a$$

$$Var(R_k) = \frac{S_k^2}{m}(1-\frac{m}{N})$$

$$S_k^2 = \frac{1}{N-1}\sum_{i=1}^{N}(r_{i,k} - A_k^a)^2$$

Confidence Interval:

$$\Pr\left(R_k - \varphi_{\frac{\alpha}{2}} s_k \sqrt{\frac{1-f}{m}} \le A_k^a \le R_k + \varphi_{\frac{\alpha}{2}} s_k \sqrt{\frac{1-f}{m}}\right) = 1-\alpha, \quad (11)$$

where $f = m/N$, $\varphi_{\frac{\alpha}{2}}$ is the upper $\alpha/2$ point on the standard normal distribution, $s_k^2 = \frac{1}{m-1}\sum_{i=1}^{m}(r_{i,k} - R_k)^2$ is the (unbiased) sample variance and is an unbiased estimator of the population MSE(Mean Square Error) S_k^2.

By the above discussions, we have the following results:

Lemma 1. *Under Gauss-Markov conditions,*

$$\mathrm{E(r}\, E(r_{i,k}\hat{\varepsilon}_{j,k}) = \begin{cases} 0, & i \ne j \\ \sigma_i^2(1-p_{k'k'}) & i = j \end{cases}$$

Proof. If $i \ne j$, since the samples $r_{i,k}$ and $r_{j,k}$ are assumed to be independent random variables in the sampling theory, $r_{i,k}$ and $\hat{\varepsilon}_{j,k}$ are independent and we have

$$E(r_{i,k}\hat{\varepsilon}_{j,k}) = E(r_{i,k})E(\hat{\varepsilon}_{j,k}) = 0.$$

If $i = j$, according to the linear regression theory, we know $\hat{\varepsilon}_{i,k}$ and $\hat{\theta}_{i,x}$ ($0 \le x \le p$) are independent, thus $E(\hat{\theta}_{i,x}\hat{\varepsilon}_{i,k}) = E(\hat{\theta}_{i,x})E(\hat{\varepsilon}_{i,k}) = 0$. Then, we have

$$E(r_{i,k}\hat{\varepsilon}_{i,k}) = E[(\hat{\theta}_{i,0} + \hat{\theta}_{i,1}k' + \hat{\theta}_{i,2}k'^2 + \ldots + \hat{\theta}_{i,p}k'^p + \hat{\varepsilon}_{i,k})\hat{\varepsilon}_{i,k}]$$
$$= E(\hat{\varepsilon}_{i,k}^2) = \sigma_i^2(1-p_{k'k'})$$

Theorem 1. *Let* $\hat{s}_k^2 = \frac{1}{m-1}\sum_{i=1}^{m}(\hat{r}_{i,k} - \hat{A}_k^a)^2$ *and* $k' = k - t$. *Then*

$$E(\hat{s}_k^2) = S_k^2 - \frac{1-p_{k'k'}}{N}\sum_{i=1}^{N}\sigma_i^2. \quad (12)$$

Proof. It can be easily shown that

$$\hat{s}_k^2 = \frac{1}{m(m-1)}\sum_{i=1}^{m}\sum_{\substack{j=1 \\ i<j}}^{m}(\hat{r}_{i,k} - \hat{r}_{j,k})^2$$
$$= \frac{1}{2m(m-1)}\sum_{i=1}^{m}\sum_{\substack{j=1 \\ i\ne j}}^{m}(\hat{r}_{i,k} - \hat{r}_{j,k})^2$$

$$S_k^2 = \frac{1}{N(N-1)}\sum_{i=1}^{N}\sum_{\substack{j=1 \\ i<j}}^{N}(r_{i,k} - r_{j,k})^2$$
$$= \frac{1}{2N(N-1)}\sum_{i=1}^{N}\sum_{\substack{j=1 \\ i\ne j}}^{N}(r_{i,k} - r_{j,k})^2$$

For each pair $(\hat{r}_{i,k}, \hat{r}_{j,k})$ ($i \ne j$, $1 \le i, j \le N$), the probability that they are both being reconstructed due to the corresponding nodes (i, j) being sampled, is $m(m-1)/(N(N-1))$. Then, with Lemma 1, we have

$$E(\hat{s}_k^2) = \frac{1}{2m(m-1)}E(\sum_{i=1}^{m}\sum_{\substack{j=1 \\ i\ne j}}^{m}(\hat{r}_{i,k} - \hat{r}_{j,k})^2)$$
$$= \frac{1}{2m(m-1)}\sum_{i=1}^{N}\sum_{\substack{j=1 \\ i\ne j}}^{N}E((\hat{r}_{i,k} - \hat{r}_{j,k})^2)\frac{m(m-1)}{N(N-1)}$$
$$= \frac{1}{2N(N-1)}\sum_{i=1}^{N}\sum_{\substack{j=1 \\ i\ne j}}^{N}E((\hat{r}_{i,k} - \hat{r}_{j,k})^2)$$
$$= \frac{1}{2N(N-1)}\sum_{i=1}^{N}\sum_{\substack{j=1 \\ i\ne j}}^{N}((r_{i,k} - r_{j,k})^2 - E(\hat{\varepsilon}_{i,k}^2) - E(\hat{\varepsilon}_{j,k}^2))$$
$$= S_k^2 - \frac{1}{N}\sum_{i=1}^{N}E(\hat{\varepsilon}_{i,k}^2) = S_k^2 - \frac{1-p_{k'k'}}{N}\sum_{i=1}^{N}\sigma_i^2$$

By replacing S_k^2 by s_k^2, $E(\hat{s}_k^2)$ by $\hat{s}_k^2$, and σ_i^2 by $\hat{\sigma}_i^2$ in Formula (13), we can estimate s_k^2 by $\hat{s}_k^2 + \frac{1-p_{k'k'}}{N}\sum_{i=1}^{N}\hat{\sigma}_i^2$. However, $\sum_{i=1}^{N}\hat{\sigma}_i^2$ can not be obtained since sampling all nodes is prohibitive in our approach. Thus, we use an upper bound of s_k^2, denoted by $(s_k^*)^2$, and estimate it by $\hat{s}_k^2 + (1-p_{k'k'})\sigma_T^2$ due to $\hat{\sigma}_i^2 \le \sigma_T^2$.

Theorem 2.

$$\Pr\left(|\hat{A}_k^a - A_k^a| \le \varphi_{\frac{\alpha_r}{2}} s_k^* \sqrt{\frac{1-f}{m}} + \frac{1}{m}\varphi_{\frac{\alpha_z}{2}}\sqrt{(1-p_{k'k'})\sum_{i=1}^{m}\hat{\sigma}_i^2}\right) \quad (13)$$
$$\ge (1-\alpha_r)(1-\alpha_z)$$

where $f = \frac{m}{N}$, $s_k^* = \sqrt{\hat{s}_k^2 + (1-p_{k'k'})\sigma_T^2}$, $\varphi_{\frac{\alpha_r}{2}}$ *and* $\varphi_{\frac{\alpha_z}{2}}$ *are respectively the upper* $\alpha_r/2$, $\alpha_z/2$ *point on the standard normal distribution.*

Proof. Define the events A, B and C respectively as

$$A: \ |\hat{A}_k^a - A_k^a| \le \varphi_{\frac{\alpha_r}{2}} s_k^* \sqrt{\frac{1-f}{m}} + \frac{1}{m}\varphi_{\frac{\alpha_z}{2}} \sqrt{\sum_{i=1}^{m} \hat{\sigma}_i^2}$$

$$B: \ |R_k - A_k^a| \le \varphi_{\frac{\alpha_r}{2}} s_k^* \sqrt{\frac{1-f}{m}}$$

$$C: \ |Z_k| \le \frac{1}{m}\varphi_{\frac{\alpha_z}{2}} \sqrt{(1-p_{k'k'})\sum_{i=1}^{m} \hat{\sigma}_i^2}$$

because $s_k^{*2} \ge s_k^2$, by Formula (12) we have

$$\Pr(B) \ge \Pr(|R_k - A_k^a| \le \varphi_{\frac{\alpha_r}{2}} s_k \sqrt{\tfrac{1-f}{m}}) = 1-\alpha_r$$

since $Z_k \sim N(0, \frac{1-p_{k'k'}}{m^2}\sum_{i=1}^{m}\hat{\sigma}_i^2)$,

$$\Pr\left(|Z_k| \le \frac{1}{m}\varphi_{\frac{\alpha_z}{2}} \sqrt{(1-p_{k'k'})\sum_{i=1}^{m} \hat{\sigma}_i^2}\right) = 1-\alpha_z$$

By Formula (9), we have $|\hat{A}_k^a - A_k^a| \le |R_k - A_k^a| + |Z_k|$. When inequalities B and C are satisfied, A must hold. Because sampling and modeling errors are independent random variables, so B and C are independent events. Then, we have

$$\Pr(A) \ge \Pr(BC) = \Pr(B)\Pr(C) \ge (1-\alpha_r)(1-\alpha_z)$$

Let $\varepsilon_k = \varphi_{\frac{\alpha_r}{2}} s_k^* \sqrt{\frac{1-f}{m}} + \frac{1}{m}\varphi_{\frac{\alpha_z}{2}} \sqrt{(1-p_{k'k'})\sum_{i=1}^{m}\hat{\sigma}_i^2}$.

Since $\hat{A}_k^s = N\hat{A}_k^a = NR_k - NZ_k$, we can easily derive the following results from the above analysis of average:

$$\Pr\left(|\hat{A}_k^s - A_k^s| \le N\varepsilon_k\right) \ge (1-\alpha_r)(1-\alpha_z) \qquad (14)$$

here Formulas (13) and (14) give the approximation error ε_k ($N\varepsilon_k$) of Average (Sum) aggregation with the probability guarantee $(1-\alpha_r)(1-\alpha_z)$.

3. Parameter Determination

From Formulas (13) and (14) we can see that with given the probability guarantee, *i.e.*, α_r and α_z, the approximation error depends on the error constraint σ_T^2 and the sample size m. In this section we discuss the selection of their values with the desired error bound for ε_k by users, denoted by ε_T.

3.1. Error Constraint σ_T^2

As shown in Formula (3), $\hat{\sigma}_i$ indicates the average error for the data reconstructed in a time window. Thus, σ_T specifies the maximum degree of the average error that the user can tolerate for the reconstructed data. A larger σ_T would allow larger errors in the reconstructed data and may enlarge the approximation error. On the other hand, a larger σ_T gives the sampled nodes more chances to transmit their model parameters instead of their original data and further reduce the communication cost. Thus, the trade-off exists between communication cost and approximation error.

Here we provide one possible solution to determine σ_T^2. During the first time window of aggregation, all sampled nodes transmit their original data to the BS. The BS fits the specified model to these data and computes the modeling errors $\{\hat{\sigma}_i^2 \mid 1 \le i \le m\}$ for all sampled nodes. A histogram is computed to count the number of error values falling into each bin, which reflects the quality of data modeling for the sensor network. According to this frequency distribution, the user can select a value of σ_T^2 as large as possible while ensuring an acceptable approximation error. Finally, the BS broadcasts σ_T^2 to the sensor network and each sensor node works on the new error variance constraint. This procedure could be conducted reactively when substantial sampled nodes start to continuously transmit their original data, which indicates the changes of the nature of data in the sensor network.

In our experiments on real data set, we show linear regression well characterizes the sensor data and incur few original data transmissions even with a small error variance constrain.

3.2. Sampling Ratio ϱ

From Formulas (13) and (14), a larger sample size m enables a smaller approximation error.

It is easily shown that we can relax ε_k to

$$\varepsilon_k = \varphi_{\frac{\alpha_r}{2}} s_k^* \sqrt{\tfrac{1-f}{m}} + \tfrac{1}{\sqrt{m}}\varphi_{\frac{\alpha_z}{2}}\sigma_T$$

without changing the inequality relationship with the probability guarantee $(1-\alpha_r)(1-\alpha_z)$ in Formulas (13) and (14). We consider the least sample size to satisfy $\varepsilon_k \le \varepsilon_T$ for any k in $[t+1, t+l]$. With an approximation of $f = m/N \to 0$ (for relative small sample size and large population), we have

$$\tfrac{1}{\sqrt{m}}\varphi_{\frac{\alpha_r}{2}} s_k^* + \tfrac{1}{\sqrt{m}}\varphi_{\frac{\alpha_z}{2}}\sigma_T \le \varepsilon_T$$

which should hold for any k in $[t+1, t+l]$. Then, we can obtain the least sample size m_k required by epoch k in the time window $[t+1, t+l]$ to ensure ε_k is less than a threshold ε_T

$$m_k = \left(\frac{\varphi_{\frac{\alpha_r}{2}} s_k^* + \varphi_{\frac{\alpha_z}{2}}\sigma_T}{\varepsilon_T}\right)^2, \quad t+1 \le k \le t+l \qquad (15)$$

For each epoch k in $[t+1, t+l]$, if the BS finds $m < m_k$, it can issue another sampling request to obtain $m_k - m$ samples. However, s_k^* cannot be obtained before sampling, we give the following estimation if the

upper bound r_{max} and lower bond r_{min} of sensor readings are known

$$s_k^* \approx \sqrt{\frac{1}{m-1}\sum_{i=1}^{m}(\frac{r_{\max}-r_{\min}}{2})^2} \quad (16)$$

We can obtain an estimation of the required sample size, denoted by m_r, for all epochs in the time window $[t+1, t+l]$ by inserting Formula (15) into Formula (16). The sampling ratio ϱ is set to be not less than m_r/N.

3.3. Time Window Size *l*

When all sampled nodes transmit their original data, the approximation error includes only the sampling estimation error and no modeling estimation error. Thus, the aggregation computation with original data needs a less sample size than with the compressed data by modeling to achieve the same approximation error. Let m_o be the sample size needed to obtain (ε_T, δ) -approximation aggregation by collecting the original data, then $\varphi_{\frac{\delta}{2}} s_k \sqrt{\frac{1-f_o}{m_o}} = \varepsilon_T$ where $f_o = \frac{m_o}{N}$. With the approximation of $f_o = m_o / N \to 0$,

$$m_o = (\varphi_{\frac{\delta}{2}} s_k)^2 / \varepsilon_T^2$$

on the other hand, we have

$$\delta = 1-(1-\alpha_r)(1-\alpha_z) \Rightarrow \delta > \alpha_r \Rightarrow \varphi_{\frac{\delta}{2}} < \varphi_{\frac{\alpha_r}{2}}$$

As above, we also have $\varphi_{\frac{\delta}{2}} < \varphi_{\frac{\alpha_z}{2}}$.

According to the above discussion on sampling ratio, we have

$$\frac{m_t}{m_o} = \frac{(\varphi_{\frac{\alpha_r}{2}} s_k^* + \varphi_{\frac{\alpha_z}{2}} \sigma_T)^2 / \varepsilon_T^2}{(\varphi_{\frac{\delta}{2}} s_k)^2 / \varepsilon_T^2} = \frac{(\varphi_{\frac{\alpha_r}{2}} s_k^* + \varphi_{\frac{\alpha_z}{2}} \sigma_T)^2}{(\varphi_{\frac{\delta}{2}} s_k)^2}$$

$$\geq \frac{(\varphi_{\frac{\delta}{2}} s_k^* + \varphi_{\frac{\delta}{2}} \sigma_T)^2}{(\varphi_{\frac{\delta}{2}} s_k)^2} \geq \frac{(s_k^* + \sigma_T)^2}{s_k^2} \quad (17)$$

$$\geq \frac{(s_k + \sigma_T)^2}{s_k^2} = (1+\frac{\sigma_T}{s_k})^2$$

Without data modeling compression, the aggregation requires $m_o(l+1)$ original data transmissions for a time window to achieve the approximation error ε_T. With the data modeling compression, our scheme requires $m(p+2)$ data transmissions to achieve the approximation error ε_T. To achieve energy savings, we should have $m_o(l+1)/m_t(p+2) > 1$, then

$$l+1 > (p+2)\frac{m_t}{m_o} \geq (p+2)(1+\frac{\sigma_T}{s_k})^2$$

In the case of $\sigma_T < S_k$, $1+\frac{\sigma_T}{s_k} < 2$, we could set $l+1 > 4(p+2)$.

4. Simulation Evaluation

To measure the performance of our secure aggregation scheme, we simulate a sensor network based on the data from a real world deployment with 54 sensor nodes (ID from 1-54) in the Intel Research lab, which includes a trace of sensor readings collected between February and April, 2004, node location and network connectivity information. The sensors collected time-stamped humidity, temperature and voltage values in 31 second intervals. We use the first 2000 epochs of the data set in the day 03/08 with the largest size among all days and assume a continuous aggregation query on the temperature attribute during this period. The periodic aggregation is conducted on it with a time window size $l = 40$, *i.e.*, 50 time windows. In the linear regression model, we let $p = 3$.

To show the performance of linear regression model for describing sensor data, we investigate the distribution of error variance and its impact on data transmission for all sensor nodes.

Figure 1 shows temperature readings (in degrees Celsius) of 52 sensor nodes in 2000 successive epochs, which are used for our simulation. **Figure 2** shows the error variance of linear regression model in every time window for all sensor nodes. For all time windows, all the sensor nodes have error variances less than 0.2. **Figure 3** shows under different choice of error constraint σ_T^2, the number of sensor nodes which has a larger modeling error variance than σ_T^2 in each time window. We can conclude most of sensor nodes at most of time windows are consistent with variance constraint. When $\sigma_T^2 > 0.07$, less than 10% of sensor nodes exceed σ_T^2; when $\sigma_T^2 > 0.1$, the number decreases to 2%.Our experiment indicates only a small portion of sampled node will transmit their original data.

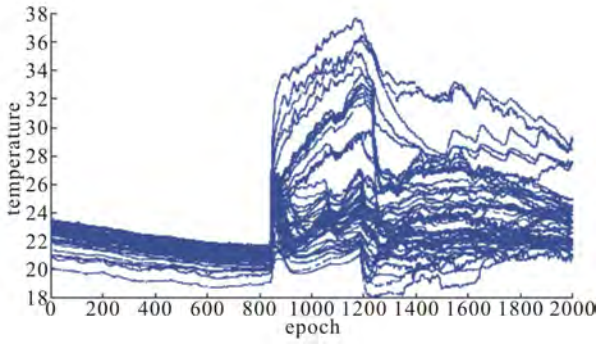


Figure 1. Temperature readings (in degrees Celsius) of 52 sensor nodes in 2000 successive epochs (excluding two nodes with incomplete data and one node with abnormal data).

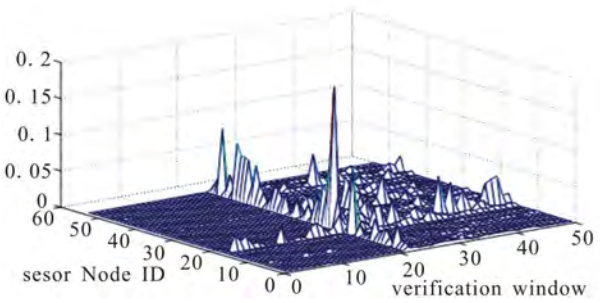


Figure 2. The error variances of linear regression model in all sensor nodes for each time window.

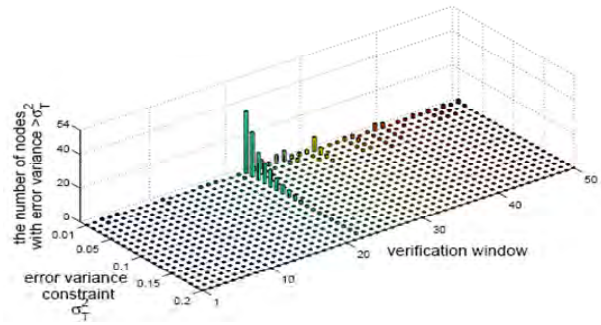


Figure 3. the number of sensor nodes with error variance $> \sigma_T^2$ in each time window.

Assuming ϱ = 0.4 and $\sigma_T^2 = 0.2$, **Figure 4** shows the difference between the average aggregation result estimated by sampling with time window based compression and the aggregation result estimated by sampling with original data transmission in every epoch. As we can see, the difference is in [–0.2, 0.25]. It indicates that our approach with data compression can obtain the estimation of average aggregations close to those obtained by the approach without data compression, even when the sample size is the same. It also indicates our approach achieves the energy efficiency while obtaining the approximate estimations, since in each time window only five numbers are sent from each sampled node.

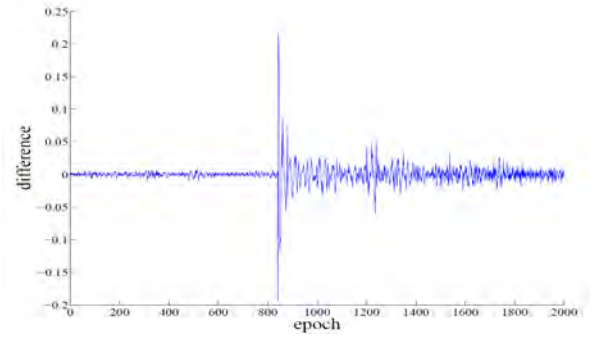


Figure 4. The difference between two average aggregation results respectively estimated by the approaches with and without data compression in every epoch.

5. Conclusions

In this paper we propose a sampling-based approach with time window based linear regression for approximate continuous aggregation. The approximation error of the aggregation results is analyzed. The determination of parameters in our approach is also discussed. By simulation results on real data set we verify the effectiveness of our approach.

6. References

[1] S. Madden, M. J. Franklin, J. M. Hellerstein and W. Hong, "The Design of an Acquisitional Query Processor for Sensor Networks," *Proceedings of International Conference on Management on Data*, California, 2003, pp. 491-502.

[2] S. Madden, M. J. Franklin, J. M. Hellerstein and W. Hong, "TAG: A Tiny Aggregation Service for Ad Hoc Sensor Networks," *Proceedings of the* 5*th Symposium on Operating Systems Design and Implementation*, New York, 2002, pp. 131-146.

[3] J. Considine, F. Li, G. Kollios and J. Byers, "Approximate Aggregation Techniques for Sensor Databases," *International Conference on Data Engineering*, Boston, 2004, pp. 449-460.

[4] S. Nath, P. B. Gibbons, S. Seshan and Z. R. Anderson, "Synopsis Diffusion for Robust Aggregation in Sensor Networks," *Sensys*, 2004, pp. 250-262.

[5] G. Cormode, M. N. Garofalakis, S. Muthukrishnan and R. Rastogi, "Holistic Aggregates in a Networked World: Distributed Tracking of Approximate Quantiles," *Proceedings of International Conference on Management on Data*, 2005, pp. 25-36.

[6] A. Deligiannakis, Y. Kotidis and N. Rossopoulos, "Processing Approximate Aggregation Queries in Wireless Senor Networks," *Information Systems*, Vol. 31, No. 8, 2006, pp. 770-792.

[7] A. Manjhi, S. Nath and P. B. Gibbons, "Tributaries and Deltas: Efficient and Robust Aggregation in Sensor Network Streams," *ACM SIGMOD*, ACM Press, 2005, pp. 287-298.

[8] S. Y. Cheng, J. Z. LI, Q. Q. Ren and L. Yu, "Bernoulli Sampling Based (epsilon, delta)-Approximate Aggregation in Larger-Scale Sensor Networks," *IEEE International Conference on Computer Communications*, California, 2010, pp. 1181-1189.

[9] S. Lin, B. Arai, D. Gunopulos and G. Das, "Region Sampling: Continuous Adaptive Sampling on Sensor Networks," *IEEE International Conference on Data Engineering*, Cancun, 2008, pp. 794-803.

[10] B. Bash, J. Byers and J. Considine, "Approximately Uni-

form Random Sampling in Sensor Networks," *Proceedings of* 1*st Workshop on Data Management in Sensor Networks*, August, 2004.

[11] C. Guestrin, P. Bodik, R. Thibaux, M. Paskin and S. Madden, "Distributed Regression: An Efficient Framework for Modeling Sensor Network Data," *ACM/IEEE IPSN*, 2004, pp. 1-10.

[12] W. Xue, Q. Luo, L. Chen and Y. Liu, "Contour Map Matching for Event Detection in Sensor Networks," *SIGMOD*, New York, 2006, pp. 145-156.

[13] H. Gupta, V. Navda, S. R. Das and V. Chowdhary, "Efficient Gathering of Correlated Data in Sensor Networks," *MobiHoc*, New York, 2005, pp. 402-413.

[14] A. Sen and M. Srivastava, "Regression Analysis: Theory, Methods, and Applications," Springer-Verlag, New York, 1990.

[15] Z. Govindarajulu, "Elements of Sampling Theory and Methods," Prentice Hall, New Jersey, 1999.

Wireless Sensor Network, **2010, 2, 683-688**
doi:10.4236/wsn.2010.29082 Published Online September 2010 (http://www.SciRP.org/journal/wsn)

Weak Greedy Routing over Graph Embedding for Wireless Sensor Networks

Zhigang Li, Nong Xiao
Computer School, National University of Defense Technology, Changsha, China
E-mail: {*lzg, nongxiao*}*@nudt.edu.cn*
Received June 11, 2010; *revised July* 22, 2010; *accepted August* 31, 2010

Abstract

In this paper we classify the greedy routing in sensor networks into two categories, strong greedy routing and weak greedy routing. Most existing work mainly focuses on weak greedy routing over geographic location network or strong greedy routing over greedy embedding network. It is a difficult job and needs much cost to obtain geographic location or greedy embedding of the network. We propose a light-weight Tree-based graph embedding (TGE) for sensor networks. Over the TGE, we design a weak greedy routing protocol, TGR. TGR can archive good performance on path stretch factor and load balance factor.

Keywords: WSN, Greedy routing, Graph embedding, TGR

1. Introduction

Wireless sensor networks (WSN) are deployed in real-world for monitoring events and collecting data from the environment [1,2]. The sensor node limitations in power, computation, storage and bandwidth lead the sensor network to be very different from the traditional networks, especially in the data forwarding and routing aspect. For example, in the Internet, route table can be used for data routing and forwarding, which is the core of the Internet routing protocol. The routing in wireless sensor networks, however, often adopts stateless routing protocol [3] rather than route table based routing protocol because of the above mentioned limitations.

Greedy routing is one kind of stateless routing widely used in real deployed wireless sensor networks [3,4]. The most popular greedy routing is based on geographic information, which is called geographic greedy routing. In such protocol, the current node often selects the nearest node to the destination as the next hop to transmit data. The biggest challenge in geographic greedy routing is the local minimal problem, when the current node cannot find a nearer node than itself to the destination even a path existing from the current node to the destination. The local minimal problem is caused by the "hole" [5,6] or the shape irregular of the network. In order to solve this problem, two approaches are proposed. The first solution tries to remedy the greedy routing rules but still relies on the original geographic information. The face routing is a classical example of the first solution. In the face routing, when the current node cannot find the next hop node by greedy routing rules, it gives up the greedy routing but adopts face routing in order to detour the "hole". The literature [3] proposes how to implement face routing in sensor networks by planarizing the sensor network and using right (or left) hand rule. The literature [4] concludes several different face routing protocols and proves that GFG and GOAFR + + can be used in any planar graph and are loop free protocols.

On the other hand, the second solution does not give up the greedy motivation, but tries to find a new embedding for the network that is satisfying the greedy characteristic [7]. A greedy embedding of an undirected graph G in a metric space (X, d) is a mapping $f: V(G) \rightarrow X$ with the following property: for every pair of distinct vertices $s,t \in V(G)$ there exists a vertex u adjacent to s such that $d(f(u),f(t)) < d(f(s),f(t))$ [8]. Unfortunately, it is not true that every finite graph has a greedy embedding into the Euclidean plane [8]. Even if there exist such embedding into the Euclidean plane for certain finite graphs, it is a difficult work to assign such embedding to a sensor network in a distributed manner. The literature [7] embeds the sensor network into hyperbolic plane by constructing a spanning tree in the network. This work can achieve greedy routing over the spanning tree. But it wastes most links of the original network and cannot reach load balance.

Inspired by these two solutions, we classify the greedy

routing into two kinds. One is strong greedy routing, *which does not give up the greedy motivation*. The other one is weak greedy routing, *which may give up greedy motivation when the greedy approach does not work*. By this classification, the embedding approach is strong greedy routing and face routing is weak greedy routing. The existing weak greedy routing protocol solution is still based on the geographic information. As well known, it is also very difficult to obtain the geographic location information of all nodes in sensor network.

In our work, we propose a new weak greedy routing method. It does not need the geographic location. It establishes a tree-based graph embedding rather than a greedy embedding. In the tree-based graph embedding, every node is assigned an interval label: [*i, r*]. Based on nodes labels, we design a new greedy routing algorithm TGR (Tree-based Greedy Routing). When the current node cannot find a next hop node by the *greedy rule*, it uses a *default rule* to find a next hop node. It is guaranteed that TGR algorithm is a loop-free routing protocol. It means that by using TGR algorithm, any source can find a path to any destination, while there's no node appears along this path twice or more. Another interesting point is that the source node can route to the destination even it only knows part of the label of the destination. By extensive evaluation, our algorithm satisfies small path stretch factor and small load balance factor.

The rest of this paper is organized as follows. We discuss graph labeling and graph embedding in Section 2. Section 3 illustrates the establishment of Tree-based Graph Embedding. Section 4 describes how to design weak greedy routing in TGE. Section 5 presents extensive simulation results that show the performance of TGR. We conclude this work in Section 6.

2. Preliminaries

In this work, we take the wireless sensor network as a finite graph. We use some techniques on graph labeling and graph embedding in graph theory.

2.1. Graph Labeling Scheme

A graph labeling scheme is an assignment of labels to the vertices or the edges of a graph subject to certain conditions [9]. There are many researches focused on such area from 1960s. The labels can be integers, integer intervals [10] or bits. There are different forms of graph labeling according different motivations, such as distance labeling, graceful labeling and harmonious labeling, *etc*. In our work, we only label the node and assign a unique integer pair to each node. An integer pair also can be taken as an integer interval. It can be formulated as follows, $L: N \rightarrow N^2$.

2.2. Graph Embedding

Graph embedding [11,12] is a technique for map- ping a guest graph G into a host graph H in graph theory. It is defined in [12] as follows. An embedding of the graph G (the guest graph), consists of two mappings: (1) The node-assignment function α maps the set of nodes in G one-to-one into the set of nodes in H. (2) The edge-routing function ρ assigns to each edge $\{u,v\} \in E(G)$ a path in H that connects nodes $\alpha(u)$ and $\alpha(v)$.

For sensor network, there are many researches on how to build a tree structure among the whole network. For tree structure, the node only keeps its parent information (the root has no parent) and children's information, which are all its first hop neighbors. Firstly our work maps a shortest path tree into a sensor network. Then a labeling process is running by visiting the tree.

3. Tree-Based Graph Embedding

In our work, we have the following assumptions. Firstly, a wireless sensor network is a connected graph. Secondly, the node in a wireless sensor network does not know its own and other nodes' location information. Thirdly, we also assume that it is a static network or in a period it keeps static, which means no node will be added and no node will fail in a certain period.

The establishment of TGE includes two steps. The first step is building a tree structure for the network and also counts the total number of sensors. Then the labels are assigned from the root using top-down approach at the second step.

3.1. Counting Nodes Number by Spanning Shortest Path Tree

At first, a node is selected randomly as root node. Then the root node broadcasts a "HELLO" message to other node. The other node figures the shortest hop number to the root node. During this process, each node also selects one neighbor node whose hop number is less than itself as its parent node. At last a spanning shortest path tree is established in the network.

After the SPT is established, every intermediate node except the leaf nodes can be seen as a root of one sub-tree. Then each leaf node initials and sends a "COUNTER = 1" message to its parent. When the parent of all leaves receives the "COUNTER = 1" message, it sends a "COUNTER = *m*" message to its parent, where m equals the number of its children plus 1. All the intermediate nodes do the same operation. At last the root can

receive a "COUNTER = n – 1" message and counts the total nodes number n of the network.

During these two processes, each node only sends 2 messages at most. The first one is the "HELLO" message and the second one is the "COUNTER = i" message. But every node may receive several pieces of messages for both "HELLO" and "COUNTER = i" message. When considering transmission and receiving cost both, the cost of the whole network is $(2 + d)\cdot n$, where d is the average node degree of the network.

3.2. Label Assignment

After the first step, the SPT is established and the root node figures out the total number of the network. Then the root node initials the label assignment process. Initially, the root node sets its label in the form of a interval [1, n]. The root node also knows the nodes number of each sub-tree rooted by each of its children nodes. Suppose it has k children nodes C_1, C_2,...,C_k. C_i. count stands for the nodes number of the sub-tree rooted by C_i . The root keeps 1, the left boundary of interval [1, n], and divides the interval [2, n] into k sub-intervals in proportion to C_1.count, C_2.count, ..., C_k. count. For example, we can assign [2, C_1.count + 1], [C_1.count + 2,C_1.count + C_2.count + 1],...,[$n - C_k$.count + 1, n] to C_1,C_2 ,...,C_k separately. More generally, for the intermediate node N, if its label is [i, r] and it has l child nodes C_1,C_2,...,C_l. Then we can assign [i + 1,i + C_1.count], [I + C_1.count + 1, I + C_1.count + C_2.count],...,[$r - C_l$.count + 1, r] to C_1, C_2,...,C_l, the same procedure as the root node.

After the label assignment process, the Tree-based Graph Embedding is also established. **Figure 1** shows an example of the TGE network. In the TGE network, each node has a label [i,r], which is also an interval. We call the left boundary i as the ID of the node and r is called the range of the node. From the process of the label assignment, we can see the integer interval [i, r] of the intermediate node N with label [i, r] includes all the nodes IDs of the sub-tree rooted by N.

4. Routing Algorithms over TGE

Suppose the source node[1] is S: [i_S, r_S] and the destination is D: [i_D, r_D]. For the source node S, when it needs to send a packet to D, it can only get the ID of node D, such as by Hash function. There are two cases for S and D as follows,

1) *Inclusion case*: $i_D \in [i_S, r_S]$.
2) *Separation case*: $i_D \notin [i_S, r_S]$.

[1]The task of the routing is to find a next hop node to forward the data. When the next hop node is selected, we take it as source node. The source node means current node hereafter.

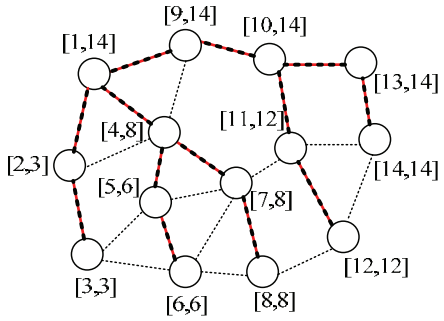


Figure 1. The TGE network and node labels.

For the *inclusion case*, there must exist one child node C: [i_C, r_C] of S, s.t., $i_C \le i_D \le r_C$. Then the source node S can send its data or query to C directly. But for the *separation case*, the source node has no such child node to be the next hop node. About the *separation case*, we have the following three approaches.

4.1. TBR: TGE-Based Basic Routing Algorithm

It is obvious that the root node: [1, n] knows how to find any other node in the network by routing along the spanning tree. So the basic idea for source node to deal with the *separation case* is sending its packet to its parent node on the tree until meeting the *inclusion case*.

4.2. TBHR: TGE-Based Basic Routing Algorithm with One-Hop Information

In the above basic routing, we only use the information of the spanning tree. It means that the path linking any two nodes is a path in the tree. In some cases, however, the node can get more information from other one-hop neighbors that are not its parent or children nodes. As in the figure 1, when node [7,8] wants to find node [12,12], it can find that node [11,12] covers node [12,12], then it can send its data to node [11,12] rather that to its parent node[4,8].

4.3. TGR: TGE-Based Greedy Routing Algorithm

4.3.1. Greedy Function

The greedy routing mainly focus on the Separation case $i_D \notin [i_S, r_S]$ for the source node S: [i_S, r_S] and the destination D: [i_D, r_D]. For the first case that $i_D \in [i_S, r_S]$, we also use the basic routing algorithm. The main task for our weak greedy routing is to designing a local monotonous function. First we give the following function

$$f(x,y)=\begin{cases}(i_D-x)\cdot \mathrm{sgn}(y-r_S) & if \;\; i_S < x < i_D \\ (x-i_D) & if \;\; i_D < x < i_S\end{cases}$$

where $sgn(n) = 1$ when $n > 0$; $sgn(n) = -1$ when $n < 0$ and $sgn(0) = 0$. This function satisfies: $f(x_1,y_1) < f(x_2, y_2)$ when (1) $i_D < x_1 < x_2 < i_S$ or (2)$i_S < x_2 < x_1 < i_D$ and $y_1 > r_S$, $y_2 > r_S$. It means that f is monotonous in an open integer interval.

4.3.2. Routing Rules

Firstly, we define Candidates neighbors as $C = \{N|N \in N(S), i_S < i_N < i_D$ when $i_S < i_D$ or $i_D < i_N < i_S$ when $i_D < i_S\}$, $N(S)$ stands for all neighbor nodes of S in the network. By greedy function, we design routing rules for Separation case as follows,

1) Greedy Rule: if $C \neq \phi$, the next hop node is the node with $min\{ f(i_N, r_N) > 0, N \in C \}$.

2) Default Rule: if $C = \phi$, the next hop node is the source's parent node.

The greedy rule is illustrated in **Figure 2**. By greedy and default rules, we design TGR, TGE-based greedy routing algorithm. TRG is a weak routing algorithm, because when it can not find a next hop node it gives up the greedy rule and uses default rule instead.

5. Evaluation

In this section, we evaluate the performance of our methods. The first important problem is about the length of the path connecting the source and destination pairs (S-D pairs). The load balance is another performance factor. First, we compare three algorithms on the path stretch factor. Then we compare their load balance when there are many S-D pairs. We also evaluate the usage of cross links that are not on the embedding tree. Our testing network has 500 nodes randomly scattered in a square area. The average degree is about 14. The diameter of the network is 16 hops.

5.1. Path Stretch

In a connected network, any S-D pair has a shortest path connecting them. In a stateless sensor network, however, it is difficult to find the shortest path without flooding. In this evaluation, we randomly select 1000 S-D pairs, and simulate their paths generated by TBR, TBHR and TGR respectively. In **Figure 3**, the X-axis stands for the length of the shortest path between S and D. **Figure 3(a)** plots the length of every actual routing path for TBR, TBHR and TGR. From **Figure 3(b)**, we figure out the average length of the routing path with the same length of the shortest path. We can see that for the two nodes with distance less than 8, the average case of the TGR is shorter than the TBHR; and when the distance is less than 11, the TGR is shorter than TBR. For the two nodes with distance longer than 8, the average case of TBHR is shorter than TGR, and when the distance is longer than 11, the average case of TBR is shorter than TGR. We can see that (1) TBHR is always better than TBR, that means the one-hop information is very important; (2) when the distance is larger, the path length generated by TGR is longer than TBR and TBHR. That is because for the long distance two nodes, their path length by traveling the tree (TBR and TBHR) approximates with their shortest path length.

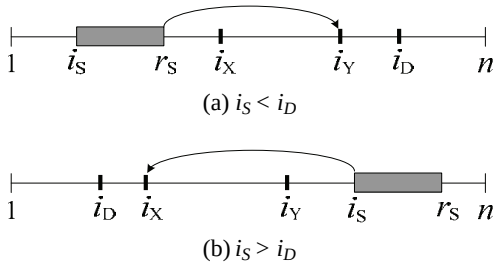


Figure 2. Greedy rule.

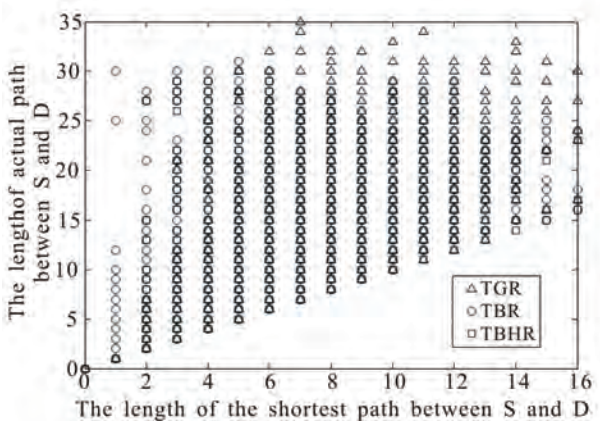
5.2. Load Balance

In **Figure 4(a)**, there are 1500 pairs of S-D pairs transmissions randomly selected in the network. If a node has forwarded a packet, its load counter adds one. After the simulation, we order the nodes according to their load counter decreasingly. We compare the load counter distribution about TBR, TBHR and TGR. It is obvious that there are more than 50 nodes among 500 nodes, which the load counter of TBR is about twice as the TGR. The load counter lines of TBR and TBHR are very close. After that, all three lines are smoothly and approximately. It is because in TBR and TBHR, the root and the nodes near the root should transmit more packets. In TGR, some source nodes can find their destinations by not passing the root node or the low-level nodes in the embedding tree, even the source and destination belong to two independent sub-trees. **Figure 4(a)** shows the transmission load from single node respect. For the whole network, we use load balance factor to metric the load performance. The load balance factor can be defined by using the variance of the packets account of all the nodes participated in the routing work,

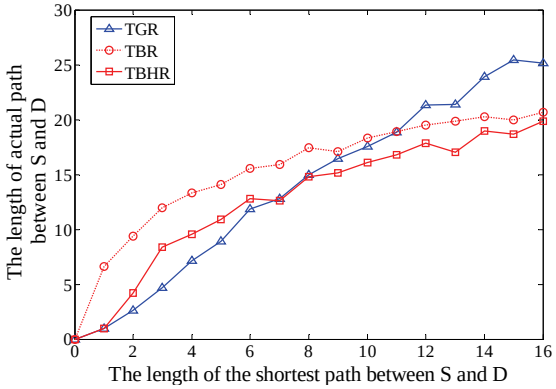
$$\phi = \frac{\sum_{i=1}^{n}(L_i - \bar{L}_i)^2}{n}$$

where L_i is the packets account passing the node i, L_i is the average packets of all nodes that have forwarded some packets, n is the total number of all nodes that have transmitted packets. If φ becomes larger, it means the load balance get worse; if φ becomes smaller, it means the load balance get better. We call φ as the *load balance factor*. In **Figure 4(b)**, for the same network, we randomly select different size of S-D pairs from 50 to 1500

increasing by 50. We can find that all the load balance factor of TBR, TBHR and TGR increasing with the size of the S-D pairs. But the TGR increases slow comparing with TBR and TBHR.

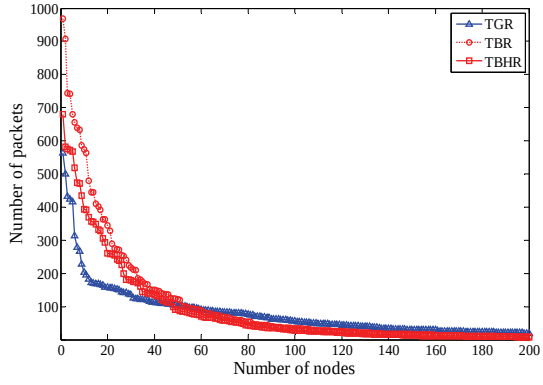


(a) The distribution of the length of actual path.

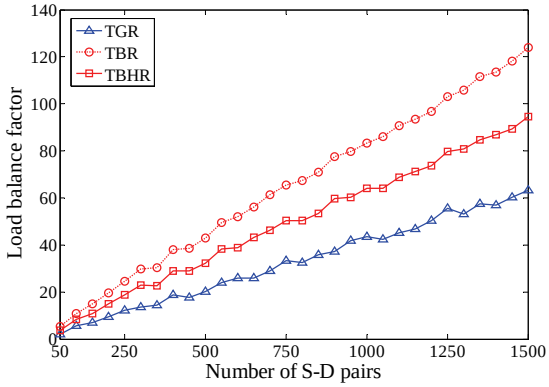


(b) The average length of actual path.

Figure 3. The average length of actual path vs. the length of the shortest path between S and D.



(a) The order of packets passing nodes.



(b) The load balance factor of different size of point-to-point transmissions.

Figure 4. Load balance comparison.

5.3. Cross Link Usage

The links of a spanning tree only covers a small part of all the links of the whole network (as **Figure 1**). TBR wastes all cross links not on the SPT. For TBHR, it has at most one chance to use cross link not on the SPT. We compare the cross link usage about TBHR and TGR. From **Figure 5**, we find that TGR uses more cross links than TBHR. The percent of cross link usage of the TGR can achieve 40% in average. This can also explain why the *load balance factor* of TGR is better than TBR and TBHR.

6. Conclusions

In this paper, we classify the greedy routing in sensor networks into two kinds, strong one and weak one. Then we propose a new weak greedy routing (TGR) over a tree-based graph embedding (TGE). TGE is a light-weight labeling scheme and graph embedding technique

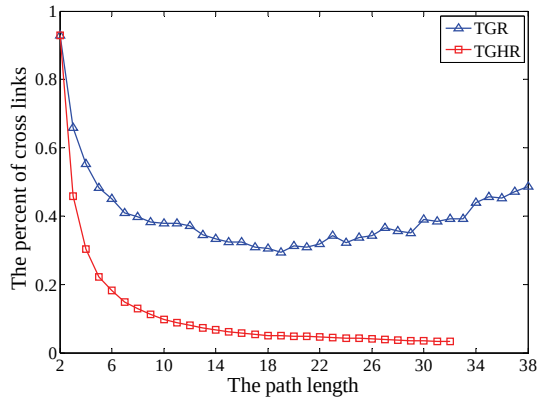


Figure 5. The label embedding network.

It assigns each node an integer interval. Base on nodes labels, TGR can achieve good performance in path stretch factor and load balance factor for a static connected network. In future network, we will study how to implement TGE and TGR in dynamic networks for resolving nodes adding and nodes failure.

7. Acknowledgements

The research was partially supported by the Program for New Century Excellent Talents in University (NCET-08-0145), the National Natural Science Foundation of China under Grant No.60736013. We would also like to thank the anonymous reviewers for their constructive comments.

8. References

[1] F. Ren, H. Huang and C. Lin, "Wireless Sensor Networks," *Journal of Software*, Vol. 14, No. 7, 2003, pp.1282-1291.

[2] L. M. Sun, J. Z. Li, Y. Chen and H. S. Zhu, "Wireless Sensor Network," Tsinghua University Press, Beijing, 2005.

[3] B. Karp and H. T. Kung, "Gpsr: Greedy Perimeter Stateless Routing for Wireless Networks," *The* 6*th ACM Annual International Conference on Mobile Computing and Networking*, Boston, 2000, pp. 243-254.

[4] H. Frey and I. Stojmenovic, "On Delivery Guarantees of Face and Combined Greedy Face Routing in Ad Hoc and Sensor Networks," *The* 12*th ACM Annual International Conference on Mobile Computing and Networking*, Los Angeles, 2006, pp. 390-401.

[5] M. Li and Y. Liu, "Rendered Path: Range-Free Localization in Anisotropic Sensor Networks with Holes," *The* 13*th ACM Annual International Conference on Mobile Computing and Networking*, Québec, 2007, pp. 51-62.

[6] X. B. Wu, G. Chen and K. D. Sajal, "Avoiding Energy Holes in Wireless Sensor Networks with Non-Uniform Node Distribution," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 19, No. 5, May 2008, pp. 710-720.

[7] A. Cvetkovski and M. Crovella, "Hyperbolic Embedding and Routing for Dynamic Graphs," *the* 28*th Conference on Computer Communication*, Rio de Janeiro, Brazil, April 2009, pp.1647-1655.

[8] C. Papadimitriou and D. Ratajczak, "On a Conjecture Related to Geometric Routing," *Theoretical Computer Science*, Vol. 344, No. 1, 2005, pp. 3-14.

[9] A. G. Joseph, "A Dynamic Survey of Graph Labeling," *the Electronic Journal of Combinatorics*, Vol. 16, 2009, pp. 1-219.

[10] J. V. Leeuwen and B. Tan, "Interval Routing," *Computer Journal*, Vol. 30, 1987, pp. 298-307.

[11] J. Newsome and D. Song, "Gem: Graph Embedding for Routing and Datacentric Storage in Sensor Networks without Geographic Information," *the* 1*st International Conference on Embedded Networked Sensor Systems*, Los Angeles, 2003, pp. 76-88.

[12] A. L. Rosenberg and L. S. Heath, "Graph Separators, with Applications," Kluwer Academic/Plenum Publishers, Norwell, Massachusetts, 2001

Wireless Sensor Network, 2010, 2, 689-697
doi:10.4236/wsn.2010.29083 Published Online September 2010 (http://www.SciRP.org/journal/wsn)

An Adaptive Key Management Framework for the Wireless Mesh and Sensor Networks

Mi Wen[1], Zhi Yin[1], Yu Long[2], Yong Wang[1]
[1]*Department of Computer Science & Engineering; Shanghai University of Electric Power; Shanghai, China*
[2]*Department of Computer Science & Engineering; Shanghai Jiaotong University, Shanghai, China*
E-mail: *superwm_9@yahoo.com*
Received June 30, 2010; *revised July* 31, 2010; *accepted August* 30, 2010

Abstract

Wireless sensor networks (WSNs) and wireless mesh networks (WMNs) are popular research subjects. The interconnection of both network types enables next-generation applications and creates new optimization opportunities. Currently, plenty of protocols are available on the security of either wireless sensor networks or wireless mesh networks, an investigation in peer work underpins the fact that neither of these protocols is adapt to the interconnection of these network types. The internal cause relies on the fact that they differ in terms of complexity, scalability and network abstraction level. Therefore, in this article, we propose a unified security framework with three key management protocols, MPKM, MGKM, and TKM which are able to provide basic functionalities on the simplest devices and advanced functionalities on high performance nodes. We perform a detailed performance evaluation on our protocols against some important metrics such as scalability, key connectivity and compromise resilience, and we also compare our solution to the current keying protocols for WSNs and WMNs.

Keywords: Wireless Mesh Sensor Network, Key Management, Adaptive Security, Group Key

1. Introduction

The success of wireless technologies today caused the international wireless network research community to have high hopes for the future. Wireless mesh sensor network (WMSN) is a new architecture that merges advantages of wireless mesh networks (WMN) and wireless sensor networks (WSN), especially on scalability, robustness and balanced energy dissipation [1]. Wireless sensor networks and wireless mesh networks are popular research subjects. The interconnection of both network types enables next-generation applications and creates new optimization opportunities [2]. Many application scenarios could benefit from a successful and optimal interconnection between WSNs and WMNs. For example, a wireless mesh network can be used as a backbone for collecting sensor data from remote sensor clusters, or, resource intensive calculations with sensor data may be performed on a mesh router instead on a sensor node. Although plenty of research is available on all aspects of either wireless sensor networks or wireless mesh networks, little information is available on the interconnection of these network types. Their difference between WMNs and WSNs in wireless technologies, addressing protocols, routing strategies and security mechanisms make an effective interconnection be challenging.

Especially, WMSNs are always deployed in hostile environments to track target, monitor battlefield, detect intruder or do some scientific explorations and the openness of the wireless environment makes security in WMSNs a critical concern in the deployment of such group applications. In wireless communication environments an adversary not only can eavesdrop the radio traffic in a network, but also can intercept the exchanged data. To prevent the malicious node impersonating good nodes for spreading misleading data intentionally, secret keys should be used to achieve data confidentiality, integrity and authentication between communicating parties [3]. But in WMN networks, security and trust is most often guaranteed using either pre-shared keys, or by relying on certificate based encryption techniques [4]. Because of the limited capacities of sensor nodes, the security approaches used in WMNs are not suitable for WSNs [5]. Some sensor nodes might be unable to implement any certificate based security mechanism at all. Therefore, the development of adaptive key management

protocols is a promising approach to enable low end devices to participate in heterogeneous network architectures securely.

Adaptive key management protocol is an effective approach to provide efficient and secure interconnection, while respecting the individual characteristics of each network type. The main difficulty with adaptive key management protocols is the creation of a basic key management protocol version that can be deployed on a very basic network node. Current popular key management protocols in WMNs such as the [6-8] are relatively complex. Even though the performance of sensor nodes will increase over time, there will always remain a class of devices that is unable to run these complex protocols. Therefore, there is a need for novel, simple techniques that are able to provide basic functionalities on the simplest devices and at the same time they can be extended to support advanced functionalities on high performance nodes. Thus, an adaptive and modular key management approach is needed.

In this paper we present a unified security framework that embodies three key management protocols which can provide adaptive security for WMSNs for the need of the applications. The framework includes three key management protocols: a) the Matrix based Pairwise Key Management (MPKM) protocol for sensor nodes with limited resources. b) the Matrix based Group Key Management (MGKM) protocol for the network with sinks or cluster heads except the sensor nodes; c) the Threshold Key Management (TKM) protocol for the network with mesh nodes in addition. All of these three protocols are interrelated elements: MGKM can be extended from MPKM, and TKM can be extended from MGKM. Therefore, the accession of MGKM and TKM in the WMSNs will not increase the storage or communication overhead of sensor nodes.

2. Related Works

Key Management is one of the main challenges in securing wireless networks, and has been addressed by many authors. In this section, we present an overview of some approaches and protocols for keying management in both WSNs and WMNs respectively.

2.1. Key Management in WSNs

To date, the key management protocols in sensor networks can be mainly classified into two types: pairwise key management protocols and group key management protocols. In pairwise key management protocols [9-14], each pair of communication nodes should establish a shared key. One attractive idea in the pairwise key management is key pre-distribution, *i.e.*, pre-installing a limited number of secrets in sensor nodes prior to actual deployment; after the deployment, if two neighboring nodes have some common keys, they can setup a secure link by the shared keys. While in the group key management protocols [15-17], the key idea is to broadcast information that is useful only for trusted nodes. Combined with its pre-distributed secrets, this broadcast information enables a trusted sensor node to reconstruct a group key. Most pairwise key and group key management protocols in WSNs are based on symmetric key cryptography, such as Du's [12] key Matrix based, Camtepe's [9] Combinatorial Design based, Liu's [13] polynomial based protocols. These solutions are designed to sustain severe computation power, storage, mobility, and energy constraints, and as a result have limited scalability and robustness. Although some research [18] shows that the right selection of algorithms and associated parameters along with code optimization can make public key cryptography feasible for sensor networks. For example, the ECC and RSA based key management protocols. The major shortcomings of them are the associated expensive computation and the high probability of likely penetration by malicious agents. Also all current asymmetric key related studies only support their feasibility for WSN's. Unfortunately, as we know, none of current works propose complete key management infrastructure compatible public and private key cryptography.

2.2. Key Management in WMNs

Secure group communication is a mature research area and has a large body of research literature. The main objective of a secure group communication protocol is to ensure the data confidentiality against outsiders such that only legitimate group members can recover the group data. Existing solutions for wired networks [19-21] are not well suited for WMNs as they fail to take into consideration the multi-hop communication paradigm featured by WMNs, as well as the communication security among mesh clients within the coverage of a mesh router. These protocols also do not exploit unique features of WMNs, such as the broadcast nature of wireless communication. ARSA [7] proposes attack-resilient security architecture for WMNs, which uses ID-based cryptography (IBC). SeGrOM [8] propose a new protocol framework for secure group overlay multicast in WMNs. LSSS [6] presents an ideal linear multi-secret sharing protocol, by using monotone span programs. Though, they achieve efficient and secure group communication in WMNs. They can not be employed in the WSNs due to their expensive energy consumption and also they can not offer modular security for WMSNs.

In general, none of the existing protocols considered the unique features of WMSNs, such as coexistence of resources constrained sensor nodes and powerful mesh nodes, increasing scalability when remote cluster sensors get interconnected thanks to the presence of a WMN, all of which can be leveraged for designing more optimized protocols. Our work tries to fill this gap by designing such a complete key management infrastructure specially for WMSNs based on our previously key management protocols [14,17].We will take into account the diversity of nodes' ability and propose a unified key management framework, which includes simple techniques that are able to provide basic functionalities on the simplest sensor devices and at the same time they can be extended to support advanced functionalities on high performance mesh nodes.

3. System Model and Assumptions

3.1. Network Model

Define our target network environment is the interconnection of WSNs and WMNs, called WMSNs. The WMNs include a set of static wireless routers, called mesh nodes (MN), organized in a backbone network and communicating through multi-hop wireless links. Mobile clients (MC) connect to the wireless mesh through a local access router, called access point (AP), and communicate with each other through the wireless mesh. While the WSN has the hierarchical architecture consisting of numerous sensor nodes (SN) grouped in clusters and each cluster has a cluster head (CH), which is responsible collecting and merging local data from sensor nodes and send it to mesh nodes. Clusters of sensors can be formed based on various criteria such as location, communication range, resource and energy capabilities, *etc*. (See **Figure 1**). Resource intensive calculations with sensed data may be performed on a MN. MN here can be considered as an actuator node in WSNs and can take immediate response when monitoring some abnormal phenomena in WSNs. Many application scenarios could benefit from successful and optimal WMSNs. For example, the WMNs can be used as a backbone for collecting sensor data from remote sensor clusters. For clarity, we describe the terms in the scope of this article is specified as follows:

Sensor nodes are network nodes with limited capabilities in terms of processing power, memory capacity and bandwidth, equipped with a sensor and/or actuator chip. As such, a sensor node can be a source of data in a network, but could as well be used as intermediate node to forward data from one sensor device to another, or to a data collection device, called a cluster head (see **Figure 1 (ii)**). Sensor nodes are small sized and limited in cost. With WSN, all forms of wireless networks between sensor devices are indicated [1]. These sensor networks are self-forming, and are used to gather data in places where the use of cabled sensors is hard, costly or undesired. No restriction is made based on network size or topology: both single hop networks between SNs and a CH, and complex multi-hop networks with meshed topologies are considered.

Mesh nodes are relatively powerful networked nodes, equipped with relatively powerful wireless interfaces and thus are able to transmit and receive at higher bandwidths than sensor nodes. With WMN or wireless mesh networks, all forms of wireless networks between mesh nodes are indicated. Again, there are no restrictions on the topology. Mesh networks are often used as a wireless backbone for the interconnection of end user devices. WMNs might also offer additional functionality to the client networks; for example, provide an uplink to the Internet (see **Figure 1(i)**). Mesh networks are self-forming and self-healing, and are therefore an ideal solution to provide connectivity in places where cabled networks cannot easily be installed. Furthermore, because of their self-organizing character, mesh networks can be rolled out fast, making them ideal candidates to be used as emergency network infrastructure.

3.2. Definitions and Notations

In the following section, we give some definitions and notations used in this paper unavoidable.

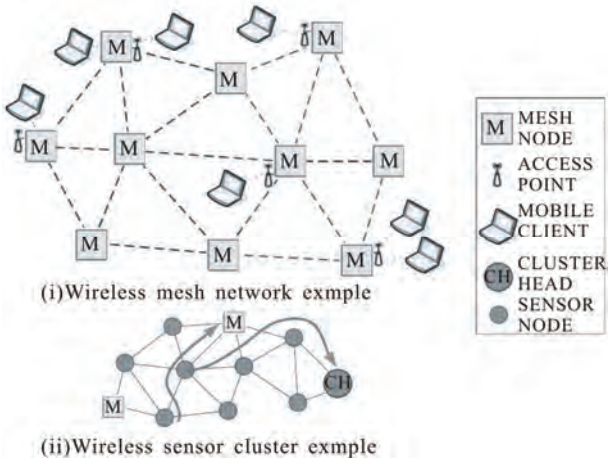


Figure 1. Wireless mesh and sensor network architecture example.

4. Proposed Framework

In this section, we first describe MPKM, our basic key management protocol handling the pairwise key establishment for the resource limited sensor nodes. We then describe two additional protocols, (a) MGKM, a key management protocol handling the group key of the WSNs and (b) TKM, a key management protocol handling the key sharing in WMN using the asymmetric cryptography. All these protocols together can manage the key establishments in WMSNs. Since MPKM and MGKM are previously proposed in [14] and [17]. Here we include them only to form the unified key managemen framework. Thus, we will focus on the TKM protocol in the latter of this paper. Due to the node resource limitation, MPKM and MGKM are based on symmetric key cryptography while TKM is based on asymmetric key cryptography. All of these three protocols are interrelated elements: MGKM can be extended from MPKM and TKM can be extended from MGKM. (See **Figure 2**). Therefore, the accession of MGKM and TKM in the WMSNs will not increase the storage or communication overhead of sensor nodes. This is one of the advantages of our framework and it just satisfies the requirement of key management technique in WMSNs, scalable and lightweight.

4.1. The Matrix Based Pairwise Key Management (MPKM) Protocol

In MPKM, each sensor node is programmed according to the application requirements before network deployment. As we all know, Blom proposed a key distribution approach [12], which allows any pair of nodes in a network to be able to find a pairwise secret key. As long as no more than nodes are compromised, the network is perfectly secure. For the sake of key updating, we modify the Blom's symmetric matrix construction in [14]. We briefly describe how to use our modified version of Blom's key distribution approach to establish paiwise key as follows. Some used notations in this paper are given in **Table 1**.

The base station (BS) in WSNs (acting as a trusted server) first computes a $n*n$ matrix B over a finite field $GF(q)$, B is considered as the public information, q is a prime, and $q<n$. One example of such a matrix is a Vandermonde matrix whose element $b_{ij}=(g^j)^i \bmod q$, where g is the primitive nonzero element of $GF(q)$ and g^j is the jth column seed. That means:

$$B=\begin{bmatrix} 1 & 1 & 1 & & 1 \\ g & g^2 & g^3 & & g^n \\ & & & & \vdots \\ g^{n-1} & (g^2)^{n-1} & (g^3)^{n-1} & \cdots & (g^n)^{n-1} \end{bmatrix}$$

Table 1. Table type styles.

N_i	A sensor node i (i=,1,…,n), where N_n is the trusted dealer and it has more power than normal sensor node, i.e. the cluster head in a cluster.
s_i	Row seed of the matrix D; the row seed used in each row i of matrix D should not bigger than s_i.
g^i	Column seed of matrix B
K_{ij}	The pairwise key between node N_i and N_j
K_G	The group key of the initial set N={ N_1, N_2, ..., N_n }
GID_i	The group or cluster identity of cluster i
ID_i	The identity of sensor node i
K_{Gi}	The group key of the sensor cluster i
sk	The secret key to be shared by the mesh nodes
sk_i	The secret key share of mesh node i

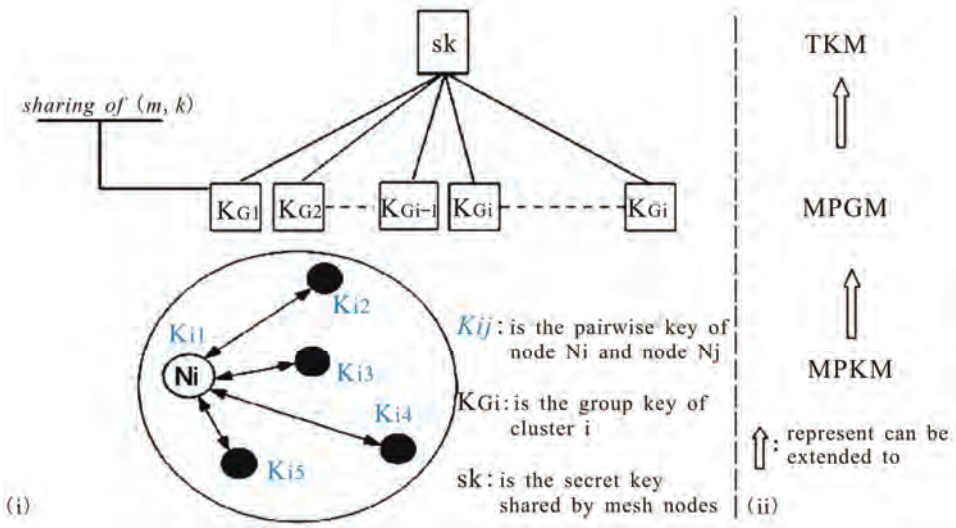


Figure 2. Overview of our key management framework.

This construction requires that $n^2 < \varphi(q)$ *i.e.*, $n^2 < q-1$. Since *B* is a Vandermonde matrix, it can be proved that the n columns are linearly independent when $g, g^2, g^3 \cdots, g^n$ are all distinct.

Next, the BS generates n row seeds $s_1, \cdots s_n$, where $s_i (i = 1, ..., n)$ is the random prime number of *GF*(*q*) and it is only known to the powerful node (N_n) in the network, *e.g.*, the cluster head of WSNs. And then BS creates a random $n*n$ symmetric matrix *D* over *GF*(*q*). Each row of the *D* is composed of hash values of the row seeds. Differing from the construction of matrix *B*, the elements in symmetric matrix *D* are generated as follows:

$$for(i = 1; i \le n; i++)$$
$$for(j = 1; j \le n; j++)$$
$$\{if\ (i > j)\ d_{ij} = H^i(s_j); else\ d_{ij} = H^j(s_i);\}$$

where d_{ij} is the element in matrix *D*. An example of matrix *D* with size $3*3$ is shown as follows:

$$\begin{bmatrix} H^1(s_1) & H^2(s_1) & H^3(s_1) \\ H^2(s_1) & H^2(s_2) & H^3(s_2) \\ H^3(s_1) & H^3(s_2) & H^3(s_3) \end{bmatrix}$$

At last, the BS computes a $n*n$ matrix $A = (DB)^T$, where *T* indicates a transposition of the matrix. The elements in matrix *A* denote as a_{ij}, where $a_{ij} = \sum_{\beta=1}^{n} d_{j\beta} b_{\beta i}$. The matrix *B* is public while the matrix *D* is kept secret by the base station. Since *D* is symmetric, the key matrix $K = AB$ can be written as:

$$K = (DB)^T B = B^T D^T B = B^T DB = (AB)^T = K^T$$

Thus *K* is also a symmetric matrix and $K_{ij} = K_{ji}$, where K_{ij} is the element of *K* at *i*th row and *j*th column. We take K_{ij} (or K_{ji}) as the pairwise key between node N_i and node N_j. To carry out the above computation, nodes N_i and N_j should be able to compute K_{ij} and K_{ji} respectively. This can be easily achieved using the following key pre-distribution procedure, for node N_i:

1) Store the *i*th row of matrix *A* at node N_i, denoted as $r_i(A)$, *i.e.*, $\mathrm{r_i}(\mathrm{A}) = [\mathrm{a_{ij}}], (j = 1, \cdots, n)$.

2) Store the *i*th column seed g^i of matrix *B* at N_i.

After deployment, each node has a piece of secret information as described above. When nodes N_i and N_j need to find the pairwise key between them, they first exchange their column seeds of matrix *B* (since *B* is the public information, it can be sent in plaintext). Then, by using the preloaded secrets, they can compute K_{ij} (or K_{ji}) respectively as: $K_{ij} = \sum_{\beta=1}^{n} a_{i\beta} b_{\beta j}$. It can be proved that the above protocol is *n*-secure because all the rows in *D* are linearly independent. And this property guarantees the uniqueness of the pairwise keys in the cluster.

4.2. The Matrix Based Group Key Management (MGKM) Protocol

When nodes with better resources, named as CHs, are deployed in the network, they can be used to collect and merge local data from sensor nodes and send it to mesh nodes. Mesh nodes then distribute the required information to the end user clients. Now, we assumed that after deploying the new nodes into the operating WSNs, the clusters can be formed based on various criteria such as capabilities, location, and communication range *etc*. [17].

Now, without loss of generality, let $N = \{N_1, N_2, ..., N_n\}$ be the initial set of participants in each cluster group that want to generate a group key. Assume that there are *n-1* sensor nodes and a powerful node N_n in a cluster (N_n may be a CH). The detailed steps of the group distribution are presented as follows.

Step 1: Initially, each node $N_i (1 \le i \le n-1)$ is preloaded a row $r_i(A)$ from matrix *A* and column seed g^i as described in Subsection 4.1. Then, after deployment, each node pre-computes K_{in}, K_{ii} (*i.e.*, $K_{ii} = \sum_{\beta=1}^{n} a_{i\beta} b_{\beta i}$) an K_{ii}^{-1}. N_i sends the enciphered message (N_i, $C_i = E_{K_{in}}(K_{ii})$) to node N_n and keeps K_{ii}^{-1} in its local memory. K_{in} is the pairwise key between node N_i and N_n. || stands for message concatenation.

Step 2: Node N_n computes K_{nn} as above. Upon receiving each $(\mathrm{N_i}, C_i)(1 \le i \le n-1)$, node N_n deciphers them and computes $x_i = K_{nn} K_{ii}$. Next, node N_n computes $K_G = K_{nn} \prod_{i=1}^{n-1} x_i$. Finally, the powerful node N_n broadcasts (N_n, x_1... x_{n-1}) to other nodes.

Step 3: On receiving the broadcast messages, each node $\mathrm{N_i} (1 \le i \le n-1)$ computes the common group key $K_G = x_j K_{jj}^{-1} \prod_{i=1}^{n-1} x_i$.

Note that the client N_i may pre-compute K_{ii}^{-1} to reduce the computational load. Until now, storage limitation is becoming less of a concerning issue as many add-on memory cards are widely available. And we can prove that the proposed protocol is a contributory group key agreement protocol. Reference [17] for detailed prove process.

4.3. The Threshold Key Management (TKM) Protocol

If a WMN is added to an already existing WSN to collect the sensed messages, should any adjustments to the WSN protocols be made? In general: the less adjustments are to be made on either WSN or WMN protocols, the faster an interconnection can be realized and the sooner an interconnection strategy might be adopted. Moreover, using either a single symmetric key or by relying on certificate based encryption techniques to achieve key management operations for the mesh nodes risks high proba-

bility of key leakage or creates a vulnerable point in the network. Thus, based on the two former key management protocols, MPKM and MGKM, we design a threshold key management protocol for mesh network. In such a system, the group keys of the WSNs will be calculated as a secret key shared by *n* mesh nodes. And the secret key can be recovered by a coalition of *t* mesh nodes.

4.3.1. Preliminaries of Threshold Secret Sharing

The proposed protocol is based on the (*m*, *k*) threshold cryptography [22]. Generally speaking, threshold cryptography is used for distribution of a secret value *S* based on polynomial interpolation, and an (*m*, *k*) threshold protocol allows *m* parties to perform cryptographic operations, so that any *k* parties can jointly perform key discovery whereas (*k*-1) parties cannot derive any information even after collusion. The parameter *k* represents the threshold. A sample threshold cryptography protocol proposed by Shamir can be explained as follows:

Consider the secret *S*, we can store the secret about *S* into *n* shares ($s_1, \ldots, s_m$) via a randomly chosen *k* degree polynomial $f(x) = a_0+a_1x+\ldots+a_{k-1}x^{k-1}$ where $a_0 = S$. Secret shares are obtained by $s_i = f(i)$, $i = 1 \ldots m$. The *m* shares of secrets are simply $\{f(1), f(2),\ldots, f(m)\}$. Given *k* points from the above *m* shares, we can derive the coefficients of $f(x)$ by interpolation and hence calculate the secret $S = \sum_{i=1}^{k} f(i)L_i(0)$, where $L_i(0)$ is the Lagrange coefficient such as

$$L_i(x) = \frac{\prod_{j \neq i}(x - x_j)}{\prod_{j \neq i}(x_i - x_j)}$$

Therefore, the above protocol is a (*n*, *k*) threshold cryptography protocol.

4.3.2. The Proposed TKM Protocol

We assume that there is a Trusted Server (TS, the base station in WSN also can act as a TS) which can calculate the secret key and the key shares for bootstrapping the mesh nodes. And we also assume that when a WMN with *m* mesh nodes is added to an already existing WSN, each mesh node is innocent and cannot be compromised during the first several minutes after deployment since compromising a node takes some time. Based on this, the system initialization process is carried out in two phases: secret key calculation and mesh nodes bootstrapping. In the first phase, the TS will collect the group keys of the sensor nodes and calculate the secret key to be shared by the mesh nodes. Here, we assumed that the messages delivered among sensor nodes and mesh nodes are always encrypted by their group keys, the collaboration of *t* mesh nodes can decrypt them. In the second phase, the TS creates an (*m*, *k*) sharing ($sk_1, sk_2, \ldots, sk_m$) and privately distributes these shares to *m* mesh nodes where ($m < M$) and *M* is the network size.

Secret key calculation phase: We assume that before the WMN is added, the WSNs are classed by *t* clusters, and each cluster has a CH and several SNs. Also, the keys in WSNs are already established by using MPKM and MGKM protocols. Thus, in this phase, the TS

- Step 1: Broadcast a hello message $\{ID_{TS}, hello\}$ to the sensor nodes in WSNs.
- Step 2: On receiving the hello message, each CH or the SNs will reply a message which contains its group keys $\{GID_i, K_{Gi}, ID_{CH}\}$.
- Step 3: By distinguish the different group keys K_{Gi} (i=1,…, t) from the WSNs. The TS calculate the secret key SK for mesh nodes as following:

$Sk = K_{G1} \oplus K_{G2} \oplus \ldots\ldots \oplus K_{Gt}$, where $\oplus$ represents XOR operation.

Mesh nodes bootstrapping phase: In this phase, the TS performs the following operations:

- Step1: Create a random polynomial of degree $k-1$: $f(x) = sk + a_1x + \cdots + a_{k-1}x^{k-1}(mod\ p)$ where *p* is a large prime number and *sk* is the shared private key of the mesh nodes.
- Step2: Calculate and send to each node *i* the corresponding share of (*sk*): $sk_i = f(i)(mod\ p)$. For simplicity *i* is assumed to be an integer and nodes to be initialized range from 1 . . . *m*.
- Step3: Calculate and store locally the decryption supplementary keys S_i for each sensor group as follows, and then deleted the K_{Gi} (i=1,…, t) permanently.

$S_i = sk \oplus K_{Gi}(i = 1,\ldots, t)$

When a mesh node j receives messages from sensor nodes, it should broadcast a request to its neighbors. After collecting *k*-1 valid shares from its neighbors, it combines them with its share in order to issue the secret key *sk*. If the WSN only have one group, then $sk = K_{G1}$, the requester mesh node *j* can use *sk* to decrypt the received messages immediately. If the WSN have two or more groups, the requester node *j* should ask the TS for help. The TS replies with the group i's decryption supplementary keys S_i. Then, the mesh node *j* decrypt the received messages by calculate $sk \oplus S_i$, where $sk \oplus S_i = K_{G1}$.

5. Post Deployment Operations

Network post-deployment issues are critical factors in determining the efficiency of any key management protocol for WMSN specific environment. Each protocols working in correspondence to these issues is explained against the following matrices.

Scalability: Each of the three protocols supports node

additions after network deployment. In case of MPKM and MPGM, when a new node $ID_{(n+1)}$ wants to join the network, the BS will generate the key information for node $n + 1$ and the Real-Time generation (RTG) program in [14] will be triggered to expand the key matrix. Thus, all nodes in its cluster will establish new pairwise key with this node. And periodically their group key also will be updated according to reference [17].

In TKM, when a new sensor cluster wants to join the existing WMSN, here we assume that the pairwise keys and group key in this cluster have been established by using the MPKM and MPGM protocols. The main issue in this procedure is the secret key updating and the key share information updating for existing mesh nodes. There are two types of techniques can address this problem.

1) *Regeneration*: in this approach, the TS first recalculate the secret key by XOR *sk* and the new cluster's group key, here we denote it as K_{Gnew}. So, the new secret key for the mesh nodes in new mesh network is $sk'=sk \oplus K_{Gnew}$. Then, the TS recreates a random polynomial of degree $k-1$: $h(x) = sk' + b_1x + \cdots + b_{k-1}x^{k-1} \ (mod\ p)$ and resends the key share for mesh nodes. Finally, the mesh nodes can establish new secret key to guarantee the network security. The drawback of this approach is that it introduces substantive communication and computation overhead.

2) *Real-Time updating*: This technique relies on the following Homomorphic property. If $(s_1, \ldots, s_n)$ is an (n, k) sharing of S and $(s'_1, \ldots, s'_n)$ is an (n, k) sharing of S', then$(s_1 \oplus s'_1, \ldots, s_n \oplus s'_n)$ is the an (n, k) sharing of $S \oplus S'$, if we set $S' = 0$, then we get a new (n, k) sharing of S.

Now, let $(sk_1,\ldots,sk_m)$ be the (m, k) sharing of sk and K_{Gnew} is the group key of the new cluster, the TS creates a random polynomial of degree $k - 1$: $p(x) = K_{Gnew} + c_1x + \cdots + c_{k-1}x^{k-1} (mod\ p)$ where p is a large prime number. And then the TS calculates the corresponding share of (K_{Gnew}): $K_i = p(i) \ (mod\ p)$ and sends it to each node i. Thus, the mesh nodes can get a new sharing $(sk_1',\ldots,sk_m')$ of sk' where $sk_i' = sk_i \oplus K_i$.

Key connectivity: Key connectivity is described as the number of keys required to be stored on each node for specified level of required network connectivity.

MPKM establishes a pairwise key for each pair of nodes in the same cluster. Since the keying information of the nodes in one cluster is coming from the same symmetric key matrix, each pair of nodes in one cluster can establish a paiwise key. This provides 100% cluster wide connectivity.

MPGM provides good key connectivity on frequent broadcast basis. In a typical information gathering scenario where the primary purpose of the nodes is to gather data and forward it to BS or MN, nodes in one cluster communicate with each other more frequently. Such communication is ensured by a common cluster wide group key. Hence, broadcast based cluster-wide key connectivity is ensured.

In TKM, to encrypt the communication between the sensor nodes and the mesh node needs the sensor's group key, which is established in MPGM. And the decryption of the communication messages needs local collaboration of the neighboring mesh nodes by using the threshold cryptography. All these keys are established and secure. Therefore, complete network-wide key connectivity is ensured by using these three protocols.

6. Performance Analysis

6.1. Security Analysis

We have proposed an adaptive key management framework for WMSNs in this paper. From simple MPKM to complex TKM, they can provide modular security services from basic functionalities to advanced functionalities on kinds of nodes. For instance, they can establish both pairwise key and group key for sensor nodes with the same pre-distributed secrets and update their keys at the same time. Meanwhile, the secret key of the mesh nodes also can be established by extending the key management of the group keys. Furthermore, our framework satisfies the security requirements of both WSNs and WMNs. In this section, we analyze the security of our proposed key management framework.

Compromise resistance: In MPKM, since the paiwise key of each sensor node in our protocol is different from each other, the discovery of the captured keys cannot give out any knowledge of the keys in the innocent nodes. Moreover, any node's failure or compromise triggers a key updating process and, thus, the keys shared by the captured nodes and the innocent nodes are get rid of. That means MPKM provides sufficient security, no matter how many sensors in the same cluster are compromised MPKM can achieve perfect compromise resilience. In MPGM, the group key is established by the collaboration of all the nodes in the same cluster, no participant can predetermine the common key and each node has the same right to verify if its contribution is included in their currently used group key. Also, the group key can be updated periodically. Thus, MPGM has perfect group fairness and compromise resilience. In TKM, any k nodes can jointly perform key discovery whereas $(k - 1)$ nodes cannot derive any information even after collusion. Thus, it is k secure and has $(k - 1)$ resistance. Also, the secret key can be updated when the group keys changed or a new cluster wants to join the WMNs.

Group confidentiality: In MPGM, nodes that are not the part of the group should not have access to any key

that can decrypt any data broadcast to the group. For the message, only the powerful node (such as BS or TS) can decrypt the message to get the contributions of the client nodes by using its pairwise key with the client nodes. With the same reason, only the contributory nodes can recover the group key by using their own contribution. Therefore, the group key is group confidential.

6.2. Comparison with other Protocols

Now let us compare our protocols with the available key management techniques for wireless sensor networks and wireless mesh networks. Such as Du's [12] key Matrix, Camtepe's [9] Combinatorial Design, Liu's [13] polynomial based protocols for WSN, ARSA [7], SeGrOM [8] and LSSS [6] for secure WMN group communication.

For convenience, the following notations are used to analyze the various properties: Y: Yes or has; N: No or hasn't; Inc.: will increase; Nor.: remain normal; λ**:** the threshold, represents that if more than λ nodes be compromised, the protocol is not secure; little p: the protocol only provides p connectivity; big P: the protocol provides 100% or strong key connectivity. **Table 2** lists the comparisons among our protocols and protocols only for the WSN or WMNs.

We consider the performance comparisons in terms of the scalability，compromise resistance，key connectivity，nesh nodes security，modular security service. It is obvious that only our protocols can be scalable when the network size changes. If some nodes are compromised by the adversary, protocols for WSN and WMN only have a certain level of resistance, if the number of compromised nodes exceed the level, the network will not secure any more. While our protocols can maintain 2P+λ resistance, which mean MPKM and MPGM can maintain perfect compromise resistance, only TKM has a threshold λ**.** In terms of nodes security, only our protocols can provide security protection for both sensor nodes and mesh nodes and the storage for nodes will not increase no matter other two protocols being included or not. Thus, from the table we can see only the protocols proposed in our framework can be adaptive for WMSNs.

7. Conclusions

The interconnection of heterogeneous WSN and WMN networks is a pilot case which can be used to derive directions for the research on future heterogeneous network architectures. One of the major challenges for the development of future network architectures is the creation of adaptive key management protocols for the diversity of network nodes. In this paper we have presented an adaptive key management framework for WMSNs. It includes three possible keying implementations for different network nodes, *i.e.* pairwise key for sensor nodes, group key for high and low level sensor nodes and secret key shares through threshold cryptography for mesh nodes. The results clearly show that they can adapt to the different resource availability and achieve levels of security. In short, no matter the addition of low end nodes or high end devices to a secure environment might introduce security risks. The design of our key management framework can give simple but effective security solutions for each level of devices. And as we know, our framework is the first one which provides adaptive and modular security service for WMSNs. This is extremely important for the future generation of integrated networks.

Adaptive security is an interesting future research track. It will remain a continuous challenge to integrate low-end devices in future networking environments. In our future work, we would like to investigate new keying mechanism with better extension properties to provide perfect security and robust continuity for future generation of integrated networks.

8. Acknowledgements

This work is supported by the National Natural Science Foundation of China under Grant No.60903188, 6086 30 01 and 60903189, the Innovation Program of Shanghai Municipal Education Commission under Grant No.10YZ 157, the Shanghai university scientific selection and cul-

Table 2. Comparisons among our protocols and protocols only for the WSN or WMNs

	Du's	Liu's	Camtepe's	ARSA	SeGrOM	LSSS	Our's
Scalability	N	N	N	N	N	N	Y
Compromise resistance	λ	*p*	*p*	*p*	*p*	*p*	2P+λ
Key connectivity	*p*	*p*	*p*	P	P	P	P
Node's Storage overhead	In.	Inc.	Inc.	Nor.	Inc.	Inc.	Nor.
Sensor security	Y	Y	Y	N	N	N	Y
Mesh nodes security	N	N	N	Y	Y	Y	Y
Modular security service	N	N	N	N	N	N	Y

tivation for outstanding young teachers in special fund No.sdl09010.

9. References

[1] S. Bouckaert, E. D. Poorter, B. L. J. Hoebeke, I. Moerman and P. Demeester, "Strategies and Challenges for Interconnecting Wireless Mesh and Wireless Sensor Networks," *Wireless Personal Communications*, Vol. 53, No. 3, 2010, pp. 443-463.

[2] J. Ishmael and N. Race, "Wireless Mesh Networks (Handbook)," Chapter 7, Lancaster University, pp. 149-166.

[3] M. Wen, L. Dong, Y. F. Zheng and K. F. Chen, "Towards Provable Security for Data Transmission Protocols in Sensor Network," *Journal of Information Science and Engineering*, Vol. 25, No. 1, 2009, pp. 319-333.

[4] S. Glass, M. Portmann and V. Muthukkumarasamy, "Securing Wireless Mesh Networks," *IEEE Internet Computing*, Vol. 12, No. 4, 2008, pp. 30-36.

[5] S. Avancha, J. Undercoffer, A. Joshi and J. Pinkston, "Security for Wireless Sensor Networks," *Wireless Sensor Networks*, Kluwer Academic Publishers, Norwell, 2004, pp. 253-275.

[6] F. H. Ching, H. C. Guo, C. Qi and J. Chen, "A Novel Linear Multi-Secret Sharing Protocol for Group Communication in Wireless Mesh Networks," *Journal of Network and Computer Applications*, Vol. 10, No. 16, 2010.

[7] Y. C. Zhang and Y. G. Fang, "ARSA: An Attack-Resilient Security Architecture for Multi-Hop Wireless Mesh Networks," *IEEE Journal on Selected Areas in Communications*, Vol. 24, No. 10, 2006, pp. 1916-1928.

[8] J. Dong, K. Ackermann and C. Nita-Rotaru, "Secure Group Communication in Wireless Mesh Networks," *Ad Hoc Networks*, Vol. 10, No. 16, 2009, pp. 1563-1576.

[9] S. A. Camtepe and B. Yene, "Key Distribution Mechanism for Wireless Sensor Networks," TR-05-07 Rensselaer Polytechnic Institute, Computer Science Department, March 2005.

[10] L. Eschenauer and V. D. Gligor, "A Key-Management Scheme for Distributed Sensor Networks," *Proceeding of the* 9*th ACM Conference on Computer and Communication Security*, Washington, DC, 2002, pp. 41-47.

[11] H. Chan, A. Perrig and D. Song, "Random Key Predistribution Schemes for Sensor Networks," *IEEE Symposium on Security and Privacy*, 2003, pp, 197-213.

[12] W. Du, J. Deng, Y. S. Han, P. K. Varshney, J. Katz and A. Khalili, "A Pairwise Key Predistribution Scheme for Wireless Sensor Networks," *ACM Transactions on Information and System Security*, Vol. 8, No. 1, 2005, pp. 228-258.

[13] D. Liu and P. Ning, "Establishing Pairwise Keys in Distributed Sensor Networks," *ACM Transactions on Information and System Security*, Vol. 8, No. 1, 2005, pp. 41-77.

[14] M. Wen, K. F. Chen, Y. F. Zheng and H. Li, "A Reliable Pairwise Key-Updating Scheme for Sensor Networks," *Journal of Software*, Vol. 18, No. 5, 2007, pp. 1232-1245.

[15] D. Liu, P. Ning and K. Sun, "Efficient Self-Healing Group Key Distribution with Revocation Capability," *Proceedings of the* 10*th ACM Conference on Computer and Communications Security*, Washington, DC, 2003, pp. 231-240.

[16] W. Zhang and G. Cao, "Group Rekeying for Filtering False Data in Sensor Networks: A Predistribution and Local Collaboration-Based Approach," *Proceedings from the Conference of the IEEE Communications Society*, 2005, pp. 503-514.

[17] M. Wen, J. S. Lei, Z. Tang, X. X. Tian, K. F. Chen and W.D. Qiu, "A Verified Group Key Agreement Protocol for Resource-Constrained Sensor Networks," *Lecture Notes in Computer Science*, Vol. 5854, 2009, pp. 413-425.

[18] G. Gaubatz, J. P. Kaps and B. Sunar, "Public Key Cryptography in Sensor Networks Revisited," *Proceedings of the* 1*st European Workshop on Security in Ad-Hoc and Sensor Networks*, Springer, 2004, pp. 2-18.

[19] S. Zhu, C. Yao, D. Liu, S. Setia and S. Jajodia, "Efficient Security Mechanisms for Overlay Multicast Based Content Delivery," *Computer Communications*, Vol. 30, No. 4, February 2007, pp. 793-806.

[20] S. Zhu, S. Setia, S. Xu and S. Jajodia, "GKMPAN: An Efficient Group Rekeying Scheme for Secure Multicast in Ad-Hoc Networks," *Mobiquitous*, Vol. 00, 2004, pp. 42-51.

[21] R. Balachandran, B. Ramamurthy, X. Zou and N. Vinodchandran, "CRTDH: An Efficient Key Agreement Scheme for Secure Group Communications in Wireless ad Hoc Networks," *Proceedings of IEEE International Conference on Communications*, Vol. 2, 2005, pp. 1123-1127.

[22] A. Shamir, "How to Share a Secret," *Communication ACM*, Vol. 22, No.11, 1979, pp. 612-613.

Wireless Sensor Network, **2010, 2, 698-702**
doi:10.4236/wsn.2010.29084 Published Online September 2010 (http://www.SciRP.org/journal/wsn)

Sensors Dynamic Energy Management in WSN

Xianghui Fan, Shining Li, Zhigang Li, Jingyuan Li
College of Computer Science, Northwestern Polytechnical University, Xi'an, China
E-mail: *fxianghui@vip.qq.com*
Received July 2, 2010; *revised August* 16, 2010; *accepted September* 12, 2010

Abstract

A wireless sensor node is typically battery operated and energy constrained. Therefore, it is apparent that optimal energy management is one of the most important challenges in WSN development. However, energy management requires in-depth knowledge and detailed insight concerning specific scenarios. After Carrying out a large number of experiments in precision agriculture, we find that it is the sensors that have never been concerned consuming the most energy of the node. In order to conserve energy and prolong the lifetime of WSN, we design and carry out a dynamic energy management strategy for sensors. The basic idea is to shut down all sensors' power when not needed and wake them up when necessary. Valuable conclusions are extracted and analyzed.

Keywords: WSN (Wireless Sensor Network), Precision Agriculture, Dynamic Energy Management, TinyOS

1. Introduction

A large number of intelligent micro-sensor nodes with sensing, processing and wireless communicating capabilities form wireless sensor network (WSN), which completes complicated tasks in some specific field, such as precision agriculture. Compared with old methods, WSN has significantly drawn extensive attention. It does not rely on fixed infrastructure and has many characteristics such as fast setup, strong survivability and so on [1]. It has been considered as a good scheme to conduct precision agriculture data collection and processing. In 2002, Intel has a project looking at how WSN can be used to improve grape production. They worked with agricultural scientists on a long-term deployment of WSN in a wine grape vineyard. By densely monitoring and analyzing they found the relationship between grape quality and climatic conditions. It has been proved that WSN could play a role in precision agriculture.

Just the same as other applications, energy constraint of sensor nodes is the major problem for precision agriculture. Data aggregation [2] and low power listening [3] algorithms are effective method to reduce energy consumption in normal wireless sensor networks. However, after a sufficient number of experiments we found that energy consumption in precision agriculture has some special issues. Generally speaking, in order to monitor the growth conditions of crops, one node has to connect with many sensors, such as Co2 sensor, air temperature sensor, air humidity sensor, light sensor, soil temperature/moisture sensor and so on. Although the sensors consume a large portion of the energy, we never pay any attention to this issue in our previous research. It is necessary to reduce the energy consumption of sensors. We design and carry out a sensor dynamic energy management (SDEM) to reduce energy consumption of sensors and extend network lifetime. The basic idea is to shut down sensors when not needed and wake them up when necessary [4,5]. The experimental results indicate that SDEM is an effective technique in reducing node energy consumption without significantly degrading performance.

The remainder of this paper is organized as follows. Section 2 gives the energy consumption of all parts of the sensors in precision agriculture. And we get a conclusion that the sensors consume most of the energy. The architecture of the SDEM is described in Section 3, and Section 4 reports the hardware and software design and some considerations about implementation. In Section 5, the actual deployment is described in detail, and finally Section 6 reports our conclusion and gives some directions on the ongoing work.

2. Sensors Energy Consumption Analysis

A sensor node has several major components: processor, memory, A/D converter, sensing unit and radio. Each node sleep mode corresponds to a particular combination

of component power modes. In general, if there are N components labeled (1, 2, … , N) each with K_i sleep modes, the total number of node sleep modes is $\sum_{i=1}^{n} k_i$ [6]. However, from a practical point of view not all sleep modes are useful. Optimizing the key issue could achieve a noticed effect. In our case, the NPUnode used in precision agriculture consists of a processor Atmel2561, which has rich resource such as 256K bytes in-system programmable flash, 4K bytes EEPROM, 8K bytes SRAM and so on, a RF230 chip acts as the radio unit, an AT45DB041B chip acts as an extern memory. And the NPUnode has been equipped with six sensors, just as showing in **Figure 1**. Each component is controlled by the micro-operating system (TinyOS). **Table 1** enumerates the characteristics of all devices of the NPUnode.

Sensor transducers translate physical phenomena to electrical signals, and are classified as either analog or digital devices depending on the type of their output. There are several sources of power consumption in a sensor, including 1) signal sampling and conversion of physical signals to electrical ones, 2) signal conditioning, and 3) analog to digital conversion. Given the diversity of sensors there is no typical power consumption number. In general, however, passive sensors such as temperature sensors etc., consume negligible power relative to other components of sensors, array sensors such as Co2 sensors can be large consumers of power. As shown in **Table 1**, the current value of Co2 sensor is 30mA, sometimes even reaches the max value 150mA. And other sensors also have power dissipation.

In accordance with **Table 1**, we draw a figure of energy consumption of major components. Just as shown in **Figure 2**, the sensors energy consumption is far more than other components. According to electric power formula $W = Pt = UIt$, we know that reducing the running time is the only way to decrease energy consumption while not change the sensors characters. T is the data collection period, while T_{woking} is the working time of sensors and T_{idle} means the period that sensors is in the idle state. And the relationship between them is $T = T_{working} + T_{idle}$ and $T_{idle} >> T_{working}$. Data collection period is a long time in agriculture application. For example, the nodes we have deployed in greenhouse in YangLing only need to report the sensor data once an hour. This means that the sensors only need to work for several minutes per hour, so they should be turned off when they are idle to save energy.

3. Sensors Dynamic Energy Management

Nodes energy consumption is application specific. In precision agriculture the sensors consume the greatest portion of the node energy. The reason is that one node is always connected with several sensors to monitor plant growth environment. However, the most time of sensors is in idle state and waste a lot of energy. For the purpose of saving energy of nodes and extending the life of WSN, we design the Sensors Dynamic Energy Management (SDEM) strategy which turns on the sensors when the node receives an acquisition command from root and turns it off when the sensors are in the idle state. The principle of SDEM is showed as **Figure 3**.

3.1. Hardware Design

SDEM needs independent hardware design supporting. The NPUnode supports power with DC 5 V or 3.6 V Nickel-Hydrogen Battery. And the sensors used to monitor the plant growth environment need different voltage. A power supply of DC3.0 V is designed for light sensor, soil temperature sensor, temperature and humidity sensor. Co2 sensor and soil Moisture sensor power are supplied with DC5.0 V. The hardware design is described as in **Figure 4.** XC6221B and TPS61202 chips which are connected with ATmega2561 I/O pins translate battery power to DC3.0 V and DC5.0 V to meet above

Figure 1. The NPUnode and equipped sensors.

Table 1.The devices of the node and their characteristics.

Device	Type	Characteristics
Processor	ATmega 2561 V	256 K Bytes Flash, 4 K Bytes EEPROM, 8 K Bytes Internal SRAM, er :2.7~5.5 V,10 mA
Radio	RF230	Power: Voltage: 1.8 V~3.6 V, SLEEP: 0.1 μA, RX: 16 mA, TX: 17 mA; Fast Power-Up Time < 1 ms
Flash	AT45DB041	Power Supply: 2.7 V~3.6 V, 4 mA Active Read
Temp and Humi Sensor	SHT11	Temperature range: −40℃ to +123.8℃; Humidity range: 0 to 100% RH; Power : 3 V, 0.5 mA
Light Sensor	ISL29002	Range: 10,000 lux~100,000 lux; Power : 2.5 V to 3.3 V, <10 mA
Soil Temp Sensor	PT1000	Range: −70°C to + 500°C; Power:3VDC, 3 mA; Response time < 10 s
Soil Mois Sensor	TDR-3	Range: 0～100%（m3/m3）; Power Supply: 4.5 ~ 5.5V DC, 50～70 mA; Response time: < 1 s
Co2 Sensor	GE/Telaire 6004	Range: 0-2000 ppm; Power: 4.3 VDC ~7.0 VDC, 30~150 mA; Response time: < 2 min

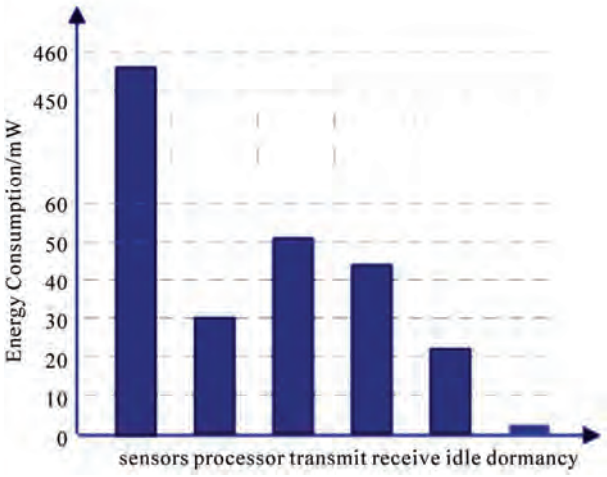


Figure 2. Energy consumption of node in precision agriculture.

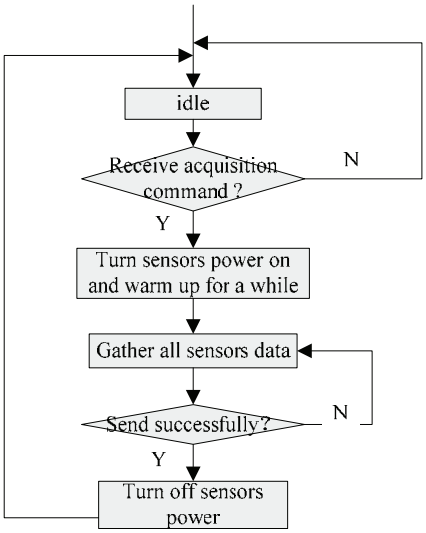


Figure 3. Principle of sensor dynamic energy management.

demand. Inputting L level signal for I/O pins we can provide power for sensors. On the other hand, we can turn off the sensors power supply by inputting H level signal for I/O pins. It is easy to implement in TinyOS.

3.2. Software Design

The software system considered here is based on TinyOS 2.1 [7] which is an embedded operating system especially for WSN applications. The extreme power limitation of nodes forces the operating system to take very different approaches than traditional computing classes. TinyOS is implemented using NesC programming language—a dialect of C programming language. It integrates with model of components/module and events driven model. Over the past few years, TinyOS has grown from a small research project to dominant operating system for low power wireless sensor networks.

At a high level, TinyOS provides three things to make writing systems and applications easier [8]. 1) a component model, which defines how you write small, reusable pieces of code and compose them into larger abstractions; 2) a concurrent execution model, which defines how components interleave their computations as well as how interrupt and non-interrupt code interact; 3) application programming interfaces (APIs), services, components libraries and an overall component structure that simplify writing new applications and services. The HplAtm256-GenerallIOC is the most important components to design the SDEM. It exposes the ATmega256's 53 digital I/O pins as 53 GeneralIO interfaces, hiding the slightly different instruction sequences needed to perform some operations on some I/O pins. Some I/O pins can be set atomically in a single assembly instruction. On the purpose of turn on/off the sensors power, we need to use the interface GeneralIO to enable/disenable the pins of Atmel2561 which connect with XC6221B and TPS61202 by inputting Low/High level signal. The GeneralIO interface offers commands to configure, read and write a typical microcontroller digital I/O pin. Firstly, after the OS initializes all needed components and booted successfully, we call GeneralIO.makeoutput() to make the pins as a controller. Then the NPUnode enters a ready

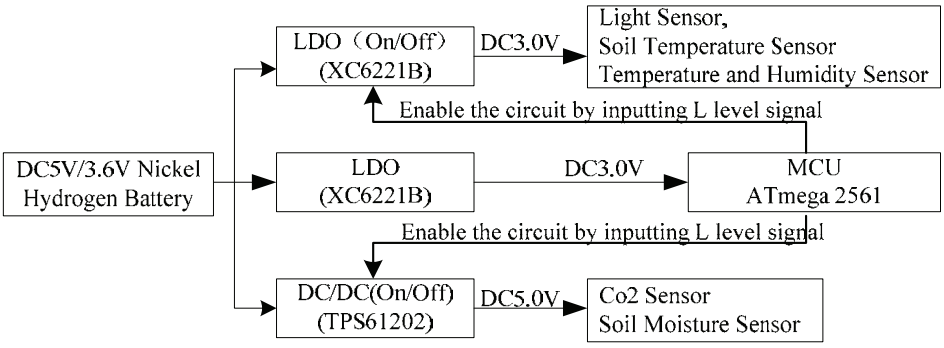


Figure 4. The hardware design of SDEM.

state waiting for the command from root. Once the node receives the sampling command, it then calls the command GeneralIO.set() to turn on the sensors power and waits for 2 minutes for warming up the sensors. After reading all sensors data and sending it back successfully, nodes calls GeneralIO.clr() to turn off the sensors power. The **Figure 5** and **Figure 6** show the implementation of SDEM.

```
module AgriMonitorC{
   uses {
        interface Boot;
        interface Read<uint16_t> as SensorReader;
        interface GeneralIO as PowerControl;
     }
}
implementation {
   event void Boot.booted(){
      ...
     PowerControl.makeOutput();
     PowerControl.clr();
     ...
   }
   event void SettingsValue.changed(){
     ...
     if( receive read sensor command ){
       call PowerControl.set(); //turn on the sensor power
       //wait for a while to warm up the sensors
       SensorReader.read();
       …        //read other sensors data
     }
   }
   event SensorReader.readDone(error_t err, uint16_t val){
     …
     if(get all sensors data)
         call sensorDataSend.send(&msg, len);
   }
   event sensorDataSend.sendDone(message_t* m, error_t err){
     if(err == SUCCESS){
       call PowerControl.clr();//power off sensor power
      }
      else{
        errorControl();
      }
    }
}
```

Figure 5.The module implementation of SDEM.

```
configuration AgriMonitorAppC{}

implementation{
  Components AgriMonitorC as App;
  Components MainC;
  App.Boot -> MainC;

   Components HplAtm256GeneralIOC;
  App.PowerControl -> HplAtm256GeneralIOC.PortC0;
   ...
}
```

Figure 6.The configuration wired the needed components.

4. Experiment Results

We deployed NPUnodes in field crop production in YangLing, LuoChuan, AnSai in ShaanXi province, just as showing in **Figure 7**. The NPUnodes are equipped with Co2 sensors, temperature and humidity sensors, light sensors, soil temperature sensors and soil moisture sensors to collect important data for plant growth environment. The collected data is gathered at the edge of the field by a field gateway and further transferred via GPRS to a PC server for data analysis. Once something odd happened the server will send message via MMS to farmers. In addition to the agronomic experiment, we expect to gather data and statistics on the behavior of NPUnodes in real-world experiment. For energy-efficiency considerations, we use SDEM strategy in NPUnodes and they reported data only once per hour. To get the exact value of sensors, we need to wait sensors warm-up for two minutes. The rest of the time we turn off the sensor power to save energy.

From collected voltage data of NPUnodes, we make a comparison between the SDEM strategy and before methods. Just showing in **Figure 8**, we can give the conclusion that the SDEM strategy has significantly saved energy. After making in-depth research we find that the

Figure 7. NPUnode deployed in greenhouses with six sensors.

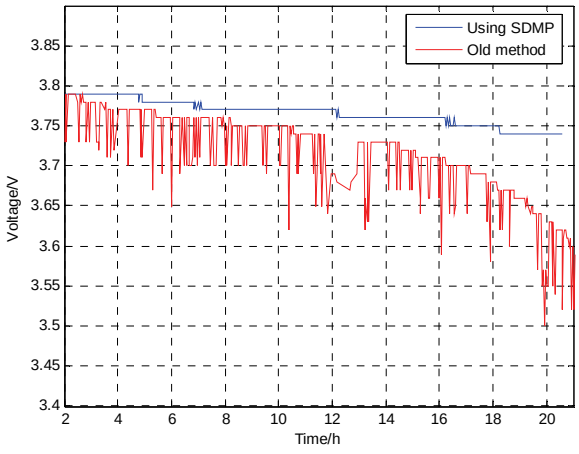


Figure 8. Lift time of sensors between SDEM and before method.

SDEM strategy almost has extended the NPUnodes life time from 1 month to 3 months. However, SDEM should save sensors energy 30 times than not using SDEM nodes, theoretically on the condition that NPUnode reports sensors data once per hour (all sensors only turn on for 2 minutes). The reason is that not only the sensors consume the energy, other components such as radio, processor etc. also consume the energy.

5. Conclusions and On-Going Work

In precision agriculture, nodes are always equipped with several sensors which consume a large portion of nodes energy, especially active sensors. In order to save energy and extend the lifetime of nodes, we design and implement the SDEM strategy based on our NPUnodes. The basic idea is to shut down devices when not needed and wake them up when necessary. And then we deployed our nodes in greenhouse. From the voltage data collected we give the conclusion that the SDEM strategy can significantly save energy of nodes, and extend the life time of nodes from one month to three month.

The nodes need continuous work for one year or more in precision agriculture. So the nodes need an unfailing supply of energy. Next time, we will make a serial research and supply solar power for NPUnodes. Maybe it is a way to solve the energy problem finally.

6. References

[1] S. M. Xiong, L. M. Wang, X. Q. Qu and Y. Z. Zhan, "Application Research of WSN in Precise Agriculture Irrigation," *International Conference on Environmental Science and Information Application Technology*, Wuhan, July 2009, pp. 297-330.

[2] B. Krishnamachari, D. Estrin and S. Wicker, "Impact of Data Aggregation in Wireless Sensor Networks," DEBS'02.

[3] J. Polastre, J. Hill and D. Culler, "Versatile Low Power Media Access for Wireless Sensor Networks," *Proceedings of the* 2*nd International Conference on Embedded Networked Sensor Systems*, Maryland, November 2004.

[4] A. C. Sinha, "Dynamic Power Management in Wireless Sensor Networks, Design & Test of Computers," *Proceedings of IEEE*, Vol. 18, No. 2, 2001, pp. 62-74.

[5] R. C. Luo, L. C. Tu and O. Chen, "An Efficient Dynamic Power Management Policy on Sensor Network," *Proceedings of the* 19*th International Conference on Advanced Information Networking and Applications*, 2005, pp. 341-344.

[6] C. Lin, N. Xiong, J. H. Park and T.-H. Kim, "Dynamic Power Management in New Architecture of Wireless Sensor Networks," *International Journal of Communication Systems*, Vol. 22, No. 6, 2008, pp. 671-693.

[7] TinyOS Tutorials, http://docs.tinyos.net/index.php /Tiny OS _Tutorials

[8] P. Levis and D. Gay, "TinyOS Programming," Cambridge University Press, England, 2009, pp. 6-7.

Wireless Sensor Network, **2010, 2, 703-709**
doi:10.4236/wsn.2010.29085 Published Online September 2010 (http://www.SciRP.org/journal/wsn)

Design of Building Monitoring Systems Based on Wireless Sensor Networks*

Qifen Dong, Li Yu, Huanjia Lu, Zhen Hong, Yourong Chen
College of Information Engineering, Zhejiang University of Technology, Hangzhou, China
E-mail: *dongqifen*0419@163.*com, lyu@zjut.edu.cn,* {*hongzhen*614, *luhuanjia*1020}@126.*com, jack_chenry*@163.*com*
Received June 9, 2010; *revised July* 11, 2010; *accepted August* 19, 2010

Abstract

Wireless Sensor Network provides a potential technique for monitoring the indoor environment. This paper presents a Building Monitoring system based on Wireless Sensor Networks. A clustering-based network specified for building monitoring is proposed, which is inspired by LEACH (Low Energy Adaptive Cluster Hierarchy) method. Further, two key ideas are used to implement the clustering-based network. First, the configuration module of building management software is used to conduct all nodes in a room forming a local cluster. This cluster formation method does not consume node energy. Second, because cluster-heads cannot directly transmit packets to the sink node due to limited wireless communication range, the cluster-head communications are represented by a multi-hop tree rooted at the sink node. The experiment has been made to demonstrate the feasibility of the proposed results.

Keywords: Wireless Sensor Networks, Building Monitoring, LEACH, Cluster

1. Introduction

According to the 2007 UN Environment Program [1], about 40 percent of total energy is used for heating, cooling, lighting and ventilation of buildings. Substantial savings can be made by applying these functions only when and where they are needed. Such control is only possible when indoor conditions such as temperature, relative humidity or light are measured. Wired sensors could certainly be installed at sensing locations, but such a step requires significant effort and an additional set of wires throughout a building. Estimates of the cost to deploy wires ranged from $2.2 per meter for new buildings to $7.19 per meter for existing constructions in 2002 [2]. In addition, running wires in existing constructions induces other problems, such as destroying appearance of buildings. Transmitting sensing data wirelessly provides a significant benefit for monitoring indoor environment. However, traditional wireless systems suffer from their own disadvantages, such as high running cost, and high energy consumption of monitoring devices.

Wireless Sensor Network (WSN) [3] which consists of dense sensor nodes that continuously observe physical phenomenon provides an opportunity for building monitoring [4,5]. Thomas Schmid [6] reported their experience with the implementation, deployment and operation of SensorScope, an indoor environmental monitoring network based on WSNs. Literature [7] demonstrated an industrial-strength wireless sensor network application for indoor environment monitoring. This application is integrated WSNs with a Building Management System. Won-Suk Jang showed how advanced WSN technologies can be used to monitor conditions in and around buildings [8]. A WSN system was deployed in a number of residential and commercial buildings in [9]. W.S. Jang and W. M. Healy investigated WSN performance metric for building monitoring applications [10]. It appears that WSN provides huge potential for building monitoring. However, it still needs getting more attention continuously.

The main contributions of this paper are as follows: 1) a Building Monitoring system based on WSNs (BMW SNs) is presented; 2) inspired by LEACH (Low Energy Adaptive Cluster Hierarchy) method [11,12], a clustering-based network specified for building monitoring is proposed; 3) the configuration module of building management software is used to conduct all nodes in a room forming a local cluster. This cluster formation method does not consume node energy; 4) because cluster-heads cannot directly transmit packets to the sink node due to limited wireless communication range, the cluster-head

*Supported by the Key Project of Zhejiang Provincial Ministry of Education under Grant NO.ZD2007003, and the Zhejiang Provincial Natural Science Foundation of China under Grant NO.Y1080163.

communications are represented by a multi- hop tree rooted at the sink node.

The remainder of the paper is organized as follows. Section 2 describes the overview of BMWSNs. The implementation of the clustering-based network is given in Section 3. Section 4 shows experimental results. Conclusion is given in Section 5.

2. System Overview

Figure 1 shows the architecture of BMWSNs. Several sensor nodes are placed in each room. Because the data, such as temperature, humidity and light intensity, sensed by nodes in the same room are highly correlated, a node termed cluster-head is installed in each room. The cluster-head receives data from other nodes in the same room, performs data aggregation, and forwards data to a sink node which is connected to a computer through a RS-232 connection. The building supervisor can get the indoor environment with the help of building management software installed on the computer. So it is possible to control the electro-devices.

Figure 2 shows hardware images. The sensor node powered by 2 AA batteries consists of a main board and a sensor board. The main board contains a microprocessor (ATMEGAL128L) and a radio (CC2420). The sensor board contains temperature/ humidity sensor (SHT11), light sensor (STL2550), and human detection sensor (BISS0001), and is plugged into the main board via a 41-pin connector. The sink node is composed of the main board and a bottom board which integrated UART connector, power outlet, and program interface. At the computer, the building management software is written in Qt, using Sqlite as the database.

The BMWSNs has the following features from the view of end-user.

1) Displaying curve: it displays curves of temperature, humidity, light intensity, and human detection of the selected room.

2) Configuring node's attributes: the node's attributes can be configured by plugging the node into the bottom board. These attributes include long address, room number, normal/abnormal measurement period for each sensor, as well as alarm threshold for each sensor. This configuration module is useful in cluster formation.

3) Tipping alarm information: the alarm information is flipped automatically when the measurement exceeds the corresponding alarm threshold.

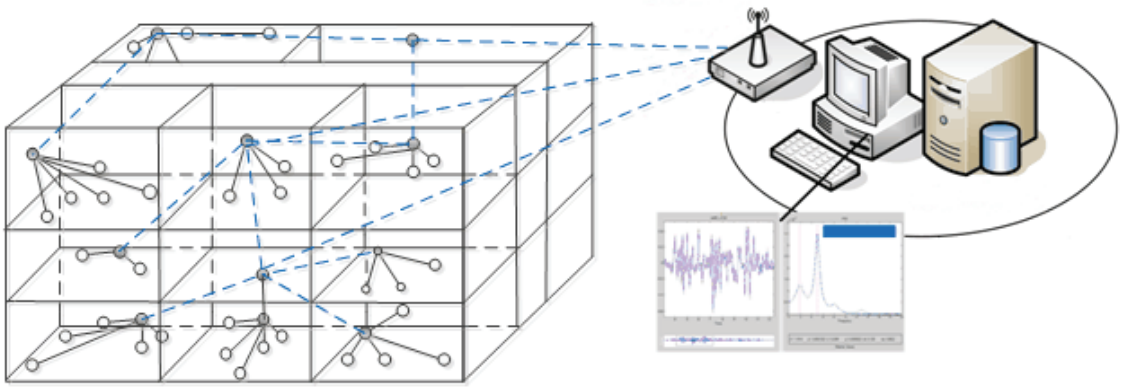
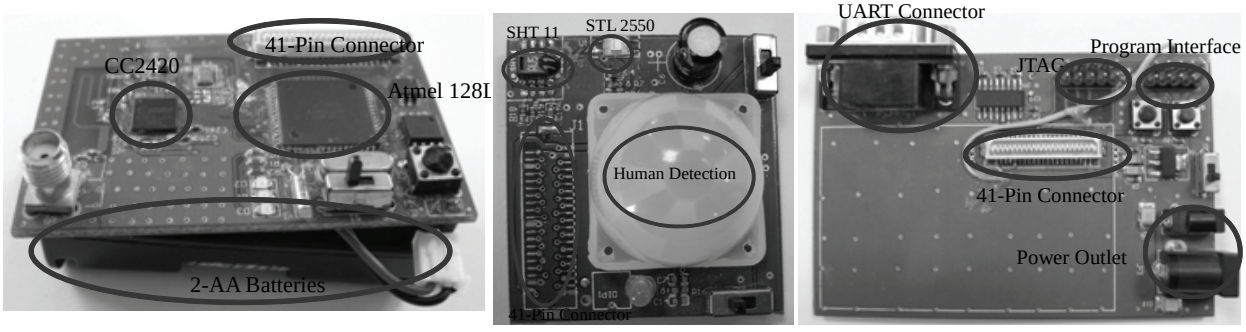


Figure 1. The architecture of WBMSNs.



(a) main board **(b) sensor board** **(c) bottom board**

Figure 2. The sensor node consists of (a) and (b); the sink node is composed of (a) and (c).

4) Storing data: the historical data is stored in the database. It can provide a reference for investigating power saving scheme.

3. Implementation of the Clustering-Based Network

The periodic listen and sleep mechanism in S-MAC [13] protocol is adopted to save node energy. This section focuses on how sensor nodes process their original sensing signals and transmit data to the sink node. This is a hot spot and difficult problem in the study of WSNs. It relates to data aggregation, routing, topology control and so on. Moreover, it should be oriented to the specified application.

As mentioned previously, there is strong correlation among the data from nodes in the same room, so we constructed a clustering-based network inspired by LEACH method. All nodes in a room form a local cluster, with one node acting as the cluster-head. All non-cluster-head nodes must transmit their data to the cluster-head.

In LEACH, it is assumed that all cluster-heads could communicate with the sink node directly. However, the wireless communication distance is limited, so this assumption is impracticable. Thus it is necessary to design a simple and feasible protocol for transmitting data from a cluster-head to the sink node.

3.1. Cluster Formation

In LEACH, nodes elect themselves to be cluster-heads with a certain probability. Due to space constraints, the details of the cluster formation in LEACH are omitted, interested readers should refer to [11,12]. However, this algorithm is not suitable for building monitoring application. There are two reasons for this. First, performing this process is difficult and power-wasting in actual application. Second, it is assumed that the data from all nodes is highly correlated. In fact, the correlation only exists among nodes in the same room. Therefore, it would be better to cause nodes in the same room to form a local cluster using a simple method. To do so, the configuration module of the building management software is used. The node's attributes need to be configured before the node is placed. Using the room number, the node's short address is calculated according to Equation (1).

$$A_{node} = F \times F_m + R \times R_m + n \tag{1}$$

where F and R denote the floor and the room of this floor, respectively. For example, if the configured room number is 324, then $F = 3, R = 24$. F_m and R_m are the maximum number of nodes that are allowed in each floor and in each room, respectively. The two values are determined by the designer. n denotes the sequence that the node is deployed in a room.

The node which is first placed in a room is treated as the cluster-head automatically. So the cluster-head's short address is known to the other nodes in the same room. Let all nodes in a room form a local cluster. Obviously, this cluster formation mechanism does not require node energy expenditure, and the relationship between the node and the room is also established. Besides, it is easy to add or remove a node as long as the number of deployed nodes in a room is less than R_m.

Each node starts to measure temperature, humidity, light intensity, and human detection periodically after deployment. The measurements are packed and sent to the cluster-head every twelve hours. Then the data aggregation is performed in the cluster-head by Equation (2).

$$result = \begin{cases} v_1 \| v_2 \| \cdots v_i \| \cdots \| v_N & \text{if for human detection sensor} \\ \dfrac{\sum_{i=1}^{N} v_i - (v_{\min} + v_{\max})}{N-2} & \text{otherwise} \end{cases} \tag{2}$$

where v_i is the sensor measurement value of node i for a certain period, N is the total number of nodes in the local cluster, and $v_{\min} = \min_{i=1,\cdots N} v_i$, $v_{\max} = \max_{i=1,\cdots N} v_i$. Exceptionally, for human detection sensor, v_i is equal to one if node i has detected the human otherwise v_i is equal to zero. Performing data aggregation at the cluster-head decreases much amount of data which needs to be transmitted to the sink node. Because the energy used for computation is much less than the energy for communication, this aggregation process reduces the overall system energy expenditure.

A node speeds up sensor acquisition frequency once one of the sensor measurements exceeds the corresponding alarm threshold. A warning message is sent to the cluster-head as soon as the measurement exceeds the alarm threshold continuously for ten periods. The cluster-head forwards this warning message directly and discards the data from other cluster members. So the electro-devices can be regulated as quickly as possible.

Remarkably, the data aggregation makes sense only when all nodes in a room are synchronized in time. To do so, the cluster-head broadcasts a synchronization message periodically. The newly added node don not run the sensors until receiving the synchronization message.

3.2. Cluster-Heads Transmission Protocol

As mentioned above, the wireless communication range is limited, so cluster-heads need transmitting packets to the sink node via multi-hop manner. Denote the sink node as S_0, and the cluster-head communications are represented through a multi-hop tree rooted at S_0 (see **Figure 3**). There are two key sub-processes in implementing the multi-hop tree. The two sub-processes are

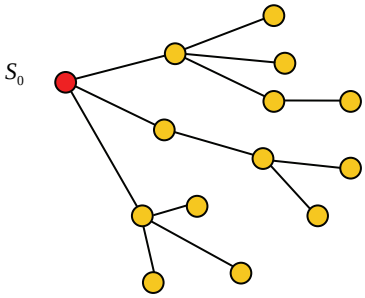


Figure 3. An example of multi-hop tree. The red circle denotes the sink node, while the yellow ones denote cluster-heads.

tree initialization and tree maintenance. The goal of the tree initialization is to initialize a tree with cluster-heads and the sink node. The tree maintenance is aimed at updating the tree structure to balance energy consumption among cluster-heads.

3.2.1. Tree Initialization

All cluster-heads are separated originally. The sink node forms a "network" and its depth in the network is zero. The aim of the tree initialization is to expand the network to construct a tree rooted at S_0. The initialization procedure is described as follows.

Step 1 A cluster-head that plans to join the network broadcasts a join-network request message. This cluster-head is termed CH. Meanwhile, a timer with fixed time period is scheduled. Then CH waits for response message during this time. The waiting time period should be long enough so that it allows multiple neighbors to response to the request.

Step 2 Neighbors of CH would receive the join-network request message. The neighbor sends out a join- network response message containing its depth in the network if it has joined the network, otherwise it discards the request.

Step 3 CH chooses the neighbor whose response message has the strongest received signal strength (RSS) as its candidate parent. The RSS is indicated by the received signal strength indicator in cc2420 chip. Then CH unicasts an association request message to the selected candidate parent.

Step 4 Upon reception of the association request message, the candidate parent responses it and adds CH to its children table.

Step 5 Receiving an association response message indicates that CH has joined the network successfully. CH stores information of the parent node and its depth in the network. CH's depth in the network is one more than the depth of parent node.

If CH cannot receive any join-network response message when the timer expires or it fails to request association, it tries to join the network again after a random delay.

Using this method, neighbors of the sink node join the network firstly. Then do the two-hop cluster-heads of the sink node. A tree rooted at S_0 is formed gradually.

3.2.2. Tree Maintenance

Each cluster-head can send packets to its parent after the tree is formed. However, the energy consumption among these cluster-heads is very different even they have the same depth in the tree, because each of them has a different number of children. Therefore, it is necessary to perform tree maintenance to balance the energy consumption.

To do so, each cluster-head updates its parent periodically, and the update period is computed randomly to avoid radio interference. The updating process is similar to the tree initialization sub-process, and the difference between them is that the join-network response message from node j contains more parameters about node j in the updating process. These parameters include the children number c_j, the residual energy $\overline{E}_j$, and the depth d_j in the tree. The combinations of them are considered in *Step* 3. Denote E as the initial energy of 2 AA batteries. A cluster-head i chooses the neighbor n as its parent with probability P_{in}, as calculated by Equation (3):

$$P_{in} = \frac{(1/c_n)^{\beta_1}(1/d_n)^{\beta_2}(\overline{E}_n/E)^{\beta_3}}{\sum_{j\in p_i}(1/c_j)^{\beta_1}(1/d_j)^{\beta_2}(\overline{E}_j/E)^{\beta_3}}, \quad n \in p_i \qquad (3)$$

where p_i is the set of cluster-heads that response to the join-network request from cluster-head i, and β_1, β_2, and β_3 are nonnegative weighting factors for each item.

4. Experiments

The BMWSNs is operated in buildings, and the material quality of the buildings, environmental condition, and so on can affect the distance and quality of the wireless communication. **Figure 4** shows the packet loss rate versus the wireless communication distance between two nodes which are separated by a wall. It indicates that the packet loss rate is lower than 20%. The packet is retransmitted if it is lost, then the packet loss rate is less than 4%. This demonstrates that deploying a cluster-head in each room can guarantee the network connectivity if all rooms are smaller than the size of 15 × 15.

The BMWSNs is tested with 21 nodes deployed randomly in several rooms. The node's attributes are configured using the configuration module of the building management software before the node is deployed. The actual deployment is shown as **Figure 5**. The sink node is deployed in room 323. Each room has a cluster-head and several sensor nodes. The figure between brackets

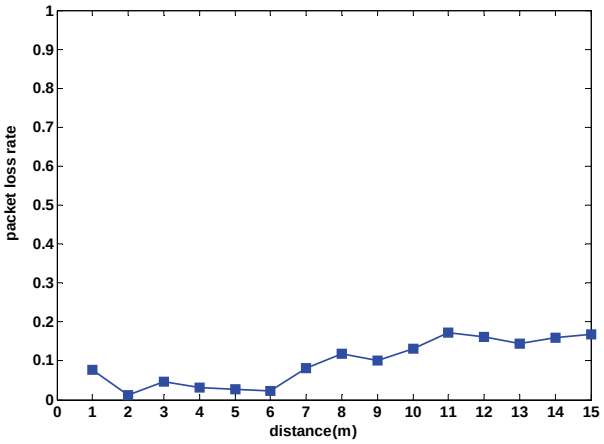


Figure 4. The packet loss rate VS. the distance.

denotes the node's short address. In the test, the maximal number of nodes that are allowed in each floor and each room are 100 and 10, respectively.

Figure 6 is a network topology graph as the BMWSNs starts up. The dotted lines with arrow denote communication links. It demonstrates the feasibility of the proposed clustering-based network. All nodes in a room form a star network centered on the cluster-head and all the cluster-heads build a multi-hop tree rooted at the sink node. Along this tree, all packets can be delivered to the sink node. For example, the route from nodes in room 322 to the sink node is as follows:

$$NS322 \rightarrow CH322 \rightarrow CH323 \rightarrow SINK$$

where NSn and CHn denote the sensor nodes and the cluster-head in room n, respectively.

Figure 7 shows the temperature curve of room 323. It also can display curves of other parameters by clicking on the icon at upper-right corner. This provides reference for controlling electrical devices in real time.

We regulate the temperature of room 322 at 16˚C (The upper and lower thresholds of temperature are 18˚C and 26˚C, respectively), then the alarm information is flipped automatically (see **Figure 8**).

Figure 9 shows historical temperature analysis chart. It provides a basis for building energy-saving plan.

5. Conclusions

The system architecture of BMWSNs is presented and an overview of system features is given. This paper focuses on the two key methodologies which are used to implement the clustering-based network specified for building monitoring. The experiment was made to demonstrate the feasibility of the BMWSNs. However, there exist shortages inevitably as a result of exploratory research on applying WSNs to building monitoring. We will perform further tests to find problems, and then improve it from both hardware and software. Future plans for the BMWSNs include:

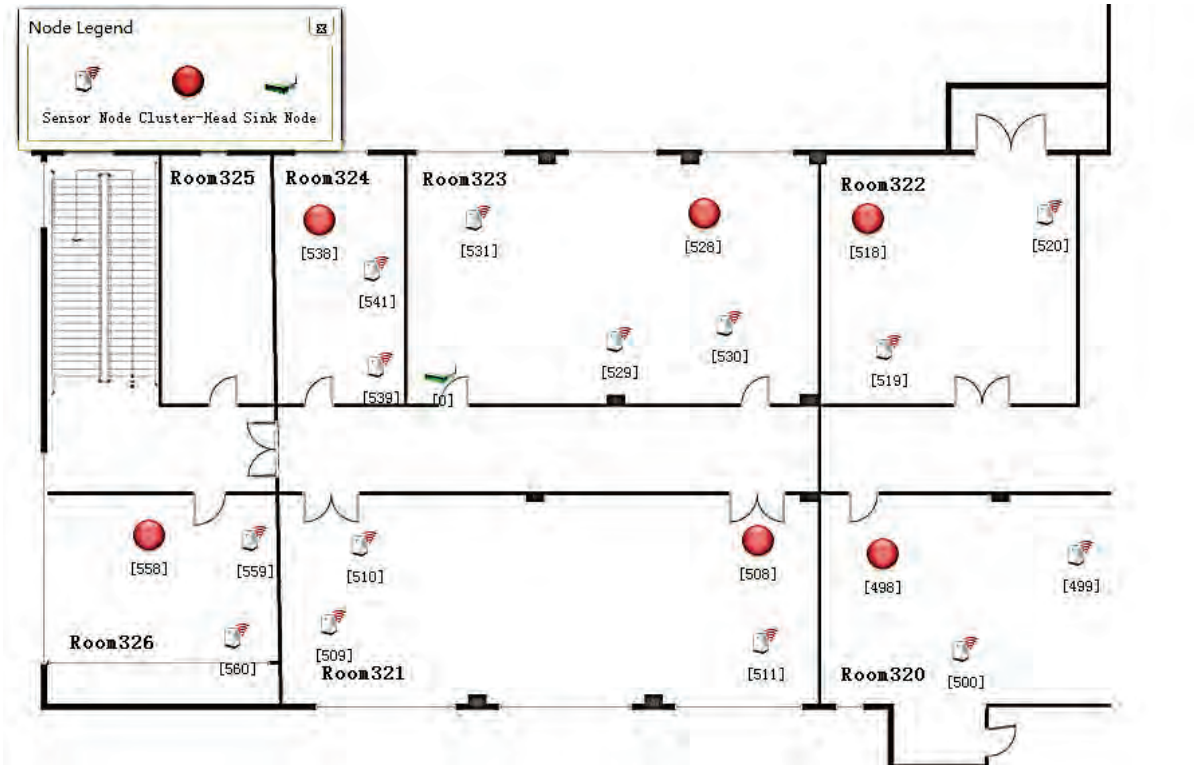


Figure 5. The actual deployment.

1) Usually, electric devices are regulated only when the data exceeds the alarm threshold. Therefore, nodes could store normal data locally in order to save communication energy consumption, and the user queries data depending on the need. Thus nodes deployed in the building construct a distributed storage and query database.

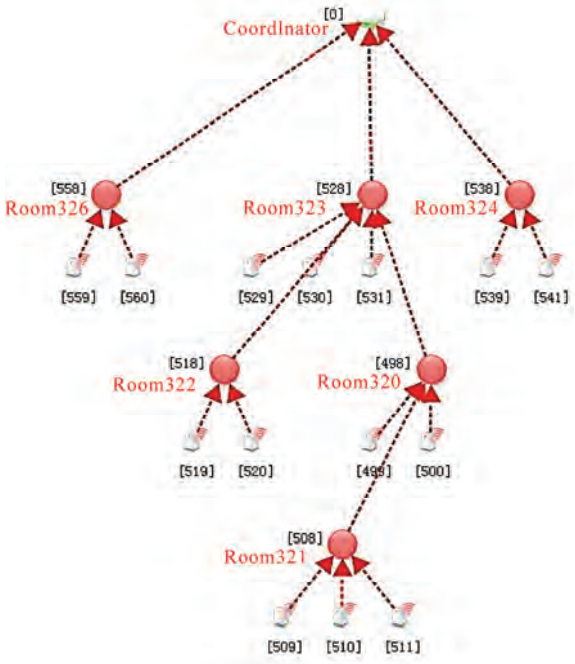


Figure 6. The network topology.

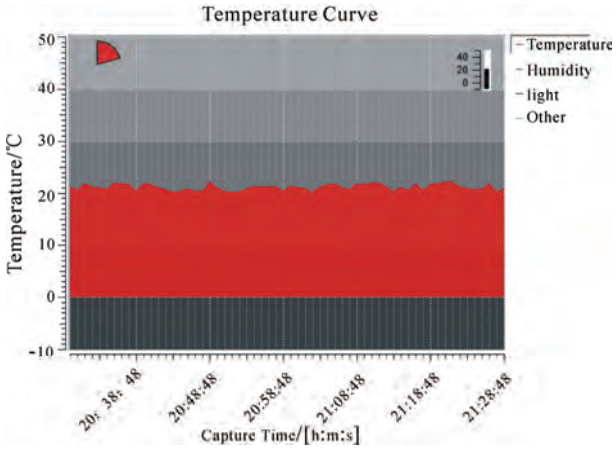


Figure 7. The temperature curve of room 323.

	Time Stamp	Room Id	Warning Infomation
0	2010/07/03 11:21:06	322	Temperature excess of the expected range
1	2010/07/03 11:21:12	322	Temperature excess of the expected range
2	2010/07/03 11:21:18	322	Temperature excess of the expected range
3	2010/07/03 11:21:24	322	Temperature excess of the expected range
4	2010/07/03 11:21:30	322	Temperature excess of the expected range
5	2010/07/03 11:21:36	322	Temperature excess of the expected range

Figure 8. The alarm information.

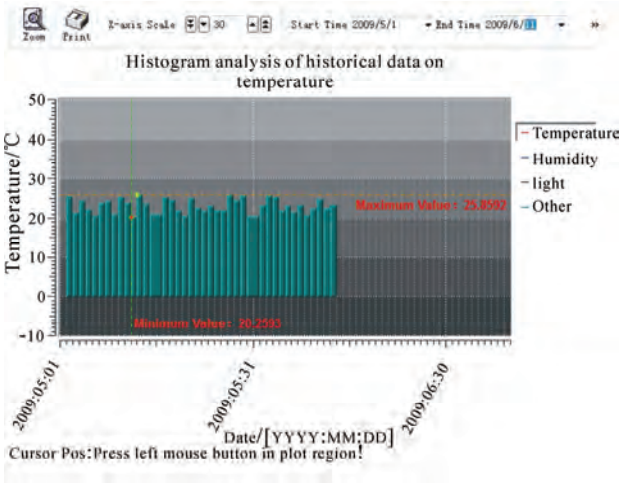


Figure 9. The historical temperature analysis chart.

2) In current WBMSNs, a node is always the cluster-head once it is configured as a cluster-head. This leads to exhausting the energy of these cluster-heads quickly. In order to solve this problem, it is necessary to explore a mechanism for switching cluster-heads automatically.

3) How to deploy nodes to optimize the network performance is also the focus of further research.

6. References

[1] B. Robert, "Buildings Can Play a Key Role in Combating Climate Change," 2007. http://www.unep.org/Documents.Multilingual/Default.asp

[2] K. M. Michael and R. B. Michael, "Pros & Cons of Wireless," *American Society and Heating, Refrigerating, and Air-Conditioning Engineers Journal*, Vol. 44, No. 11, November 2002, pp. 54-61.

[3] L. M. Sun, J. Z. Li, Y. Chen and H. S. Zhu, "*Wireless Sensors Networks*," Tsinghua University Press, Beijing, 2005.

[4] A. Willing, "Wireless Sensor Networks: Concept, Challenges and Approaches," *Elektrotechnik & Informationstechnik*, Vol. 123, No. 6, June 2006, pp. 224-231.

[5] C. Adam and W. Jakub, "On Applications of Wireless Sensor Networks," *Internet-Technical Development and Applications*, Springer, 2009, pp. 91-99.

[6] S. Thomas, D. F. Henri and V. Martin, "Sensor Scope: Experiences with a Wireless Building Monitoring Sensor Network," *Proceedings of the Workshop on Real-World Wireless Sensor Networks*, Stockholm, 2005.

[7] J. Y. Chang, J. Y. Kim and O. Kwon, "Demo Abstract: Control City—Integrating Wireless Sensor Networks and Building Management Systems," *Proceedings of the* 2009 *International Conference on Information Processing in Sensor Networks.* San Francisco, February, 2009, pp. 421-422.

[8] W. S. Jang, W. M. Healy and J. S. Miroslaw, "Wireless Sensor Networks as Part of a Web-Based Building Environmental Monitoring System," *Automation in Construction*, Vol. 17, No. 6, August 2008, pp. 729-736.

[9] D. Tessa, G. Elena and B. James, "Wireless Sensor Networks to Enable the Passive House-Deployment Experiences," *Smart Sensing and Context*, Berlin Heidelberg: Springer, 2009, pp. 177-192.

[10] W. S. Jang and W. M. Healy, "Wireless Sensor Network Performance Metrics for Building Applications," *Energy and Buildings*, Vol. 42, No. 6, 2010, pp. 862-868.

[11] W. B. Heinzelman, A. P. Chandrakasan and H. Balakrishnan, "An Application-Specific Protocol Architecture for Wireless Micro-Sensor Networks," *IEEE Transactions on Wireless Communications*, Vol. 1, No. 4, 2002, pp. 660-670.

[12] W. B. Heinzelman, A. P. Chandrakasan and H. Balakrishnan, "Energy-Efficient Communication Protocol for Wireless Micro-Sensor Networks," *Proceedings of the* 33*rd Hawaii International Conference on System Sciences*, Hawaii, 2000, pp. 1-10.

[13] W. Ye, J. Heidemann and D. Estrin, "An Energy-Efficient MAC Protocol for Wireless Sensor Networks," *Proceedings of the* 21*st International Joint Conference on IEEE Computer and Communications Societies*, New York, June 2002, pp. 1567-1576.

Wireless Sensor Network, 2010, 2, 710-717
doi:10.4236/wsn.2010.29086 Published Online September 2010 (http://www.SciRP.org/journal/wsn)

Data-Centric Routing Mechanism Using Hash-Value in Wireless Sensor Network

Xiaomin Zhao, Keji Mao, Shaohua Cai, Qingzhang Chen*
College of Computer Science and Technology Zhejiang University of Technology, Hangzhou, China
E-mail: *qzchen@zjut.edu.cn*
Received July 2, 2010; *revised August* 18, 2010; *accepted September* 15, 2010

Abstract

Traditional routing protocols as TCP/IP can not be directly used in WSN, so special data-centric routing protocols must be established. The raised data-centric routing protocols can not identify the sensor nodes, because many nodes work under a monitoring task, and the source of data is not so important some times. The sensor node in the network can not judge weather data is come from the some sink node. What's more, the traditional method use IP to identify sensors in Internet is not suitable for WSN. In this paper, we propose a new naming scheme to identify sensor nodes, which based on a description of sensor node, the description of a sensor node is hashed to a hash value to identify this sensor. The different description generates different identifier. Different from IP schema, this identifier is something about the information of the sensor node. In the above naming scheme, we propose a new data-centric routing mechanism. Finally, the simulation of the routing mechanism is carried out on MATLAB. The result shows our routing mechanism's predominate increase when network size increase.

Keywords: Wireless Sensor Network, Routing, Hash Value, Sensor Identifier

1. Introduction

Wireless sensor network has become a research focus of computer technology; it is a complex system which combines the sensing, embedded computing, distributed processing, wireless communications, and many other technologies. Since the concept of sensor network was proposed, a growing number of research institutions began to join into the field. The WSN could collect information from physical world directly, then it links with the logic world through the network [1], it greatly extends the traditional network ability and the ability of human being to control the physical world.

2. Recent Research

Wireless Sensor Network is a large scale network of hundreds or thousands of sensor nodes. These sensor nodes are networking by self-organization. WSN with many features: sensor nodes are random spread in sensor field, so it is infrastructure-less; the sensor node is energy restricted, so node can not support long communication range, and they communicate with other nodes by multihop. What's more, WSN is scalability, easy deployment, low cost, application-related and so on.

For those features, WSN can not direct use traditional networking technologies. And researchers have begun to study its exclusive technology. Wireless sensor networks have many key technologies, such as routing protocols, MAC protocols, location, time synchronization, etc., in these key technologies, the routing protocols is a research hotspot.

The routing protocols of traditional networks focus on the availability of a high quality of service and the equitable and efficient of network bandwidth. In the wireless sensor network, the node energy is limited and nonrenewable, the node can not support the large distance of information transmission, so the data packets pass through network to reach the destination node by multihop way. So the routing protocols need to use energy efficiently. At the same time, the number of nodes is very large in WSN, the node can only get the nearby network topology information, and every node should find its routing path according this partly information. These problems are not encountered in traditional networks, which determine the routing algorithm of traditional networks can not apply to wireless sensor

*Corresponding author.

networks. One important goal of the routing protocol of sensor network is to maintain a longer network lifetime. Currently, researchers have classified the proposed routing protocols in WSN into four categories [3-5]. The first category is hierarchical routing protocol, such as LEA CH; the second type is geographical routing, such as GAF, GEAR. The third category is a reliable route routing protocol, such as SPEED. The fourth category is data- centric routing protocols, such as SPIN, Rumor, DD and so on.

For wireless sensor networks is a network which closely related to application, and it is data-centric, more researchers focus on data-centric routing protocols and got many achievements.

SPIN [6] is a kind of data-centric routing protocols; it sends messages though network by negotiation. When a node A want to send a message, it first send a ADV, ADV is used to broadcast meta-data which is a description of the data that ready to send; another node B who receive the ADV and it is willing to receive the data send REQ back to A to request data; at last, the node A send the data to B.

Directed diffusion [7] is a data-centric routing mechanism. "Interests" in particular sensing information are disseminated over the sensor network starting from the sink. "Gradients" back towards the sink are constructed in the meanwhile. This essentially uses flooding to subscribe to interested events. Once a sensor detects the interested events, an energy-efficient routing path between this sensor and the sink will be reinforced. To maintain robust paths for information flows, the sinks need to periodically cast their "interests" to the sensor network. Directed diffusion also supports in-network processing. Every sensor is equipped with local memory to cache sensor data for identical data aggregation or suppression.
Rumor routing [8]: when a sensor node in sensor field sensing an event, it generates a proxy message (agent), agent messages randomly select a neighbor node to forward, at the same time the query sent by sink node is also spread in the network randomly. When the two of them meet, the path from source node to sink node formed. Rumor overcomes the defect of energy consuming DD in broadcast interest in network, but it is so randomness that the delay of data is obviously.

3. Main Title

The nodes in SPIN, Rumor, and directed diffusion routing mechanism algorithms are not have an identifier. Because there are too many nodes in a WSN, to maintain an identifier will consuming lots unnecessary energy, what's more, WSN is a network of data-centric, care little about where the data come from but the detail of the data; so identifier in WSN is seems not so important as in traditional network. But this no identifier also bring many problems, we can not know where the data is exactly come from. Take DD for example, a node in network can not judge the different interest come from which specific node, in **Figure 1**, several sink nodes all broadcast a interest , and they reach node A, in this moment A can not distinguish the source of the interest, and do not know how to deal with it. And in **Figure 2**, an interest reach node A from one sink node in different path. The node can throw all interest but the first one or it can establish gradient for each interest. But what about the situation in **Figure 3**.

In this paper we propose a new naming schema to give an identifier to each node in WSN. Considering the feature WSN have, the identifier is not just numbered like IP address, but something about data-centric. Base on this naming schema, we propose a new routing mechanism. At the last of paper, we have simulate to verification the effectively of the routing mechanism.

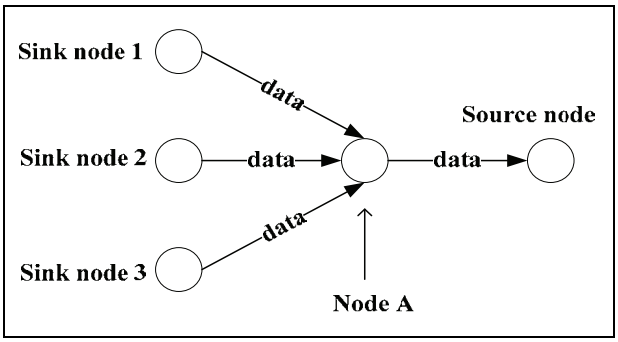


Figure 1. Interests from different sink nodes.

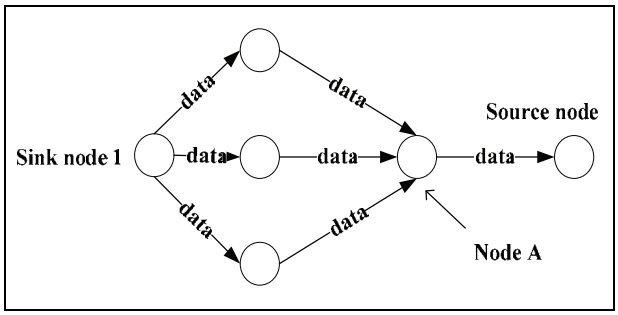


Figure 2. Interests from one sink node.

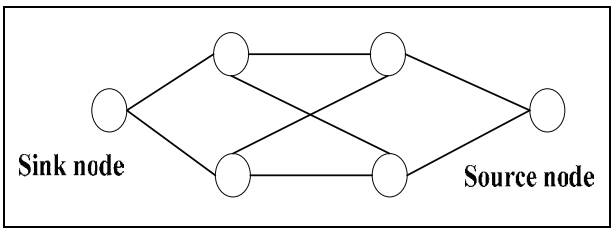


Figure 3. The complex topology from source node to sink node

4. Naming Scheme Based on Hash Value

Paper [9,10] proposed a data-centric storage scheme, use data itself to describe its storage location, that is the name of the data represent a keyword, you can use this keyword to find the data. And queries can be routed to the data directly by the name of the corresponding node. In this paper, we propose a new naming scheme according to data-centric scheme above. We first describe the data by attribute pairs, for example: there is a sensor node in the room 621 of library to monitor the temperature, and then it has a description below:

ATTRIBUTION{
Service = temperature
*Room = R*621
Building = library
} (1)

If the node has more than one sensor model, performs several monitor task, for example the sensor node in room 621 not only monitor the temperature but also object monitor, then it can also describe like this:

ATTRIBUTION{
Service = Object Monitor
*Room = R*621
Building = library
} (2)

One node could have several descriptions, but one description can only describe one node.

Meanwhile, we use the same naming scheme to name the query, such as a query to check the room 621's temperature, and then the description of this query is:

ATTRIBUTION{
Service = temperature
*Room = R*621
Building = library
} (3)

After each node and query has its description, we use a certain hash algorithm to generate a hash value, and use this hash value to identify the node or query. For the above description (1) and (3), (1) describes the detail of node in room 621 and (3) describes the query whose destination is node in room621. We can find that the description of (1) and (3) is identical; by the same hash function the query will generate the same hash value with its destination node. So it can be routed directly and correctly. Thus each node in network will have a unique identifier.

According to this naming scheme, we can map the complex physical topology of a network to one-dimensional logical topology. As shown in **Figure 4**. One node could have several ID to identify itself, for example v6 mapped to a3 and a6, but one ID can unique identify a node.

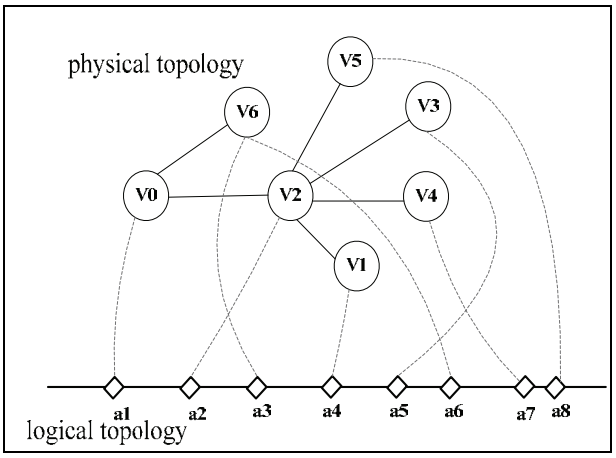


Figure 4. Mapping from physical topology to logical topology.

This naming scheme based on hash value is aiming to solve the problems in raised data-centric routing mechanism which has no uniform identification. And this naming scheme is different from the traditional network which base on IP address, the ID of node is not just a number, but a keyword of data, it is data-centric.

5. Routing Mechanism

Base on the above naming schema, each node maintains a routing table, a logical neighbor table and a physical neighbor table, and according to those routing information, messages can be delivered to destination efficiently.

5.1. Tables of Routing Information

5.1.1. Routing Table

Routing table is used to help messages to deliver to destination, it maintains three parameters. The first parameter is leader node, leader nodes are those identifying hash-value that closest to $R/2^n$ away from the hash value of local sensor, while R as the range of the hash domain, and n is the routing scope. The second parameter is path, it record the path to the leader and the third parameter is cost it spends from local node to leader. **Figure 5** is the structure of the routing table.

For one node, there are several scopes about leader. The first scope is the node whose ID is closest to R/2, and the second scope is the node whose ID is closest to R/4, and third scope R/8…, we select leader by the formula below:

$$U_n = v + \frac{R}{2^n}$$

"*n*" is the scope of leader. So, node v0's namespace can be segmented as in **Figure 6**.

Leader	Path	Cost

Figure 5. Routing table.

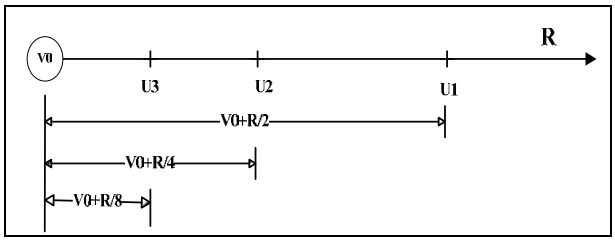


Figure 6. Name space of node v0.

With the segmentation of the name space and the select of leader, routing table records the path to the nodes whose identifying hash values are exponentially changed, and it makes message to reach the destination quickly.

5.1.2. Logical Neighbor Table

Each node also maintains a logical neighbor table; the logical neighbor records the node whose hash value is closest to local node. It provides a shortcut to reach the destination.

Logical neighbor: the node whose hash value is closest to local node. Path: the path from local node to logical neighbor. Cost: the spending from local node to logical neighbor. **Figure 7** shows the logical neighbor table.

5.1.3. Physical Neighbor Table

Each node also maintains a physical neighbor table, it record the physical neighbor which is one hop away from local node.

The parameters maintain by physical table is similar with logical neighbor table.

Physical neighbor: the node which is one hop away from local node. Path: the path from local node to physical neighbor. Cost: the spending from local node to physical neighbor. **Figure 8** shows the physical neighbor table.

5.2. Routing Process

Wireless sensor network has many restrictions in various resources, so the process of routing should be simple and efficient. When the networking begins, nodes in network broadcast "hello" packet to other nodes. And node constructs its routing tables by received packet. The step is below: (local node with hash value V0 receive a message send by node whose hash value is V1)

Logic neighbor	Path	Cost

Figure 7. The logic neighbor table.

Physical neighbor	Path	Cost

Figure 8. The physical neighbor table.

1) Node V0 receives a hello packet which contains its destination V1.

2) Node V0 judges whether the packet is sent from its physical neighbor who is only a hop away, if yes, records the node to its physical neighbor table, and continue.

3) Node V0 judges whether it is the first packet received, If yes, records V1 to its logical neighbor table, else compare with the item which is already in logical table, if it is closer to local node's hash value, replace the original item with new V1.

4) Node V0 should determine whether the V1 is a leader, a simple calculation of distance between the hash value can be drawn. If yes, records it in routing table and records the path to it, else forward it.

5) Wireless sensor networks based on specific size and the number of nodes to define the hello packet time to live. Before the routing mechanism works, we should design a reasonable life time for "hello" packet to save energy consumption of network. **Figure 9** is the flow chart.

The judge in dashed box is to construct node's routing table, **Figure 10** shows the detail flow chart.

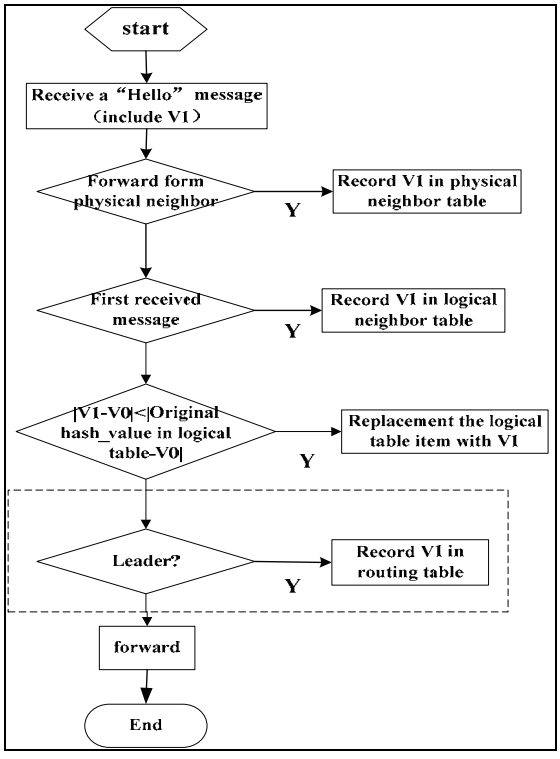


Figure 9. Flow chart of routing tables establish.

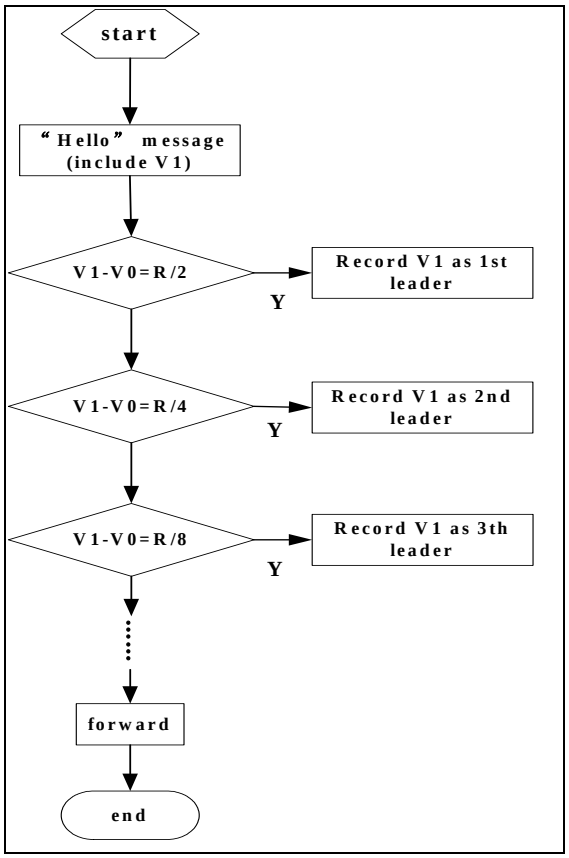


Figure 10. Flow chart of leader definite.

With the three tables of routing information, the node can forward every packet wherever its destination is. The routing mechanism as:

1) When a node receives a packet from other node, it first checks its physical neighbor table and logical neighbor table to determine whether there is a path to the destination node, if yes, send the packet to node record by either table.

2) Otherwise, the node calculate its own hash value and the packet hash value to determine which scope of leader the node need to help to forward the packet. For example: If the packet's hash value is closer to the 1st leader, then it forwards the packet to its 1st leader.

3) Repeat step one, step two, until the packet accurately reach its destination.

5.3. Route Maintenance

By the networking is complete, we need maintenance the routing information regular to ensure the correctness. The simplest method is to periodically broadcast the node's hash value contains in a "hello" packet, to declare its own survival. If one node hasn't broadcast a hello packet for a long time, then other node will think it is for some reasons departed from network, and the path contains this node will failure.

When invalid nodes leaver or new nodes join the network, the routing table and neighbor table need to be updated.

5.3.1. New Nodes Join the Network

When a new node needs to join a network, its routing information should be constructed first.

1) To join a sensor network, a new sensor first contacts one of its "physical" neighbors randomly to construct physical neighbor table. A sensor's physical neighbors are those sensors geographically close.

2) Then this sensor generates $\lceil \log R - 1 \rceil$ joining requests with a key R/2n plus its identifying hash value. Conceptually, these joining requests will be forwarded to the sensors that are numerically closest to each key. The Sensors visited by a joining request will be recorded in this request to track the routing path. When those requests reach the destination, the destination node records the new node and reply. After collecting all the replies, the new sensor can construct its own routing table.

3) This sensor generates a joining request with a key of its own identifying hash value. This joining request is used to construct the neighbor table. When the numerically closest sensor to that hash value receives the joining request, it returns its logical neighbors and the paths to the neighbors to the new sensor and meanwhile updates its neighbor table with that hash value and the routing path recorded in the request. The reply of this reply provides the information of node's logical neighbor.

5.3.2. Nodes Withdraw from the Network

Wireless sensor network is a kind of dynamic, adaptive network, if a node's energy is exhaust, or for other reasons lost its contact with the network. So the routing information is not correctness, to support robustness of the sensor network, the sensors need to update the routing tables periodically.

5.3.3. Support for Mobile Node

Although most of the nodes in wireless sensor network are fixed, there is a part of node need mobility, and those nodes always play an important role. How to support mobile nodes properly is a new challenge in wireless sensor network. **Figure 11** shows the WSN with mobile node A.

By the naming schema raised in paper, each nodes use a hash value to identify themselves, so we can map the complex physical topology into a 1-D logical topology, as **Figure 1** describe. In this case, as shown in **Figure 7**, a mobile node S5, it moved from T0, T1, T2 moment, but the hash value of S5 is changeless, so whenever the

physical topology of the network changes, the logical topology is changeless, shows in **Figure 12**, because the identifier of the mobile node is abiding.

In this paper, we support mobile nodes by the routing information that each nodes maintains. Firstly, we request node add an attribute to describe the node's mobility. And then we add constraints to routing tables:

1) Each node maintain a routing table, all the nodes maintained in this table must be immovable.

2) An immovable node will maintain two different logical neighbors, an immovable logic neighbor table and a mobile logical neighbor table. However, the movable sensor needn't maintain a mobile logical neighbor table.

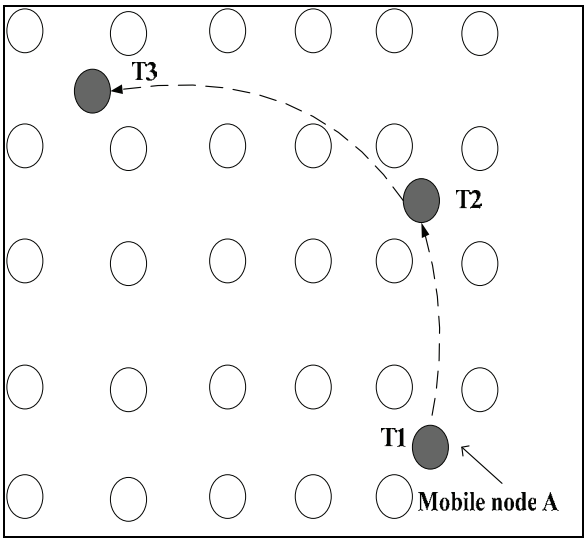


Figure 11. Mobile nodes in WSN.

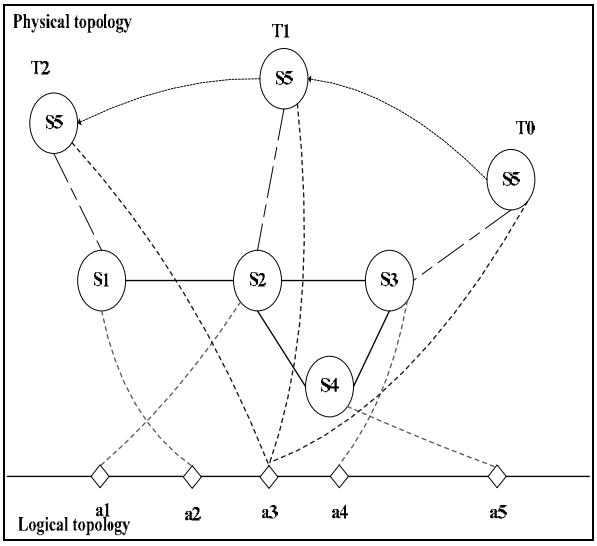


Figure 12. Mapping from physical topology to logical topology with a mobile node.

3) Movable sensors also maintain an immovable logic neighbor table and a mobile logical neighbor table. Different from tables immovable node maintain, it didn't maintain the path to neighbor node for those path is variable.

By the constraints written above, When a mobile node willing to send a packet to the immovable node, it can just forward packet to its nearest immovable physical neighbor, and then the packet will be forwarded according to routing mechanism. If a node has a packet to send to the mobile node, there is no direct path to the mobile node, the sending node will just sent it to the mobile node's logical neighbors. Mobile node sends regular send hello messages back to its neighbor to get the packet. In this way, data flow from mobile node to immovable node is established.

Here follows the concrete steps:

1) The mobile node broadcast a hello packet to other node when reach the new place to get its physical neighbor information.

2) When a mobile node to send a packet, it send the packet to its physical neighbors.

3) When an immovable node has to communicate with mobile node, either node sends packet to it or replies. It sends the packet to mobile node's logical neighbor.

4) The mobile node periodically sends hello packets to its logical neighbors to get the packet.

6. Simulation

We compare proposed routing mechanism to Directed Diffusion, Rumor, and Flooding. **Figure 9** is describes 100 nodes random distribution in the 100 meter multiply 100 meter space. **Figure 13** depict node A's logical neighbor and leader. We can see from **Figure 14** that the logical neighbor and leader are well-proportioned in the region. In **Figure 14**, red arrows point to node A's leader, and blue arrows point to its logical neighbor.

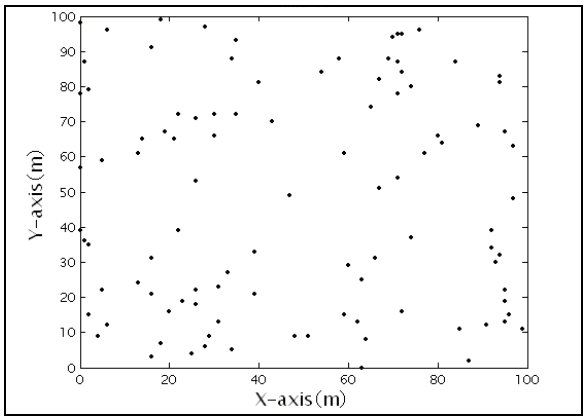


Figure 13. 100 nodes in 100m*100m.

The **Figure 15** shows the packet number required for a success query

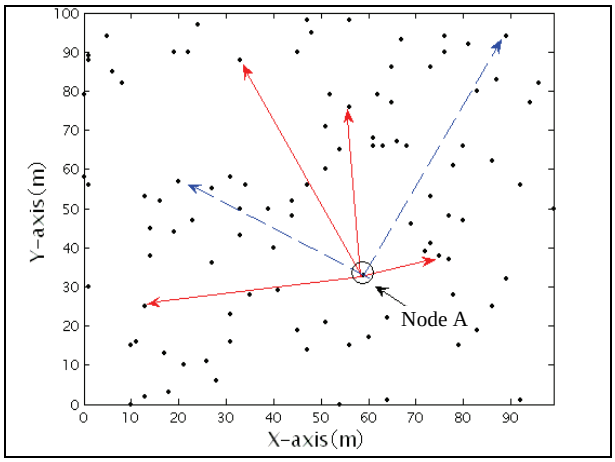


Figure 14. Logical neighbor and leader of node A.

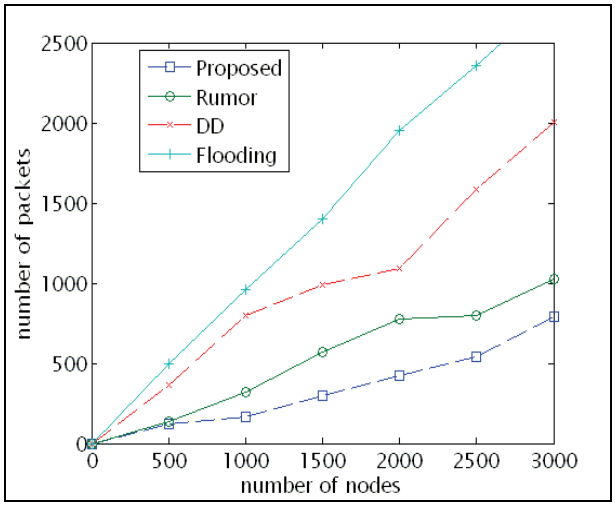


Figure 15. Packet number of a query.

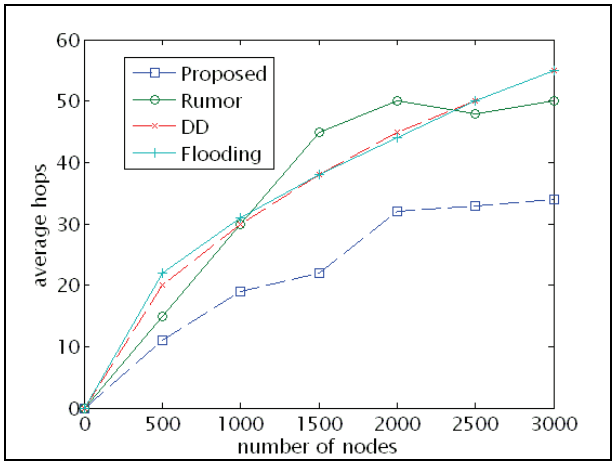


Figure 16. Average hops of a query.

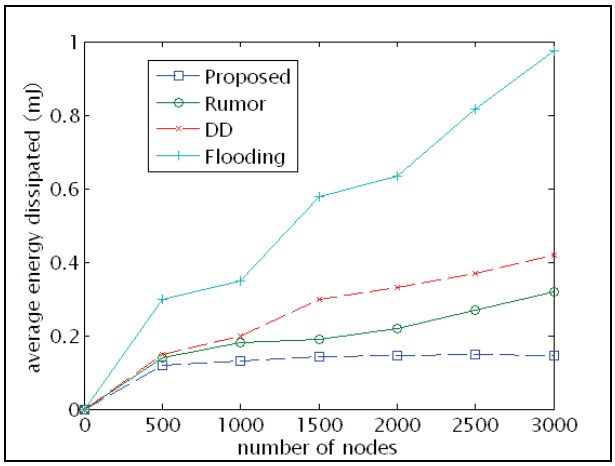


Figure 17. Average energy dissipated of a query.

The **Figure 16** shows the average hops a packet cost to destination node. From which we can see that proposed routing mechanism is advantageous when the network scale increased.

The **Figure 17** is the average energy dissipated for a discovered routing path.

7. Conclusions

The proposed routing mechanism is superior to other data-centric routing, especially when nodes in a WSN increased.

8. References

[1] D. Chen, Z. W. Zheng and J. J. Li, "Research on Wireless Sensor Network," *Computer Measurement and Control*, Vol. 12, No. 8, 2004, pp. 701-704.

[2] L. M. Sun, "Wireless Sensor Network," Tsinghua University Press, Beijing, 2005.

[3] W. Ye, J. Heidemann and D. Estrin, "An Energy-Efficient MAC Protocol for Wireless Sensor Networks," *Proceedings of IEEE Infocom*, New York, 2002, pp. 1567-1576.

[4] W. Ye, J. Heidemann and D. Estrin, "Medium Access Control with Coordinated Adaptive Sleeping for Wireless Sensor Networks," *IEEE/ACM Transactions on Networking*, Vol. 12, No. 3, December 2004, pp. 493-506.

[5] Y. Li, W. Ye and J. Heidemann, "Energy and Latency Control in Low Duty Cycle MAC Protocols," *Proceedings of the IEEE Wireless Communications and Networking Conference*, New Orleans, March 2005, pp. 676-682.

[6] J. Kulik, W. R. Heinzelmann and H. Balakrishnan, "Adaptive Protocols for Information Dissemination Information in Wireless Sensor Networks," *Proceedings of the* 5*th ADM/IEEE Mobicom Conference*, Seattle, 1999, pp. 174-185.

[7] C. Intanagonwiwat, R. Govindan and D. Estrin, "Directed

Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks," *Proceedings of the* 6*th annual international conference on Mobile computing and networking*, Boston, 2000, pp. 56-67.

[8] A. Boulis, S. Ganeriwal, M. B. Srivastava, "Aggregation in Sensor Networks: an Energy-Accuracy Trade-Off," *Sensor Network Protocols and Applications*, Vol. 1, No. 2, September 2003, pp. 317-331.

[9] S. Ratnasamy, B. Karp, L.Yin, F. Yu, D. Estrin, R. Govindan and S. Shenker, "GHT: A Geographic Hash Table for Data-Centric Storage," *Proceedings of the* 1*st ACM International Workshop on Wireless Sensor Networks and Applications*, Atlanta, September 2002.

[10] S. Shenke, S. Ratnasamy, B. Karp, R. Govindan and D. Estrin, "Data-Centric Storage in Sensornets," *ACM SIGCOMM Computer Communication Review*, Vol. 33, No. 1, January 2003, pp.137-142.

Wireless Sensor Network, **2010, 2, 718-723**
doi:10.4236/wsn.2010.29087 Published Online September 2010 (http://www.SciRP.org/journal/wsn)

Opportunistic Routing for Time-Variety and Load-Balance over Wireless Sensor Networks

Nan Ding, Guozhen Tan, Wei Zhang
Department of Computer Science, Dalian University of Technology, Dalian, Liaoning, China
E-mail: {*dingnan,gztan*}*@dlut.edu.cn, wesley.cheung@*163*.com*
Received June 28, 2010; *revised August* 2, 2010; *accepted September* 4, 2010

Abstract

To aware the topology of wireless sensor networks (WSN) with time-variety, and load-balance the resource of communication and energy, an opportunistic routing protocol for WSN based on Opportunistic Routing Entropy and ant colony optimization, called ACO-TDOP, is proposed. At first, based on the second law of thermo-dynamics, we introduce the concept of Opportunistic Routing Entropy which is a parameter representing the transmission state of each node by taking into account the power left and the distance to the sink node. Then, it is proved that the problem of route thinking about Opportunistic Routing Entropy is shown to be NP-hard. So the protocol, ACO-TDOP, is proposed. At last, numerical results confirm that the ACO-TDOP is energy conservative and throughput gainful compared with other two existing routing protocols, and show that it is efficacious to analyze and uncover fundamental of message transmission with Opportunistic Routing in wireless network using the second law of thermodynamics.

Keywords: Wireless Sensor Network, Load-balance, Time-variety, Opportunistic Routing Entropy

1. Introduction

MULTI-HOP wireless networks, the same as mobile Ad hoc networks and wireless mesh networks, is considered to be convenient and cost-effective solutions in many important areas, such as traffic monitoring, environmental monitoring, and military security. Due to effects of multi-path fading, signal interfering and path losses, the link of WSN is time-varying. Besides, the power and working state of each node is time-varying, too. So the key challenge of multi-hop wireless networks protocol is ensuring that the "best" receiver of each packet forwards it.

At present, most of the traditional multi-hop routing protocols in WSN typically adopt routing schemes and techniques similar to those in wired networks [1]. The protocols, such as DSR, OLSR, AODV and LEACH, have proposed to update the network routing periodically in order to adapt to the change of network, but messages are still transmitted through static routing. These methods can partly solve the problem of energy consumption balance; however they cannot solve the dynamic changes of the link which is caused by the time-variety of the network [2].

The opportunistic routing which is proposed in recent years is a new dynamic routing based on multi-hop. It proposes a transmission method based on the opportunistic, that is, next relay is selected dynamically for each packet and each hop. Through a large number of experiments [3-6,7], they have verified the rationality and practicality of the opportunistic routing compared with traditional route in the network traffic process. To ensure the reliability and validity of the dynamic transmission network, the key of opportunistic routing algorithm is how to select the next hop node. Currently, there are two researching aspects on opportunistic routing:

One aspect is to establish best routing for wireless network [8,9]. This aspect primarily builds the shortest path or near the approximate shortest path to improve the network throughput performance. Kai Zeng *et al*. [10] set up an optimization model for the throughput of the opportunistic routing through a combination of graph theory and network flow theory.

The other aspect is on the balance of energy consumption in opportunistic routing [11-13]. Lakshmi V. *et al*. [14] proposes two opportunistic transmission strategies referring to energy, one of which is to eliminate the nodes whose residual energy is less than threshold during establishing the transmission path dynamically, and the

other is that the network nodes are divided into concentric ring-type structure while the receive node as the center. With the latter strategy, when the message is sent to the ring whose nodes are close to the sink node, it will consider the residual energy to make the distribution of energy consumption reasonable.

According to the studies above, the network transmission throughput rate and the energy balance are the key factors of dynamic establishment of opportunistic routing, and they are interactional and interdependent. For example, in order to achieve maximum network throughput rate and minimum delay, it is necessary to establish the shortest path, while the shortest path will lead to unbalanced energy consumption of the network. On the contrary, in order to balance the energy consumption within the network, packet transmission will avoid some node which is on the shortest path but has little energy left, which will increase the number of hops of packet transmission, increasing the time delay and reducing the network throughput rate. To coordinate these two factors, some scholars have proposed to use the opportunistic routing which is mainly to use heuristic algorithms, such as ant colony algorithm, to get the transmission path selection strategy [12,15]. But at present, we lack the optimization model and analysis methods of transmission routing based on time-varying characteristic, which is necessary for improving the performance of opportunistic routing, and the theory related with static networks is unable in the time-varying networks [16].

To solve the problem, this paper gives the data transfer rule that refers to the residual energy of the node and the shortest path of the network, based on the second law of thermodynamic.

2. Problem Formulation

Taking into account the shortest path and energy balance with the time-varying characteristics of the network, we propose Opportunistic Routing Entropy which provides a theoretical basis for using opportunistic routing to create dynamic data transmission path. And we prove that it is a np-hard problem to get the best path which is time- varying and opportunistic.

2.1. Network Model and Assumptions

This article assumes that nodes in the network are distributed randomly in a rectangular area, and there properties as follows: 1)Within the network, the unique sink node is deployed in the center of the area, the rest of sensor nodes will not move any more after deploying, and the location information of each node is unknown. 2) The identity of each node is unique. 3) The status of each node is equal, with the same computing and communication capabilities, as well as initial energy.

Combining the properties above, the network model is defined as follows:

Definition 1 Network communication diagram.

The communication graph of an N-node wireless sensor network is an undirected graph, $\boldsymbol{G(V,E)}$. V is the set of communication node, and $\boldsymbol{V}= \boldsymbol{S_{sink}} \cup \boldsymbol{S_{sensor}}$, where $\boldsymbol{S_{sink}}$ is the sink node and $\boldsymbol{S_{sensor}}$ is the set of sensor nodes. Each sensor node has a fixed communication range, R. d_{ij} is the distance between node i and node j. $\boldsymbol{E}$ is the set of edges and $\boldsymbol{E} = e_{ij}(i,j \in \boldsymbol{V})$, $d_{ij} \le$ R.

Definition 2 Communication distance.

In graph $\boldsymbol{G}$, the communication distance is $\boldsymbol{D}(\boldsymbol{o}, \boldsymbol{s})$ which value is the hop count between node $\boldsymbol{o}$ and node $\boldsymbol{s}$.

Definition 3 Set of neighbor node.

The set of neighbor node is $\boldsymbol{N(u)}$, and $\boldsymbol{N(u)} = n_i(n_i \in \boldsymbol{V}, e_{uni} \in \boldsymbol{E})$, which includes the nodes which could communicate directly with node u.

Theorem 1 In WSN, the shortest path problem based on time-varying and the opportunistic method for transmission is shown to be NP-hard.

Proof. With the opportunistic method, the nodes of WSN could choose the best link to repeat the messages considering the condition of current network. Even though the same node sends two messages in succession, it may take different paths for transmission these two messages, if conditions of the network transmission link changed, for example interference *etc*. And the latter message may arrive earlier than former.

So the WSN transmission link working in the opportunity method is NO-FIFO, when thinking about the time-varying characteristics of communication links. Ariel Orda *et al* has proved that in the time-varying network, if the communication link is NO-FIFO, its complexity of the shortest path problem is NP-hard. Therefore, in Opportunistic Routing, the shortest path problem based on time-varying is shown to be NP-hard [16].

2.2. Opportunistic Routing Entropy and Programming

The second law of thermodynamicshe (also called entropy law) developed by Rudolf Clausius, is the essential theory of thermodynamic. Entropy indicates that the system develop to the internal stable state without external interference. The greater the entropy is, the more stable the system is.

Because nodes of WSN require the separate power supply as well as independent operation on normal conditions, its data packet transmission is in a relatively independent system. Nodes will consume their energy during the data packet transmission process. In order to

balance the energy consumption of each node to extend the network lifetime, we should choose the node with more residual energy, less energy consumption and shorter number of hops from the sink node as the next hop. Therefore, according the second law of thermodynamicshe and the data packet transmission process in the time-varying network model, we have defined the Opportunistic Routing Entropy and established the optimization model.

Definition 4 Opportunistic Routing Entropy.

Opportunistic Routing Entropy, S_{op}, is the measure of transmission state of each node in WSN. Referred to as *Shannon entropy*, Sop cannot decrease too. It is given by (1). The more S_{op}, the less energy left and the more hops to sink node.

$$S_{op}(u,\ t) = \frac{h_u / w_u(t)}{E_{op}(t)} \tag{1}$$

In (1), h_u is the hop count between nodes u and the sink node, $w_u(t)$ is the weight that node o communicates with node u which is in neighbor node set $\boldsymbol{N(o)}$ of node o at moment t. $\boldsymbol{E_{op}(t)}$ is residual energy, which can be calculated by A/D converter in real time.

During data transmitting, it will choose the node with small Opportunistic Routing Entropy as data forwarding node, and the Opportunistic Routing Entropy of this forwarding node will be progressively larger. S_{op} is on behalf of the probability of each node to be selected as next hop during data transmission. Smaller S_{op} is, more probability the node will be selected as next hop.

Therefore, during forwarding the data, it will choose the neighbor node with small S_{op} as the next hop node.

$$\begin{aligned} &\min.\ S(u,t) = \frac{h_u}{E_{op}(t)w_u(t)} \\ &s.t. \qquad E_{op}(t) = E_l(t) - E(t) \\ &\qquad\qquad E_l(t) - E(t) > 0 \\ &\qquad\qquad E_l(u),\ E(u) > 0 \\ &\qquad w_u(t) > 0 \\ &\qquad 1 - w_u(t) > 0 \\ &\qquad\qquad h_u > 0 \\ &\qquad w_u(t), h_u > 0 \end{aligned} \tag{2}$$

Referring to Theorem 1, it is shown to be NP-hard to get the shortest path of our model, so the problem of route thinking about Opportunistic Routing Entropy is also shown to be NP-hard.

According to computational complexity theory, any np-hard problem do not exist completed algorithm within polynomial-time unless P = NP [17]. At present, there are two solutions to such problems, one is completed algorithm which can get the optimal solution but with high time complexity, and the other is to use heuristic algorithms which can only get the approximate optimal solution but within polynomial-time. As a result, we propose a heuristic algorithm based on ant colony optimization for the model which we establish.

3. ACO-TDOP Based on Ant Colony Optimization

Ant colony optimization (ACO) takes inspiration from the foraging behavior of some ant species. Routing algorithms based on ACO algorithm have achieved good results in the network, especially in wireless sensor networks.

3.1. Probabilistic Decision Rule of ACO Based on S_{op}

When the source node o wants to send data, it will select the next hop node by following the random-proportional rule (see Equation(3)). It is a function thinking about the S_{op} and the amount of pheromone trail present on the connections between the nodes. So, it could choose the node with maximum value of $P(j, t)$ as the repeat node.

$$\mathrm{P}(j,t) = \frac{\tau(j,t) / S_{op}(j,t)^{\beta}}{\sum_{i \in N} (\tau(i,t) / S_{op}(i,t)^{\beta})},\ j \in N \tag{3}$$

where:

$P(j, t)$: the probability that the ant in the data packet move to node j at moment t.

$\tau(j, t)$: the pheromone level of node j at moment t.

$S_{op}(j, t)$: the selection entropy of node j at moment t.

$\boldsymbol{N}$: the set of neighbors of the current node.

β: a parameter which determines the relative influence of heuristic values $S_{op}(j, t)$ ($\beta > 0$).

3.2. Pheromone Updating Rule

When received the data packets from node u, node j will update its Pheromone $\tau(j, t)$. Introducing of the residual rate $\rho(t)$ (Equation (6)), we have improved the existing calculation model of pheromone.

Pheromone $\tau(j, t)$ based on residual rate:

$$\tau(j,t) = \rho(t)\tau(j,t\text{-}1) + \Delta\tau \tag{4}$$

$$\Delta\tau(t) = [1 + (h_u - h)] \times \Delta\omega \tag{5}$$

$$\rho(t) = \frac{E_{op}(t)}{E_{init}} \tag{6}$$

where $\rho(t)$ is the pheromone accumulated parameter, which indicates the proportion of pheromone remaining at moment t. When the node has little residual energy, its residual rate $\rho(t)$ is small and its evaporation of pheromone is faster relatively. $\mathbf{E_{init}}$ represents of the initial energy of the node, and the initial energy of each node is fixed and equal in our model. h_u is the hop count between nodes *u* and the sink node.

In Equation (4), $\tau(j, t)$ is the pheromone level of current node at moment t, and $\tau(j, t\text{-}1)$ is the pheromone level of current node at moment *t-1*. $\Delta\tau$ is the new added pheromone, and in Equation (5),if the value of $(h_i - h_j)$ is greater than zero, then it can conclude that node *j* is closer to the sink node than node *i*. Hence, the algorithm will reward the path from node *i* to node *j* by depositing more pheromones. If the value of $(h_i - h_j)$ is equal to zero, then it means that both nodes *i* and *j* have the same hop count to the sink node. As a result, the algorithm will deposit the amount of pheromone $\tau(j, t)$ on the path. If the value is less than zero, the algorithm will not deposit pheromone on this path.

3.3. ACO-TDOP Algorithm

Combining Opportunistic Routing Entropy with ant colony algorithm, we propose a new routing algorithm, ACO-TDOP, and **Figure 1** presents its Working state diagram.

The main process is as follows:

1) **Initial state**: Starting from the state, **S0**, initialize the system variable, and the sink node will flood the initialization packet to all the nodes in the network firstly. After the node receives this packet, it will compute the hop count to the sink node.

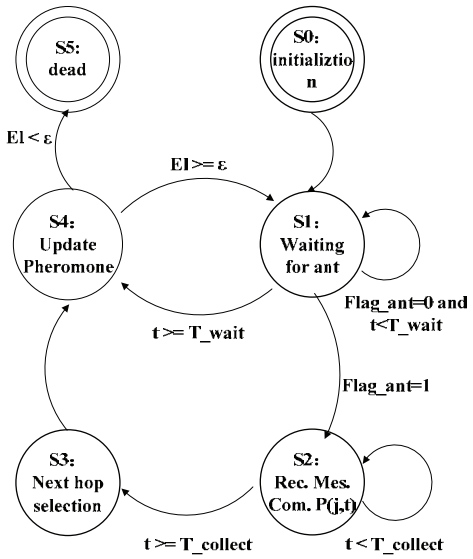


Figure 1. State machine of ACO-TDOP.

2) **Ant waiting and information gathering**: After initialization, the source node will enter the state **S1** to wait to send ants. When the source node wants to send ants, it will send a request packet to each neighbor node, and wait for feedback messages from neighbors, that process is the state **S2**. During the waiting time T_collect, we will calculate the selection probability $P(j, t)$ of each neighbor on the basis of feedback message, which include Opportunistic Routing Entropy S_{op} and pheromone $\tau(j, t)$. After the waiting time T_collect, the node will enter to state **S3**.

3) **Next hop node selection**: We will select the node whose $P(j, t)$ is largest among the neighbors in $\boldsymbol{N}$ as the next hop to send the ant.

4) ***Pheromone update***: This is the state **S4**, and there are two kinds of mode which can be converted to this state. One mode is to update the Pheromone immediately after sending a message over, the other is that the pheromone will be evaporated when the node is idle for a period of time such as T_wait. If the residual energy of the node is less than the threshold ε, then the node will be die and close its communication so as not to transmit any information.

4. Simulation Results

4.1. Simulation Environment

This experiment simulation environment is NS-2. For these simulation experiments, we assumed that there are 300 sensor nodes distributed randomly in a 300×300 square region. All nodes have the same transmission range. There is a single sink node located at coordinates (150, 150) of the wireless sensor networks, which receives the data of all source nodes for all the simulations. The parameters of simulations are listed in **Table 1**.

Simulation is mainly on two aspects, one of which is to compare the ACO-TDOP protocol with other routing protocols including traditional Ant Colony algrithm and EEABR on the performance, and the other is to get a dynamic path with ACO-TDOP protocol by tracking a packet from the source node to sink node.

4.2. Comparison with Other Protocols

For the Simulation, we will compare ACO-TDOP algorithm with other two algorithms. The first is the routing algorithm based on traditional ant colony algorithm which is used to solve the approximate shortest route in [7] and [13], and the second is EEABR routing algorithm which is proposed in [11]. EEABR is also based on ant colony but refer to the energy in the route, which means

that if there is more energy in one path, the density of pheromone in this path will increases more and the path will have more chance to be selected.

In the same simulation environment and net topology scale, we have done experiment for 100 times to compare the network lifetime of ACO, EEABR and ACO-TDOP algorithm. Network lifetime can be defined as the time elapsed until the first node (or the last node) in the network depletes its energy (dies). From **Figure 2**, we can see that ACO has shortest lifetime than others, and EEABR is shorter than ACO-TDOP. Because ACO-TDOP considers the left energy of each node in the path.

In order to analyze the residual energy distribution of nodes at the end of WSN, we also have done test more than 100 times. As a result shown as **Figure 3**, EEABR and ACO-TDOP algorithm has a more balanced energy consumption of the network, and there are more than 80% of the nodes whose residual energy is less than 35% of the initial value.

Figure 4 shows the comparison of the throughput rate after 100 tests. We can see that the throughput rate of BACO, EEABR and ACO-TDOP are quite similar to each other. As the test carry on, the BACO has a relatively stable throughput rate owing to without taking into account of energy consumption, while the throughput rate of EEBAR and ACO-TDOP is less than BACO for considering the residual energy of nodes. However, the throughput of EEBAR and ACO-TDOP is more than BACO, that is to say they make full use of the residual energy of the network, so they are better than BACO.

Table 1. Parameters of simulations.

Parameters	value
Simulated area	300×300
Number of sensor nodes	300
Initial energy	50J
Sink node location	(150,150)
Transmission range	25m
Transmitter energy consumption	50×10-8J/bit
Computing energy consumption	50×10-8J/bit
Circuit energy consumption	50×10-8J/bit

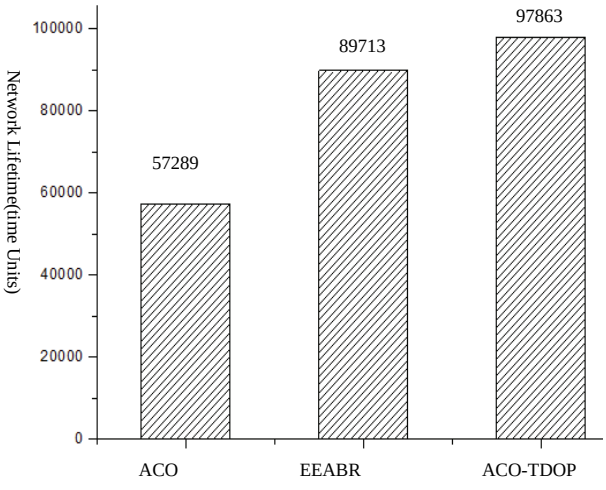


Figure 2. Lifetime of the networks.

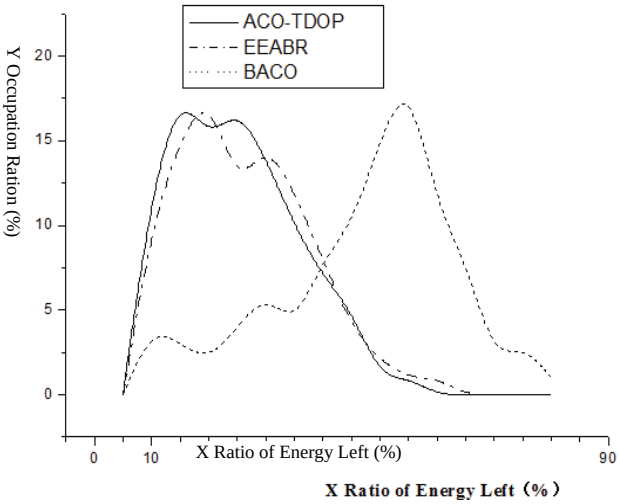


Figure 3. Distribution of Average Energy.

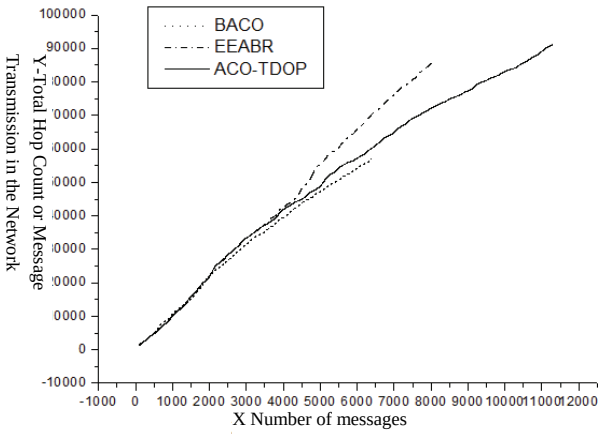


Figure 4. Distribution of throughput gains.

4.3. Path Tracking

In order to verify that ACO-TDOP routing algorithm can dynamically adjust the transmission path, we have recorded several dynamic paths between source node s and the sink node, as shown in **Figure 5**. We can see that the length of the path is only 10 hops at first, as the consumption of energy, the length of the path can reach to 16 hops.

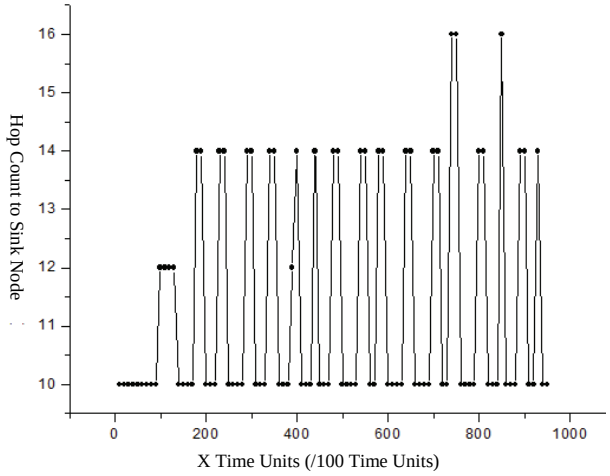


Figure 5. Path formation to sink node from the first run to network lifetime.

5. Conclusions

Based on the analysis of available opportunistic routing protocols, this paper analyzes and describes the message transmission process in the time-varying network model referring to the second law of thermodynamics, and proposes the Opportunistic Routing Entropy to replace the original entropy so as to indicate the network transmission status of each node. Further, combining Opportunistic Routing Entropy and Ant Colony Optimization, this paper raises an opportunistic routing protocol, ACO-TDOP, for time-varying network model considering the load balancing of the network resources. So the message is ensured that the "best" receiver of each packet forwards it. At last, numerical results confirm that the ACO-TDOP is energy conservative and throughput gainful compared with other two existing routing protocols, ACO and the EEBAR opportunistic routing protocol.

6. Acknowledgments

This paper was supported in part by the National Basic Research Program under No. 2005CB321904.

7. References

[1] D. Tse and S. Hanly, "Multi-Access Fading Channels: Polymatroid Structure, Optimal Resource Allocation and Throughput Capacities," *IEEE Transactions on Information Theory*, Vol. 44, No. 7, 1998, pp. 2796-2815.

[2] J. Lian, K. Naik and G. Agnew, "Data Capacity Improvement of Wireless Sensor Networks Using Non-Uniform Sensor Distribution," *Journal of Distributed Sensor Networks*, Vol. 2, 2006, pp.121-145.

[3] S. Biswas and R. Morris, "ExOR: Opportunistic Multi-Hop Routing for Wireless Networks," *Proceeding of ACM SIGCOMM* 05, ACM Press, Pennsylvania, pp. 133- 143, 2005.

[4] H. F. Hu and Z. Yang, "Collaborative Opportunistic Routing in Wireless Sensor Networks," *Journal on Communications*, Vol. 30, No. 8, 2009, pp.116-123.

[5] C. Luk, W. Lau and O. Yue, "An Analysis of Opportunistic Routing in Wireless Mesh Network," *Proceeding of IEEE International Conference on Communications*, Beijing, 2008, pp. 2877-2883.

[6] S. Flody, V. Jacobson, *et al*., "A Reliable Multicast Framework for Light-Weight Sessions and Application Level Framing," *IEEE/ACM Transactions on Networking*, Vol. 5, No. 6, 1997, pp.784-803.

[7] K. Zeng, W. J. Lou, *et al*., "Capacity of Opportunistic Routing in Multi-Rate and Multi-Hop Wireless Networks," *IEEE transactions on wireless communication*, Vol. 7, No. 12, 2008, pp. 5118-5128.

[8] Y. Zhang, D. Lukas, *et al*., "Improvements on Ant Routing for Sensor Networks," *Lecture Notes in Computer Science*, 2004, pp.154-165.

[9] K. Zeng, W. J. Lou, *et al*., "On End-to-End Throughput of Opportunistic Routing in Multi-Rate and Multi-Hop Wireless Networks," *Proceedings of the IEEE INFOCOM*, Phoenix, 2008, pp. 1490-1498.

[10] J. N. Laneman and G. Wornell, "Energy-Efficient Antenna Sharing and Relaying for Wireless Networks," *Proceeding of IEEE Wireless Communications and Networking, Chicago*, 2000, pp.7-12.

[11] T. Camilo, C. Carreto, *et al*., "An Energy-Efficient Ant-Based Routing Algorithm for Wireless Sensor Networks," *Ant Colony Optimization and Swarm Intelligence*, 2006, pp. 49-59.

[12] J. Li and P. Mohapatra, "Analytical Modeling and Mitigation Techniques for the Energy Hole Problems in Sensor Networks," *Pervasive and Mobile Computing*, Vol. 3, 2007, pp. 233-254.

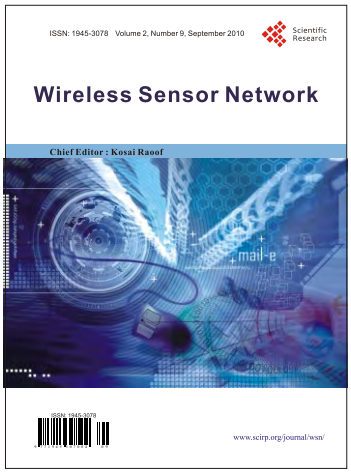
[13] V. Lakshmi, Thanayankizil, Aravind Kailas, *et al*., "Two Energy-Saving Schemes for Cooperative Transmission with Opportunistic Large Arrays," *Proceedings of Global Telecommunications Conference*, Washington, DC, 2007, pp.1038-1042.

[14] S. Okdem and D. Karaboga, "Routing in Wireless Sensor Networks Using Ant Colony Optimization," *Proceedings of Adaptive Hardware and Systems*, Istanbul, 2006, pp. 401-404.

[15] M. Dorigo and V. M. Etal, "The Ant System: Optimization by a Colony of Cooperating Agents," *IEEE Transaction on Systems, Man, and Cybernetics—Part B*, Vol.26, No. 1, 1996, pp. 1-13.

[16] A. Orda and R. Rom, "Shortest-Path and Minimum-Delay Algorithms in Networks with Time-Dependent Edge-Length," Vol. 37, No. 3, 1990, pp. 607-625.

[17] M. R. Garey and D. S. Johnson, "Computers and Intractability: A Guide to the Theory of NP Completeness," Whfreemanand Company, New York, 1979.



Wireless Sensor Network (WSN)

Call for Papers

http://www.scirp.org/journal/wsn

ISSN 1945-3078 (Print) ISSN 1945-3086 (Online)

WSN is an international refereed journal dedicated to the latest advancement of wireless sensor network and applications. The goal of this journal is to keep a record of the state-of-the-art research and promote the research work in these areas.

Editor-in-Chief

Dr. Kosai Raoof , GIPSA LAB, University of Joseph Fourier, Grenoble, France

Subject Coverage

This journal invites original research and review papers that address the following issues in wireless sensor networks. Topics of interest are (but not limited to):

- Network Architecture and Protocols
- Self-Organization and Synchronization
- Quality of Service
- Data Processing, Storage and Management
- Network Planning, Provisioning and Deployment
- Integration with Other Systems
- Software Platforms and Development Tools
- Routing and Data Dissemination
- Energy Conservation and Management
- Security and Privacy
- Developments and Applications
- Network Simulation and Platforms

We are also interested in short papers (letters) that clearly address a specific problem, and short survey or position papers that sketch the results or problems on a specific topic. Authors of selected short papers would be invited to write a regular paper on the same topic for future issues of the WSN.

Notes for Intending Authors

Submitted papers should not have been previously published nor be currently under consideration for publication elsewhere. Paper submission will be handled electronically through the website. All papers are refereed through a peer review process. Authors are responsible for having their papers checked for style and grammar prior to submission to WSN. Papers may be rejected if the language is not satisfactory. For more details about the submissions, please access the website.

Website and E-Mail

http://www.scirp.org/journal/wsn

Email: wsn@scirp.org

Wireless Sensor Network, 2010, 2, 655-723
Published Online September 2010 in SciRes (http://www.SciRP.org/journal/wsn/).

TABLE OF CONTENTS

Volume 2 Number 9 **September 2010**