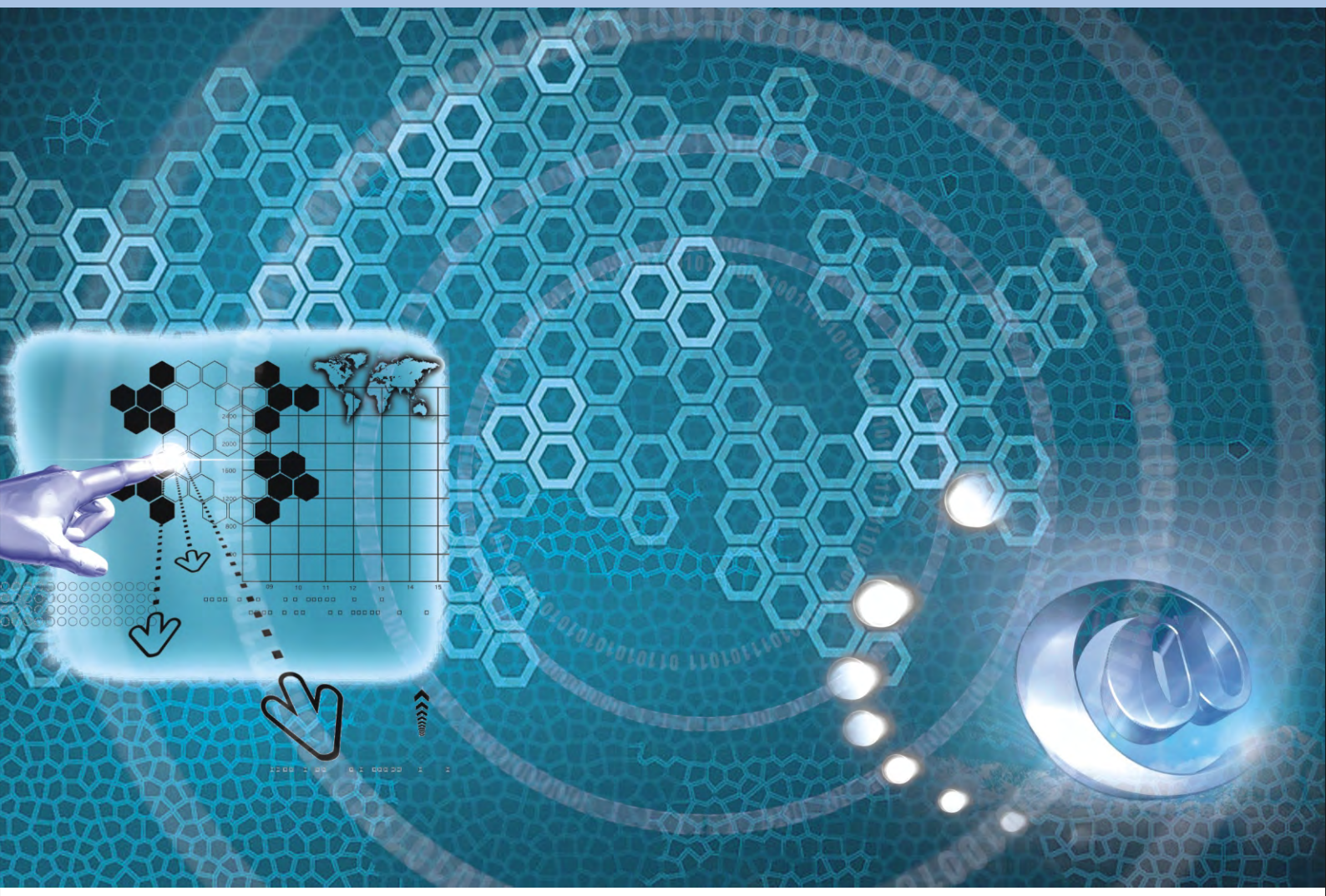
ISSN: 1945-3078 Volume 2, Number 7, July 2010

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/

Wireless Sensor Network, **2010, 2, 493-570**
Published Online July 2010 in SciRes (http://www.SciRP.org/journal/wsn/)

TABLE OF CONTENTS

Volume 2 Number 7 July 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, 493-503
doi:10.4236/wsn.2010.27061 Published Online July 2010 (http://www.SciRP.org/journal/wsn)

Modelling and Analysis of TCP Performance in Wireless Multihop Networks

Hannan Xiao[1], Ying Zhang[2], James Malcolm[1], Bruce Christianson[1], Kee Chaing Chua[3]
[1]*School of Computer Science, University of Hertfordshire, Hatfield, UK*
[2]*Datalink Electronics Group, Loughborough, UK*
[3]*Department of Electrical and Computer Engineering, National University of Singapore, Buona Vista, Singapore*
E-mail: *h.xiao@herts.ac.uk*
Received February 23, 2010; *revised March* 18, 2010; *accepted April* 21, 2010

Abstract

Researchers have used extensive simulation and experimental studies to understand TCP performance in wireless multihop networks. In contrast, the objective of this paper is to theoretically analyze TCP performance in this environment. By examining the case of running one TCP session over a string topology, a system model for analyzing TCP performance in multihop wireless networks is proposed, which considers packet buffering, contention of nodes for access to the wireless channel, and spatial reuse of the wireless channel. Markov chain modelling is applied to analyze this system model. Analytical results show that when the number of hops that the TCP session crosses is fixed, the TCP throughput is independent of the TCP congestion window size. When the number of hops increases from one, the TCP throughput decreases first, and then stabilizes when the number of hops becomes large. The analysis is validated by comparing the numerical and simulation results.

Keywords: Wireless Multihop Networks, TCP Modelling, TCP, Ad Hoc Networks

1. Introduction

An adhoc network has no fixed infrastructure: data transmission depends on the temporary location of nodes and the transit distribution of traffic in the fly. Originally designed for the wired Internet, TCP exhibits anomalous performance features in adhoc wireless networks, such as mobility induced retransmission [1], capture effect and unfairness [2] over the IEEE 802.11 MAC layer protocol, and RED-like packet dropping behaviour from link layer contention [3]. TCP performance in multihop adhoc wireless networks has been an active research topic. Extensive simulations have been conducted to obtain a better understanding of TCP behaviour in adhoc wireless networks and to find ways of enhancing TCP performance in that environment. There is not much effort, however, contributing to theoretical analysis of TCP performance in adhoc wireless networks.

This research work was motivated by the interesting observations of TCP in adhoc wireless networks (in simulations) reported in [1,2,4]: when a TCP connection runs over a static *string* topology (**Figure 1**), the TCP throughput measured decreases rapidly when the number of hops increases from one, and then stabilizes when the number of hops becomes large. Gerla gave an expression in [2] and predicted that the throughput would drop faster than exponentially with hop length, without giving further theoretical analysis.

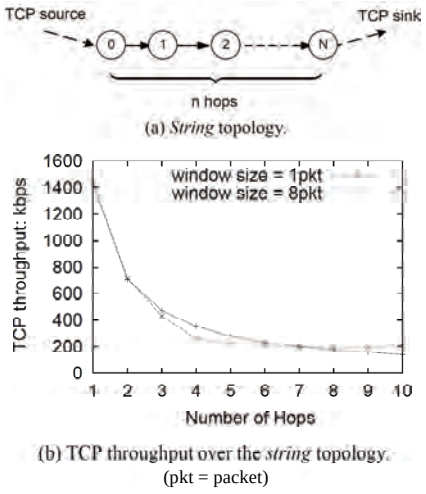


Figure 1. TCP performance over a *String* topology.

In this paper, a system model for analyzing TCP performance in adhoc wireless networks is proposed after examining the case of running one TCP session over a string topology. The system model considers the features of wireless multihop network such as contention of nodes for access to the wireless channel, packet buffering intermediate nodes, and spatial reuse of the wireless channel. A Markov chain modelling is applied to analyze this system model starting from a single hop case to multiple hop cases. Analytical results show that when the number of hops that the TCP session crosses is fixed, the TCP throughput is independent of the TCP congestion window size. When the number of hops increases from one, the TCP throughput decreases first, and then stabilizes when the number of hops becomes large. The analysis is validated by comparing the numerical and simulation results.

The rest of the paper is organized as follows. Section 2 discusses related work in two areas, *i.e.*, TCP performance in multihop wireless networks and system models for TCP analysis in different networks. Section 3 proposes a system model for analyzing TCP performance in multihop wireless networks, following which Section 4 applies a Markov chain modelling of the system model. Section 5 presents the theoretical analysis of the TCP throughput over the *string* topology from specific cases to the general case and Section 6 evaluates the analysis by comparing the simulation and numerical results. Finally Section 7 concludes the paper.

2. Related Work

2.1. TCP Performance in Wireless Multihop Networks

TCP was initially designed for wired networks to provide reliable data transmission by retransmitting lost packets due to network congestion that is detected at the sender by time-out or duplicated acknowledgments. Packet loss triggers TCP congestion control which reduces the congestion window size, effectively the sending rate, as a reaction to network congestion. TCP, however, has performed poorly in wireless multihop networks where packet loss may happen from various reasons other than network congestion, such as error-prone wireless link, link layer contention and routing failure or breakage. Many approaches have been proposed to improve TCP performance in the environment of wireless multihop networks from different aspects. Based on the nature of the approaches, they may be summarized into three groups as in [5]:

- TCP with feedback [1,6-11] where feedbacks from networks are sent back to the sender to distinguish packet losses due to congestion from link error, link contention or routing breakage or failure.
- TCP without feedback [12,13] where TCP congestion control strategy is modified according to the scenarios in multihop wireless networks.
- TCP with lower layer enhancement [14-18]. This approach is more popular recently as more work have been done to understand the cross-layer interaction, especially lower layer impact on TCP performance in wireless multihop networks.

The above work shares one similarity *i.e.*, they all observe TCP performance in multihop wireless networks by simulations and then propose improvements. Very few works, however, have been devoted on theoretical analysis of TCP performance in multihop wireless networks. We proposed a system model for TCP analysis over one-hop *string* topology as early as in 2001 [19] and then extended the modelling to multihop cases in [20]. So far our analysis only applies when there is no packet loss in the multihop wireless TCP packet transmissions. Multiple lossy links are modelled in [21] which also consider different proportions between the interference range and transmission range in the spatial reuse property of the wireless channel.

2.2. System Models for TCP Analysis

There are various approaches to modelling TCP in the literature. Some papers have tried to capture the essential TCP dynamics through closed-form expressions. Lakshman and Madhow [22] and Kumar [23] use Markovian analysis to develop a closed-form expression for the throughput of TCP connections by observing the cyclical evolution of the TCP transmission window. The latter work introduces some extensions for several versions of TCP, incorporating such features as coarse timers, fast retransmit and fast recovery. Mathis *et al*. [24] focus on the stochastic behavior of the congestion avoidance mechanism, deriving an expression for the throughput that is then applied to study the behavior of several flavors of TCP sources and queueing techniques. Padhye *et al*. [25] have derived a steady-state model that approximates the throughput of bulk TCP flows as a function of loss rate and round trip time, comparing their estimates with real-life traces of TCP traffic. In [26], instead of trying to establish a closed-form expression for key metrics such as throughput, queueing delay and packet loss, a novel methodology, reciprocal model tuning, combines a Markovian model of a single TCP source in isolation, and the analysis of superposition and interaction of several TCP sources through standard queueing analysis techniques. Notably, some researchers [27,28] have chosen a different approach whereby the observation of "actual" TCP traces is the foundation of empirical models. These efforts imply collecting hours' (days') worth of data, and finding suitable statistical distributions for the observed data.

There are two system models in the literature most similar to our model. The first one is used by Chaskar, Lakshman and Madhow in [29] to analyze TCP performance over a wireless channel with link level error control. There, ACK packets are assumed to arrive in the source node after a constant delay and are never lost. Thus the path of ACKs is omitted (**Figure 2(a)**). The second one is proposed by Lakshman, Madhow and Suter in the same research group to analyze TCP performance with random loss and bidirectional congestion [30]. There, for the first time the path followed by ACKs is explicitly modeled (**Figure 2(b)**). These models, however, cannot be used in our case of multihop wireless networks where the TCP packets and ACKs are contented to use the same wireless channel. Instead of starting with analyzing the TCP protocol with assumptions as in [22-26], our approach is to study the process of how TCP works between a source and destination pair by examining the simulation trace file. The resulting model considers the channel access probability of both the source and the destination (**Figure 3**).

3. System Model

Consider the simple adhoc wireless network scenario shown in **Figure 1** nodes form a *string* with length *N*. Each node has the same transmission radius, the same carrier sense radius, and the same interference radius. One TCP connection is run from node 0 to node *N* crossing all the intermediate nodes in the *string*.

We model the communication process of the TCP connection as in **Figure 3** after analyzing the trace files from simulations carefully. The source and destination nodes both have a First In First Out (FIFO) forward buffer of size. The source has infinite data to send, so that TCP packets are always of the maximum packet size. For each packet that is received by the destination, a cumulative acknowledgment (ACK) is generated which can be modeled as containing the next expected segment number. TCP packet service rate is U_t, the typical number of TCP packets the system serves per unit time when it is constantly busy. ACK packet service rate is U_a, the typical number of ACK packets the system serves per unit time when it is constantly busy. The "unit time" includes TCP/ACK packet transmission delay, propagation delays, processing delays at the nodes from layer to layer and the average MAC layer contention delays. The TCP/ACK packet service rates are affected by the spatial reuse of wireless channel in the *string* topology. As all nodes compete to use the channel, the chance for any node to send packet is determined by the underlying MAC protocol and traffic distribution. Generally, we denote the *average* probability of the source to send a TCP packet as q, and the *average* probability of the destination to send an ACK packet as p. The reason to use the average probability of the node accessing the channel is because the performance metric of interest is the average TCP throughput.

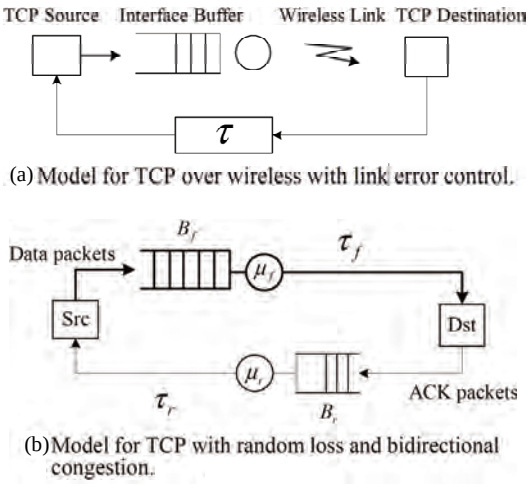


Figure 2. Reference system models for TCP analysis.

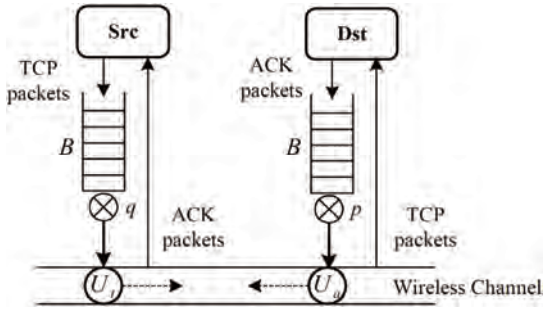


Figure 3. System model of TCP in wireless ad-hoc networks.

The system model captures the unique communication features of TCP in adhoc wireless networks including link layer channel contention and channel spatial reuse. It is similar to but different from system models used in analyzing TCP performance in other kind of networks [29,30].

4. Markov Chain Modelling

We use Markov chain modelling to analyze the proposed system model. Firstly, a discrete Markov chain is formed for the case when the source and the destination are single-hop away. The Markov chain is then extended to the general case when the source and the destination are multiple hops away. The assumptions made are as follows:

- There is no random loss of packets due to channel error. The channel appears error-free to the upper layer because of the error coding schemes and link layer retransmission protocols.

- The channel also appears collision-free to the upper layer because the MAC layer protocol is collision avoided, such as MACAW [31].
- The maximum TCP congestion window size is less than the buffer size at each node.
- There is no packet dropping due to buffer overflow, and no packet loss due to link layer contention. This assumption is supported by simulation results demonstrating that packet loss due to buffer overflow is rare and packet loss due to link contention is very low for single TCP over a *string* topology.
- The distribution of inter-service time of TCP packets and ACK packets are exponentially distributed which include packet retransmissions. This assumption is based on the fact that exponential backoff during contention is commonly used in MAC protocols, e.g., IEEE 802.11.

4.1. Single-Hop Case

There are only TCP packets at the source and only ACK packets at the destination. Either the source or the destination sending packets will change the number of the packets queued in their buffers. Ignoring the slow start phase which is a small part of the data transmission, the sum of the TCP packets at the source and the ACK packets at the destination equals to the TCP congestion window size in packets. As it is assumed that no packet loss exists due to channel error, buffer overflow, or link contention, the TCP connection thus increases its congestion window size to the maximum value and stabilizes there.

Discrete-time Markov chain is applied to analyze TCP performance here. Let us focus attention at times 0, δ, 2δ …, $k\delta$,… as in [32] (pages 162-173), where δ is a small positive number. Let F denote as the number of TCP packets in the source node buffer at time $k\delta$. F thus is a Markov chain on the state-space $\{m : W \geq m \geq 0\}$, where W the maximum TCP congestion window size. The transition diagram of the Markov chain is shown in **Figure 4**.

When there are W TCP packets in the source node, *i.e.*, the first state in **Figure 4**, the destination node does not have ACK packets to send. Therefore, the source node catches the wireless channel for sure. The resulting transition probability from the state of $F = W$ to $F = W - 1$ is δU_t, where U_t is the service rate of TCP packets and the inter-service time is assumed to be exponentially distributed. Similarly, the transition probability from the state of $F = 0$ to $F = 1$ is δU_a Where U_a is the service rate of ACK packets and the inter-service time is assumed to be exponentially distributed.

When there are *F*, where $W - 1 \geq F \geq 1$ TCP packets in the source, there are $W - F$ ACK packets in the destination node. The source node and destination node compete with each other to use the channel. The source node has average probability of q to access the channel (**Figure 3**), the transition probability from the state of $F = W - i$ to $F = W - i - 1$ is thus $q\delta U_t$, where $1 \leq i \leq W - 1$. Similarly, the transition probability from the state of $F = W - i$ to $F = W - i + 1$ is $p\delta U_a$, where p is the average probability of the destination to access the channel.

4.2. Multiple-Hop Case

When the source and destination are multiple hops away, an accurate Markov model at times $k\delta$ would be on the state space $\{(F_0, A_0), (F_1, A_1), ..., (F_N, A_N)\}$, where (F_i, A_i) are the numbers of TCP and ACK packets at node i. This multi-dimensional Markov chain modelling considers the number of TCP packets and ACK packets in the source, the destination and the intermediate nodes along the *string* topology. Unfortunately, such a modelling is difficult to tackle. For example, even when the *string* topology is 2-hops long, and the TCP maximum congestion window size is 2 packets, the Markov chain already has 11 states as shown in **Figure 5**, and the number of states increases quickly with the increase of the TCP maximum congestion window size and the length of the *string*. This happens due to the sharing of wireless channel between nodes. When two nodes within interference range both have packets to send, there are two possible next states. The more states the Markov chain has, the more difficult it is to solve. A heuristic alternative is applied as follows.

The more states the Markov chain has, the more paths are available in the transition diagram for a TCP packet

$$Q = \begin{bmatrix} 1-\delta U & \delta U & 0 & 0 & \cdots & 0 \\ p\delta U & 1-p\delta U-q\delta U & q\delta U & 0 & \cdots & 0 \\ 0 & p\delta U & 1-p\delta U-q\delta U & q\delta U & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 & p\delta U & 1-p\delta U-q\delta U & q\delta U \\ 0 & 0 & \cdots & 0 & 0 & \delta U & 1-\delta U \end{bmatrix} \quad (1)$$

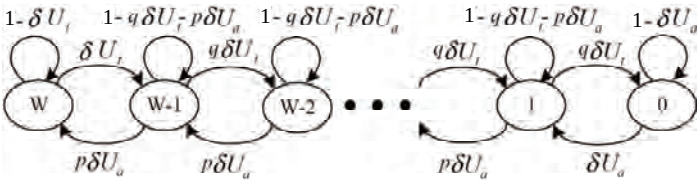


Figure 4. Transition diagram when the number of hop is 1 and TCP maximum congestion window size is *W*.

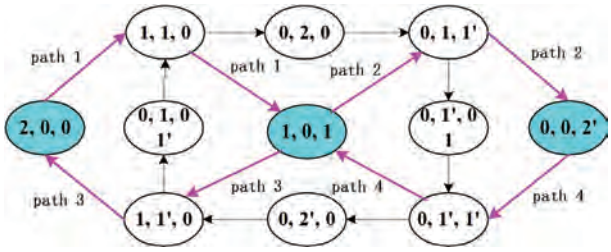


Figure 5. Transition diagram when the number of hop is 2 and the maximum TCP congestion window size is 2 packets. The number of TCP packets at nodes are shown as 1 or 2; the number of ACK packets are shown as 1' or 2'. (1, 1', 0) denotes that there is 1 TCP packet at the source node, 1 ACK packet at the middle node, and none packet at the destination node.

to be transmitted from the source to the destination, or for an ACK packet from the destination to the source. Whatever states the packet goes through in between, it finally passes all the nodes of the *string*. Let us denote the path for a TCP packet to be transmitted from the source to the destination as a *forward* path; and the path for an ACK packet to be transmitted from the destination to the source as a *backward* path. A transmission round of TCP is composed of one *forward* path and one *backward* path.

The average utilization of the *string* topology should be the average of all the transmission rounds. Intuitively, ***for each forward path, a corresponding complementary backward path exists which makes this round of transmission to be exactly the average value***. With this understanding, the analysis can be sought out based on a specific transmission round with a pair of complementary *forward* and *backward* paths only. All the TCP packets go through the specific *forward* path, and all the ACK packets go through the specific *backward* path in such analysis.

A simple transmission round is chosen as follows: for the *forward* path, when the source node gets chance to start sending a TCP packet, the packet passes all the intermediate nodes continuously until it reaches the destination; the *forward* path can be path 1 or path 2 in **Figure 5**. Likewise, for the *backward* path, when the destination node gets chance to start sending an ACK packet, the ACK packet passes all the intermediate nodes continuously until it reaches the source; the *backward* path can be path 3 or path 4 in **Figure 5**. These paths are *complementary*.

In the above specific transmission round, once one packet in the source node is sent out, the number of packets in the buffer of source node decreases by 1. After the packet goes continuously to the destination, the destination absorbs the TCP packet and generates an ACK packet. The reverse process is the same. Consequently, the changing of number of packets in the source and the destination is the same as in the single-hop scenario. The Markov modelling of this complete transmission round is thus exactly the same as in **Figure 4** when we only consider the marked paths and states in **Figure 5**.

In the following subsections, we are going to analyze the discrete Markov chain in **Figure 4** for both the single-hop and multiple-hop cases.

5. Analysis

5.1. TCP Throughput

The transition matrix Q of the Markov chain in **Figure 4** is listed in (1). Let $\pi = (\pi_{\mathrm{w}}, \pi_{\mathrm{w}-1}, \ldots \pi_0)$ denote the stationary probability distribution of the Markov chain, we have $\pi = \pi \times Q$ at steady state. From this it is derived that

$$\begin{aligned}
\pi_{\mathrm{w}-1} &= \frac{1}{q}\left(\frac{q}{p}\rho\right)\pi_{\mathrm{w}} \\
\pi_{\mathrm{w}-2} &= \frac{1}{q}\left(\frac{q}{p}\rho\right)^2\pi_{\mathrm{w}} \\
&\cdots \quad \cdots \\
\pi_{\mathrm{w}-i} &= \frac{1}{q}\left(\frac{q}{p}\rho\right)^i\pi_{\mathrm{w}}, \quad 1 \le i \le W-1 \\
&\cdots \quad \cdots \\
\pi_0 &= \frac{p}{q}\left(\frac{q}{p}\rho\right)^W\pi_{\mathrm{w}}
\end{aligned} \tag{2}$$

where ρ is the ratio of U_t to U_a, *i.e.*, $\rho = \dfrac{U_t}{U_a}$.

From $\sum_{i=0}^{W}\pi_i = 1$, it is further derived that

$$\pi_{\mathrm{w}} = \frac{1}{1+\sum_{i=1}^{W-1}\frac{1}{q}(\frac{q}{p}\rho)^i + \frac{p}{q}(\frac{q}{p}\rho)^W} \tag{3}$$

From the transition diagram, we denote the TCP throughput of state i by λ_i and the average throughput of TCP packets by λ. Thus $\lambda = \sum_{i=0}^{W} \pi_i \lambda_i$, where $\lambda_w = U_t, \lambda_i = qU_t$ when $W-1 \geq i \geq 1$, and $\lambda_0 = 0$. Therefore,

$$\lambda = U_t \pi_w \frac{1-(\frac{q}{p}\rho)^w}{1-\frac{q}{p}\rho} \tag{4}$$

The source and destination nodes are located at the two ends of the *string* topology with $N-1$ intervening nodes. The positions of the source and destination are the same, and they also have the same number of packets to send since we assume that one ACK is generated for each TCP packet. Thus, the source and destination nodes have the same *average* probability to access the channel successfully, *i.e.*, $p = q$. (3) and (4) can then be simplified as:

$$\pi_w = \frac{1-\rho}{1-\rho^{W+1}+(\frac{1}{q}-1)\rho(1-\rho^{W-1})} \tag{5}$$

$$\lambda = U_t \frac{1-\rho^W}{(1-\rho^{W+1})+(\frac{1}{q}-1)\rho(1-\rho^{W-1})} \tag{6}$$

ρ is the ratio of TCP packet service rate to the ACK packet service rate. Since TCP packets and ACK packets are transmitted at the same channel, and they travel the same number of hops between the source and the destination, ρ is always approximately equal to the ratio of the TCP packet service rate to the ACK packet service rate in the case when there is only one hop between the source and the destination. This is further approximately equal to the ratio of the ACK packet size to the TCP packet size. The TCP packet (say, 1460 bytes) is normally much larger than the ACK packet (say 40 bytes), ρ is thus much smaller than 1 (say, around 40/1460 = 0.027), *i.e.*, $\rho \ll 1$. The maximum TCP congestion window size W pkts is an integer and when it is big enough, it is expected that

$$1-\rho^W \approx 1-\rho^{W-1} \approx 1-\rho^{W+1} \tag{7}$$

Thus from (6) and (7), the *average* TCP throughput is derived as:

$$\lambda = U_t \frac{1}{1+(\frac{1}{q}-1)\rho} \tag{8}$$

In the above analysis, it is shown clearly that the ratio of ACK packet size to TCP packet size being much less than 1 is required to carry out the approximation.

5.2. TCP Service Rate

We have defined U_t as the service rate of TCP packets seen q as the average probability for the source node to access the wireless channel. Their values, however, change with the length of the *string*, *i.e.*, the number of hops that the TCP session crosses from the source to the destination (**Figure 1**). This is because of the effect of global channel spatial reuse and local channel contention in adhoc wireless networks. By definition, service rate is the "typical number of customers the system serves per unit time when it is constantly busy" [32] (page 152). When the number of hops is N, let the average number of TCP packets being transmitted (served) in the system be $\overline{I_N}$ and the service time to transmit a packet be T_N, the definition of service rate gives:

$$U_{tN} = \frac{\overline{I_N}}{T_N} \tag{9}$$

Let q_N be the average probability that the source node accesses the wireless channel when the number of hops is N. As N changes, $\overline{I_N}$, T_N, and q_N also change. In the following, we evaluate the values of $\overline{I_N}$, T_N, and q_N as N changes.

5.2.1. N is 1, 2, 3 or 4

When N is 1, $\overline{I_1} = 1$ pkt, $q_1 = \frac{1}{2}$. Let $T_1 = T$, where T is the service time to transmit a packet over one hop when the source and destination nodes are one hop away.

When N is 2, each TCP packet travels two wireless links to reach the destination. Therefore, approximately $T_2 = 2T$. During the transmission from node 0 to node 1, and then from node 1 to node 2, only 1 TCP packet can be transmitted in the system without collision. Therefore $\overline{I_2} = 1$ *pkt*. In the *string* topology, the source node only forwards TCP packet, the destination node only forwards ACK packets but the middle node forwards both TCP and ACK packets. Since one ACK packet is assumed for each TCP packet, the middle node sends out packets twice as much as the source and destination node. As a result, the average probability of the middle node to access the channel successfully is twice as much as that of the source and destination node. In addition, the summation of the average probability of all nodes to access the channel is 1. Derivation from these relationships gives $q_2 = \frac{1}{4}$. When N is 3, by similar analysis we have $\overline{I_3} = 1$ pkt, $T_3 = 3T$, $q_3 = \frac{1}{6}$.

When N is 4, it looks that node 0 and node 3 could

send packets concurrently without collision. However, although node 3 is outside of node 1's transmission range, it is within the carrier sensing range and interference range of node 1. Node 3 is thus a potential hidden terminal of the transmission pair node 0 to node 3. Consequently, only 1 TCP packet can be transmitted in the system without collision. This analysis is consistent with that in [3]. Further derivation gives $\overline{I_4} = 1$ pkt, $T_4 = 4T$, $q_4 = \frac{1}{8}$.

We summarize the value for $\overline{I_N}$, $\overline{q_N}$ as shown in **Table 1**.

5.2.2. N is 5, 6, 7, or 8

When N is 5, each TCP packet crosses five wireless hops to reach the destination. Among the five hops from node 0 to node 5, the 1st hop (*i.e.*, from node 0 to node 1) and the 5th hop (*i.e.*, from node 4 to node 5) can be utilized at the same time as shown in **Figure 6**. But when the 2nd , 3rd or 4th hop isused, only that hop can be used without collision.

Assuming that the system is utilized fully, the number of packets in the system is:

$$I_5 = \begin{cases} 2 \text{ pkts} & \text{when the 1st and 5th hops are used} \\ 1 \text{ pkt} & \text{when the 2nd , 3rd or 4th hops are used} \end{cases}$$

Since each hop has equal opportunity to be utilized, the average number of packets in the system is: $\overline{I_3} = 2 \times \frac{2}{5} + 1 \times \frac{3}{5}$ pkts.

Table 1. N is 1, 2, 3 or 4

N	T_N	$\overline{I_N}$	q_N
1	T	1 pkt	$\frac{1}{2}$
2	2T	1 pkt	$\frac{1}{4}$
3	3T	1 pkt	$\frac{1}{6}$
4	4T	1 pkt	$\frac{1}{8}$

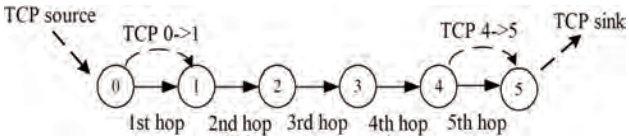


Figure 6. Two TCP packets are transmitted together when N = 5.

Although there are six nodes, however, node 0 competes to use the channel locally with nodes 1, 2, 3 and 4 only. The probability for node 0 to access the channel can therefore be taken as the same as when there are 4 hops. Thus $q_5 = \frac{1}{8}$, $T_5 = 5T$.

When N is 6, each TCP packet crosses six wireless links to reach the destinatiopn. **Figure 7** shows the scenarios when two of the six wireless links are utilized at the same time. The number of packets in the system is:

$$I_6 = \begin{cases} 2 \text{ pkts} & \text{when the (1st or 2nd) and (5th or 6th) hops are used} \\ 1 \text{ pkt} & \text{when the 3rd or 4th hop is used} \end{cases}$$

The average number of packets in the system is: $\overline{I_6} = 2 \times \frac{4}{6} + 1 \times \frac{2}{6}$ pkts. It is also easy to get $q_6 = \frac{1}{8}$, $T_6 = 6T$.

Similarly, when N is 7, we get

$$I_7 = \begin{cases} 2 \text{ pkts} & \text{when the (1st , 2nd or 3rd) and (5th , 6th or 7th) hops are used} \\ 1 \text{ pkt} & \text{when the 4th hop is used} \end{cases}$$

$$\overline{I_7} = 2 \times \frac{6}{7} + 1 \times \frac{1}{7} \text{ pkts}, \; q_7 = \frac{1}{8}, \; T_7 = 7T$$

When N is 8, $I_8 = \overline{I_8} = 2$ pkts , when the (1st, 2nd, 3rd or 4th) and (5th, 6th, 7th or 8th) hops are occupied, and $q_8 = \frac{1}{8}, T_8 = 8T$.

5.2.3. Generalization of TCP Service Rate

Let $N = 4M + j$, where $M \geq 1$, $j \subseteq \{1,2,3,4\}$, the generalization of the above analysis is as follows.

When N is $4M$, we have $I_{4M} = \overline{I_{4M}} = M$ pkts.

When N is $4M+1$,

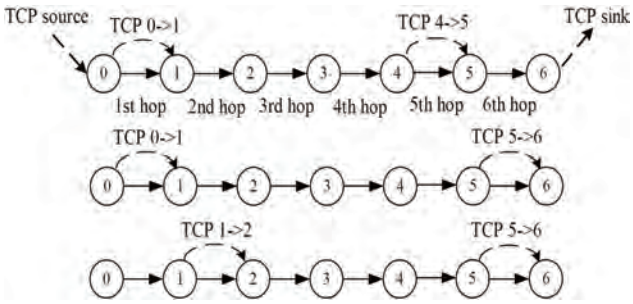


Figure 7. Two TCP packets are transmitted together when N = 6.

$$I_{4M+1}=\begin{cases} M+1 \text{ pkts} & \text{when the } 1^{st} \text{ and } 5^{th} \text{ and } \ldots \\ & (4M+1)^{th} \text{ hops are occupied} \\ M \text{ pkt} & \text{when the } (2^{nd}, 3^{rd} \text{ or } 4^{th}) \text{ and } \ldots \\ & (6^{th}, 7^{th} \text{ or } 8^{th}) \text{ and } \ldots \\ & \left((4M\text{-}2)^{th}, (4M-1)^{th} \text{ or}\right. \\ & \left.(4M)^{th}\right) \text{ hops are occupied} \end{cases}$$

$$\overline{I_{4M+1}}=(M+1)\times\frac{M+1}{4M+1}+M\times\frac{(4M+1)-(M+1)}{4M+1}\text{pkts}$$

Similar derivation gives:

When N is $4M+2$,

$$I_{4M+2}=\begin{cases} M+1 \text{ pkts} & \text{when the } (1^{st} \text{ or } 2^{nd}) \text{ and} \\ & (5^{th} \text{ or } 6^{th}) \text{and } \ldots \text{ and} \\ & (4M-3)^{th} \text{ or } (4M-2)^{th} \\ & \text{hops are occupied} \\ M \text{ pkt} & \text{when the } (3^{rd} \text{ or } 4^{th}) \text{ and} \\ & (7^{th} \text{ or } 8^{th}) \text{ and } \ldots \text{and} \\ & \left((4M\text{-}1)^{th} \text{ or } (4M)^{th}\right) \\ & \text{hops are occupied} \end{cases}$$

$$\overline{I_{4M+2}}=(M+1)\times\frac{2(M+1)}{4M+2}+M\times\frac{(4M+2)-2(M+1)}{4M+2}\text{pkts}$$

When N is $4M+3$,

$$I_{4M+3}=\begin{cases} M+1 \text{ pkts} & \text{when the } (1^{st}, 2^{nd} \text{ or } 3^{rd}) \text{ and} \\ & (5^{th}, 6^{th} \text{ or } 7^{th}) \text{and } \ldots \text{ and} \\ & (4M-3)^{th} \text{ or } (4M-2)^{th} \text{or} \\ & (4M-1)^{th} \text{hops are occupied} \\ M \text{ pkts} & \text{when the } 4^{th} \textit{ and } 8^{th} \text{ and } \ldots \\ & \text{and } (4M)^{th} \text{ hops are occupied} \end{cases}$$

$$\overline{I_{4M+3}}=(M+1)\times\frac{3(M+1)}{4M+3}+M\times\frac{(4M+3)-3(M+1)}{4M+3}\text{pkts}$$

Finally, it is summarized that:

$$\overline{I_{4M+j}}=(M+1)\times\frac{j(M+1)}{4M+j}+M\times\frac{(4M+j)-j(M+1)}{4M+j}\text{pkts} \quad (10)$$

Where $M\geq 1,\ j\subseteq\{1,2,3,4\}$.

Combining (10) and the analysis when N is 1, 2, 3 or 4 in **Table 1**, we have the average number of packets in the system as:

$$\overline{I_{4M+j}}=\frac{j(M+1)^2+(4-j)M^2}{4M+j} \quad (11)$$

Where $M\geq 0,\ j\subseteq\{1,2,3,4\}$.

The service time to transmit a packet is generalized as $T_{4M+j}=(4M+j)\times T$. From (9), the TCP service rate is derived as:

$$U_{t(4M+j)}=\frac{j(M+1)^2+(4-j)M^2}{(4M+j)^2}\times\frac{1}{T} \quad (12)$$

where $M\geq 0,\ j\subseteq\{1,2,3,4\}$.

The average probability of the source to access the channel is:

$$q_{4M+j}=\begin{cases}\dfrac{1}{2j} & \text{when } M=0 \text{ and } j\subseteq\{1,2,3,4\} \\ \dfrac{1}{8} & \text{when } M\geq 1 \text{ and } j\subseteq\{1,2,3,4\}\end{cases} \quad (13)$$

5.3. Generalization of TCP Throughput

Substituting (12) and (13) into (8), the TCP throughput becomes:

$$\lambda_{4M+j}=\begin{cases}\dfrac{1}{j(1+(2j-1)\rho)}\times\dfrac{1}{T} \\ \textit{when } M=0 \textit{ and } j\subseteq\{1,2,3,4\} \\ \dfrac{j(M+1)^2+(4-j)M^2}{(4M+j)^2}\times\dfrac{1}{1+7\rho}\times\dfrac{1}{T} \\ \textit{when } M\geq 1 \textit{ and } j\subseteq\{1,2,3,4\}\end{cases} \quad (14)$$

When M goes to infinity, $N=4M+j$ goes to infinity, we have

$$\lim_{M\to\infty}\lambda_{4M+j}=\frac{1}{4}\times\frac{1}{T}\times\frac{1}{1+7\rho} \quad (15)$$

Equation (14) shows that the TCP throughput is independent of the maximum TCP congestion window size W; instead it is decided by (a) $4M+j$, the value of the number of hops of the *string* topology, (b) ρ, the ratio of service rate of TCP packet to ACK packet, and (c) T, the time needed for a TCP packet to be transmitted over a single hop. Furthermore, (15) illustrates that as the number of hops increases and goes to infinity, TCP throughput converges to a constant value.

6. Evaluations

The analytical results are verified by the comparison of

simulation results in *ns*2 and numerial results in *Matlab*.

In simulations, $N+1$ nodes forma *string* (**Figure 1**) with adjacent nodes 200 m apart. All nodes communicate with identical, half-duplex wireless radios which have a bandwidth of 2 Mbps and a nominal transmission radius of 250 m. Nodes have carrier sense radius of 550 m and interference radius of 550 m. They are configured with the Dynamic Source Routing (DSR) protocol. One TCP Reno session is introduced from node 0 (TCP source) to node N (TCP destination) to transfer FTP bulk data, crossing N hops. TCP and ACK packets size are of 1460 bytes and 40 bytes respectively. Simulations were run with various numbers of hops and various maximum TCP congestion window sizes. One simulation with the same number of hops and the same maximum TCP congestion window size was run for five times, each for 300 secs, and the overall throughput was measured from 50 secs to 250 secs. During the time the throughput is quite stable, but the average of five simulations was used as the simulation result.

To get the numerical results in *Matlab*, two parameters are needed:

- ρ : the ratio of service rate of TCP packet to ACK packet. Here $\rho = \frac{40 \text{ bytes}}{1460 \text{ bytes}} = 0.027$, *i.e.*, the ratio of ACK packet size to TCP packet size.
- $\frac{1}{T}$: the average transmission rate of one packet over a single hop link, where T is defined as the average transmission time of one packet over one hop link. However, $\frac{1}{T}$ cannot be easily assigned a value considering the bandwidth used on the channel contention, MAC control packets exchange, etc. Instead, we decide the value of $\frac{1}{T}$ with the help of simulation results so that the numarical results could best match the simulation results

Figure 8 shows the comparison of simulation and numerical results of the TCP throughput as the number of hops changes from 1 to 11. Simulation results are presented when the maximum TCP congestion window size is 8 packets. Numerical results are presented with two values of $\frac{1}{T}$, so there are three curves in **Figure 8**: the simulation results, the numerical results when $\frac{1}{T} = 1300$ kbps , and the numerical results when $\frac{1}{T} = 900$ kbps .

It is observed that the numerical results with $\frac{1}{T} =$ 1300 kbps match the simulation results well at $N = 1,$ $2,3$ or 4 hops , and the numerical results with $\frac{1}{T} =$ 900 kbps match the simulation results well at $N \geq 5$. The reason is as follows. When $N = 1,2,3$ or 4 , no link is used simultaneously due to the hidden terminal problem. When $N \geq 5$, there are links used simultaneously because of channel spatial resue. However, although channel spatial reuse helps improve the overall system utilization, it has higher local contention overhead. This results in a lower payload transmission rate. Therefore, $\frac{1}{T}$ is chosen as 900 kbps when $N \geq 5$ and 1300 kbps when $N = 1,2,3$ or 4 respectively. In **Figure 8**, TCP throughput is shown to decrease rapidly when the number of hops increases from 1, and stabilizes when the number of hops becomes large. This is in line with the observation we described in the introduction and verifies the analysis in (13) and (14).

Next, we look at the trend of TCP throughput with varying maximum window size. Simulation results of TCP throughput is shown in **Figure 9** with varying number of hops (from 1 to 11 hops) and TCP maximum window size (from 1 to 12 packets). When the number of hops is fixed, TCP throughput is kept constant independent of the maximum window size, *i.e.*, the throughput curve is almost a straight line parallel to the x-axis. This is consistent with (14) where the TCP throughput (λ)is not a function of TCP maximum window size (W). This result, however, looks inconsistent with the claim in [3] that "given a specific network topology and flow patterns, there exists a TCP window size W^*, at which TCP achieves best throughput via improved spatial channel

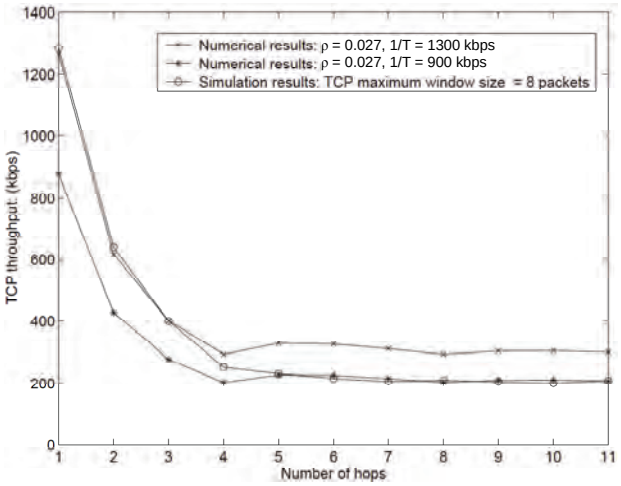


Figure 8. Comparison of simulation and numerical results: TCP throughput with varying number of hops. Simulations run for 300 seconds.

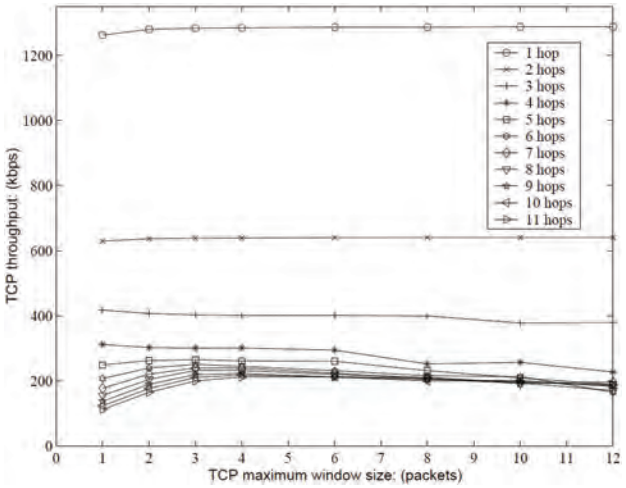


Figure 9. Simulation result: TCP throughput with varying number of hops and maximum window size, simulations run for 300 seconds.

reuse". However, a close look at **Figures 2(a)** and **(b)** in [3] reveals that TCP throughput at the "optimal" TCP window size is not much higher than those at non-optimal TCP window sizes; furthermore, TCP throughput is a constant at most values of TCP window size. This observation is consistent with our analysis here.

We notice a phenomenon in the left lower part of the figure. When the number of hops is greater than 5 and the maximum window size is 1 or 2 packets, TCP throughput is less than the constant value when the maximum window size is greater or equal to 3 packets. This is due to the fact that we assume the system is always busy, *i.e.*, all the available links are fully used at every moment. However when the window size is 1 packet, the whole system goes into a stop-and-wait process and the source will only issue a TCP packet after it receives an ACK from the destination. Therefore the system is not fully occupied. The achieved throughput is thus less than the fully occupied value. The same reason explains the case when the maximum window size is 2 packets but the *string* topology can support a maximum of 3 packets when the number of hops is 9, 10, or 11.

The above analytical result is interesting as it shows that the setting of TCP congestion window size does not matter that much in wireless multihop networks, which is different from the scenario in wired networks where TCP performance is largely affected by its congestion window size. However, our analysis considers single a TCP connection over static chain without interfering background traffic. Further evaluations are necessary.

7. Conclusions

This study takes a step to theoretically analyze TCP performance in adhoc wireless networks. The analysis is based on a *string* topology containing N hops. A system model for analyzing TCP performance in adhoc wireless networks is proposed, which considers packet buffering, contention of nodes for access to wireless channel and spatial reuse of wireless channel. Markov chain modelling is applied to analyze the system model. Analytical results are presented to show that when the number of hops that the TCP session crosses is fixed, the TCP throughput is independent of the TCP congestion window size. When the number of hops increases from one, the TCP throughput decreases first, and then stabilizes when the number of hops becomes large. The analysis is validated by comparing the numerical results with the simulation results.

8. References

[1] G. Holland and N. H. Vaidya, "Analysis of TCP Performance over Mobile Ad Hoc Networks," *Proceedings of IEEE/ACM MobiCom*'99, Seattle, August 1999, pp. 219- 230.

[2] M. Gerla, R. Bagrodia, L. Zhang, K. Tang and L. Wang, "TCP over Wireless Multi-Hop Protocols: Simulation and Experiments," *Proceedings of IEEE ICC*'99, Vancouver, Vol. 2, June 1999, pp. 1089-1094.

[3] Z. Fu, H. Luo, P. Zerfos, S. Lu, L. Zhang and M. Gerla, "The Impact of Multihop Wireless Channel on TCP Performance," *IEEE Transactions on Mobile Computing*, Vol. 4, No. 2, March 2005, pp. 209-221.

[4] H. Xiao, K. G. Seah, A. Lo and K. C. Chua, "On Service Differentiation in Multihop Wireless Networks," *ITC Specialist Seminar on Mobile Systems and Mobility*, Lillehammer, March 2000, pp. 1-12.

[5] X. Chen, H. Zhai, J. Wang and Y. Fang, "TCP Performance over Mobile Ad Hoc Networks," *Journal of Electrical & Computer Engineering*, Vol. 29, No. 1/2, 2004, pp. 129-134.

[6] K. Chandran, S. Raghunathan, S. Venkatesan and R. Prakash, "A Feedback-Based Scheme for Improving TCP Performance in Ad Hoc Wireless Networks," *IEEE Personal Communications*, Vol. 8, No. 1, 2001, pp. 34-39.

[7] Z. Fu, X. Meng and S. Lu, "A Transport Protocol for Supporting Multimedia Streaming in Mobile Ad Hoc Networks," *IEEE Journal on Selected Areas in Communications*, Vol. 21, No. 10, 2003, pp. 1615-1626.

[8] Z. Fu, P. Zerfos, H. Luo, S. Lu, L. Zhang and M. Gerla, "The Impact of Multihop Wireless Channel on TCP Performance," *IEEE INFOCOM*'03, San Francisco, Vol. 3, March 2003, pp. 1744-1753.

[9] X. Yu, "Improving TCP Performance over Mobile Ad Hoc Networks by Exploiting Cross-Layer Information Awareness," *Proceedings of IEEE/ACM MobiCom'*04, Philadelphia, September 2004, pp. 231-244.

[10] R. Oliveira and T. Braun, "A Smart TCP Acknowledgment Approach for Multihop Wireless Networks," *IEEE Transactions on Mobile Computing*, Vol. 6, No. 2, 2007,

pp. 192-205.

[11] X. Li, P. Y. Kong and K. C. Chua, "DTPA: A Reliable Datagram Transport Protocol over Ad Hoc Networks," *IEEE Transactions on Mobile Computing*, Vol. 7, No. 10, 2008, pp. 1285-1294.

[12] F. Wang and Y. Zhang, "Improving TCP Performance over Mobile Ad Hoc Networks with Out-Of-Order Detection and Response," *Proceedings of IEEE/ACM MobiHoc*'02, Lausanne, June 2002, pp. 217-225.

[13] K. Chen, Y. Xue and K. Nahrstedt, "On Setting TCP's Congestion Window Limit in Mobile Ad Hoc Networks," *Proceedings of IEEE ICC*'03, Anchorage, Vol. 2, May 2003, pp. 1080-1084.

[14] K. Xu, M. Gerla, L. Qi and Y. Shu, "Enhancing TCP Fairness in Ad Hoc Wireless Networks Using Neighborhood RED," *Proceedings of IEEE/ACM MobiCom*'03, San Diego, September 2003, pp. 16-28.

[15] L. Yang, W. K. G. Seah and Q. Yin, "Improving fairness among TCP flows crossing wireless ad hoc and wired networks," *Proceedings of IEEE/ACM MobiHoc*'03, New York, June 2003, pp. 57-63.

[16] V. Anantharaman, S. J. Park, K. Sundaresan and R. Sivakumar, "TCP Performance over Mobile Ad Hoc Networks: A Quantitative Study," *Wireless Communications and Mobile Computing*, Vol. 4, No. 2, 2004, pp. 203-222.

[17] K. Nahm, A. Helmy and C.-C. Jay Kuo, "TCP over Multihop 802.11 Networks: Issues and Performance Enhancement," *Proceedings of MobiHoc*'05, New York, 2005, pp. 277-287.

[18] F. Klemm, Z. Ye, S. V. Krishnamurthy and S. K. Tripathi, "Improving TCP performance in ad hoc networks using signal strength based link management," *Ad Hoc Networks*, Vol. 3, No. 2, 2005, pp. 175-191.

[19] H. Xiao, K. C. Chua, K. G. Seah and A. Lo, "A Quantitative Analysis of TCP Performance over Wireless Multihop Networks," *Proceedings of IEEE VTC*2001-*Fall*, Atlantic, Vol. 2, May 2001, pp. 610-614.

[20] H. Xiao, K. C. Chua, J. A.Malcolm and Y. Zhang, "Theoretical Analysis of TCP Throughput in Ad Hoc Wireless Networks," *IEEE Global Telecommunications Conference*, Atlanta, Vol. 5, 2005, pp. 2714-2719.

[21] X. Li, P. Y. Kong and K. C. Chua, "TCP Performance in IEEE 802.11-Based Ad Hoc Networks with Multiple Wireless Lossy Links," *IEEE Transactions on Mobile Computing*, Vol. 6, No. 12, 2007, pp. 1329-1342.

[22] T. V. Lakeshman and U. Madhow, "The Performance of TCP/IP for Networks with High Bandwidth-Delay Products and Random Loss," *IEEE/ACM Transactions on Networking*, Vol. 3, No. 3, June 1997, pp. 336-350.

[23] A. Kumar, "Comparative Performance Analysis of Versions of TCP in a Local Network with a Lossy Link," *IEEE/ACM Transactions on Networking*, Vol. 6, No. 4, August 1998, pp. 485-498.

[24] M. Mathis, J. Semske, J. Mahdavi and T. Ott, "The Macroscopic Behavior of the TCP Congestion Avoidance Algorithm," *Computer Communication Review*, Vol. 27, No. 3, July 1997, pp. 67-82.

[25] J. Padhye, V. Firoiu, D. Towsley and J. Kurose, "Modeling TCP Throughput: A Simple Model and its Empirical Validation," *ACM Computer Communication Review*, Vol. 28, No. 4, September 1998, pp. 303-314.

[26] C. Casetti and M. Meo, "A New Approach To Model The Stationary Behavior of TCP Connections," *Proceedings of IEEE INFOCOM* 2000, Tel-Aviv, Vol. 1, March 2000, pp. 367-375.

[27] V. Paxson and S. Floyd, "Empirically-Derived Analytic Models of Wide-Area TCP Connections," *IEEE/ACM Transactions on Networking*, Vol. 2, No. 4, August 1994, pp. 316-336.

[28] V. Paxson and S. Floyd, "Wide-area Traffic: The Failure of Poisson Modeling," *ACM Computer Communication Review*, Vol. 24, No. 4, October 1994, pp. 257-268.

[29] H. M. Chaskar, T. V. Lakshman and U. Madhow, "TCP over Wireless with Link Level Error Control: Analysis and Design Methodology," *IEEE/ACM Transactions on Networking*, Vol. 7, No. 5, October 1999, pp. 605-615.

[30] T. V. Lakshman, U. Madhow and B. Suter, "TCP/IP Performance with Random Loss and Bidirectional Congestion," *IEEE/ACM Transactions on Networking*, Vol. 8, No. 5, October 2000, pp. 541-555.

[31] V. Bharghavan, A. J. Demers, S. Shenker and L. Zhang, "MACAW: A Media Access Protocol for Wireless LANs," *ACM SIGCOMM*, Vol. 24, No. 4, 1994, pp. 212-225.

[32] D. Bertsekas and R. Gallager, "Data Networks," 2nd Edition, Prentice Hall, London, 1992.

Wireless Sensor Network, **2010, 2, 504-511**
doi:10.4236/wsn.2010.27062 Published Online July 2010 (http://www.SciRP.org/journal/wsn)

Scientific Research

Packet Compression Ratio Dependent Spanning Tree for Convergecast

Changjin Suh, Jisoo Shin
Department of Computing, Soongsil University, Seoul, Korea
E-mail: *cjsuh@ssu.ac.kr, jsshin@netwarking.com*
Received April 12, 2010*; revised May* 4, 2010*; accepted May* 6, 2010

Abstract

A convergecast is a popular routing in sensor networks. It periodically forwards collected data at every sensor node along a configured routing path to the outside of a sensor network via the base station (BS). To extend the lifetime of energy-limited sensor networks, many previous researches proposed schemes for data compression. However, few researches investigated the relation between packet compression ratio and spanning trees. We propose packet Compression ratio dependent Spanning Tree (CST) which can provide effective routing paths in terms of the tree length for all ranges of compression ratio *f*. CST is equivalent to the Shortest Path spanning Tree (SPT) which is optimum in the case of no-compression ($f = 0$) and is equivalent to the Minimum Spanning Tree (MST) in the case of full-compression ($f = 1$). CST outperforms SPT and MST for any range of f ($0 < f < 1$). Through simulation we show CST provides shorter paths than MST and SPT in terms of the tree length by 34.1% and 7.8% respectively. We confirm CST is very useful in convergecasts.

Keywords: Packet Compression, Convergecast, CST, Spanning Tree, Sensor Network

1. Introduction

A sensor network is a distributed wireless network to monitor various conditions of a remote area through an autonomously configured routing path [1-4]. This paper studies packet compression-dependent convergecasts. A convergecast is a type of communication in the reverse direction of a broadcast in which all collected packets are forwarded to a single control node called a base station (BS) [2,5]. To perform convergecast effectively, each node forwards its packets to its neighbor once and only once. If we trace these packet routes, we can build up a spanning tree. A convergecast is very popular in sensor networks [3,4]. It is common in sensor networks that intermediate nodes try to reduce packet size during forwarding. This is called packet compression. Most previous researches focused on no-compression or full-compression cases for simplicity.

The compression ratio *f* is represented by the relatively decreased packet lengths over all added packet lengths to a unit length at each sensor node [2,6,7]. At no-compression ($f = 0$), a node relays all collected packets without reducing them. At full-compression ($f = 1$), a node generates an *x*-bit packet after merging many received *x*-bit packets and an *x*-bit packet generated at the node.

The Shortest Path spanning Trees (SPTs) and the Minimum Spanning Trees (MSTs) are useful for convergecasts to save energy because transmission energy typically proportional to the square of distance. SPT minimizes the length of paths from the root to all nodes, and MST minimizes the sum of lengths of a set of links which are used to connect all nodes. In terms of packet compression, SPT and MST are optimum in total transmission energy for a convergecast in no-compression and full-compression cases respectively.

There are only a few previous researches about packet compression-related spanning trees. Upadhyayula *et al*. [7] tried to find a spanning tree that minimizes energy and 'time' for a convergecast. Time indicates the number of time slots required for a convergecast in a wireless TDMA environment. To achieve less time, the paper explains how to configure spanning trees and suggests an algorithm for building them. The proposed algorithm tends to prefer balanced trees.

Luo *et al*. proposed Minimum Fusion Steiner Tree (MFST) [8] and Binary Fusion Steiner Tree (BFST) [9]. Both assume that sensor nodes must perform complex calculations spending energy for packet compression. They tried to minimize the sum of transmission energy and packet compression energy. In MFST, packet compression occurs at every intermediate node. In reality,

packet compression is not always beneficial to energy saving if a compression ratio is low or the energy for packet compression is large. BFST proposes a feasibility test of packet compression. If a node passes the test, it compresses packets and transmits the reduced packet. If it fails, it sends packets without packet compression. These researches focused on the packet compression side, leaving transmission oversimplified. Spanning trees were not their main interest. They assumed all transmissions spend a unit of energy regardless of transmission distance. We could not find any packet compression related paper that investigates the interaction between spanning trees and packet compression ratio.

We propose three ideas in this paper. The first one is a packet compression-related metric in defining the tree lengths. Second, we propose a one-time compression model that does not allow infinitely repeated packet compression. The third idea is a packet Compression ratio dependent Spanning Tree (CST) which builds a spanning tree using the one-time compression model. This new type of tree bridges the gap between SPT and MST. CST can be briefly described as a mixture of f times MST and $(1 - f)$ times SPT. CST is equivalent to SPT at no-compression and MST at full-compression.

This paper consists of the following sections. Section 2 defines a new distance that includes SPT, MST and CST, and merges them into a unified tree and defines our problem. Section 3 describes how to build CST. In Section 4, we give simulation results and compare CST with SPT and MST in terms of tree length. In Section 5, we reach the conclusions.

2. Definition of Distance and Our Problem

This section defines distance rules and our problem using the defined terminologies. Subsection 2.1 introduces three metrics that define distances including our proposed metric $d(\cdot)$ which is a function of the compression ratio f. If we use $d(\cdot)$, no-compression ($f = 0$) and full-compression ($f = 1$) network problems are automatically solved because they are incidences of the generic f-compression network problem ($0 \le f \le 1$). Subsection 2.2 defines this paper's main problem, a unified CST problem using $d(\cdot)$ which covers SPT, MST and CST problems.

2.1. Definition of Distance

We first define several terms. We represent the link from a node i to a node j by $l(i, j)$ and the link cost by $C(i, j)$. The kth ancestor node of a node q is defined as $p_k(q)$. $p_1(q)$ is the parent node of q, and $p_0(q)$ is q itself. The root node is BS for a convergecast and is represented by R. For convenience we assume all ancestor nodes of R are also R, providing:

$$p_k(R) = R, \quad k = 0,1,2,.... \tag{1}$$

The distance of a node q is defined as the shortest path from q to R along the configured spanning tree. We use three metrics (distance rules) to define distance: 1) normal distance $\delta(\cdot)$, 2) constant compression distance $\lambda(\cdot)$, and 3) our one-time compression distance $d(\cdot)$. Later definitions commonly assume q is the kth (k = 1, 2, …) descendent node from R.

$\delta(\cdot)$ defined in (2) simply adds costs of all active links to R without considering packet compression. Active links are the set of links that constitutes the routing path.

$$\delta(q) = \sum_{i=1,2,...,k} C(p_{i-1}(q), p_i(q)). \tag{2}$$

We define the constant compression distance as $\lambda(\cdot)$ which is the most general compression model. Upadhyayula *et al*. [7] use $\lambda(\cdot)$. This rule assumes packets are compressed at a constant ratio f in all intermediate nodes. We formulate $\lambda(\cdot)$ as

$$\lambda(q) = \sum_{i=1,2,...,k} (1-f)^{i-1} \cdot C(p_{i-1}(q), p_i(q)). \tag{3}$$

The last distance $d(\cdot)$ we propose is defined as

$$d(q) = \begin{cases} C(q, p_1(q)) & \text{if} \quad p_1(q) = R, \\ C(q, p_1(q)) + (1-f) & \\ \cdot \sum_{i=1,2,...,k} C(p_{i-1}(q), p_i(q)) & \text{otherwise}. \end{cases} \tag{4}$$

In $d(q)$, packets are compressed with the compression ratio f only once at the parent node along the path to R. **Figure 1** shows the sum of packet lengths transmitted at each link $l(i, j)$. A node q generates an m_q-byte packet and forwards it to R along the spanning tree. Although this paper assumes the lengths of the generated packets are identical, we use m_q for clear understanding. Note all packet lengths can be represented by the linear equation of f : $(a_{ij} \cdot f + b_{ij})$. This property is utilized to calculate the best estimate compression ratio f from statistics in the Appendix.

We can use any defined metric in defining the tree length $L(T_S)$ of a spanning tree T_S. This paper chooses only $d(\cdot)$ because $d(\cdot)$ includes $\delta(\cdot)$ and $\lambda(\cdot)$. $L(T_S)$ in (5) is defined as the total distance from every node q to the

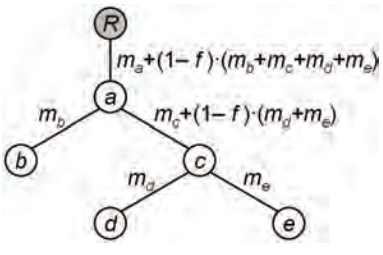


Figure 1. An example of one-time packet compression $d(\cdot)$. The length of transmitted packet is written on each wireless link.

root node R in T_S. The notation $\sum_{\{q|q\in T_S\}}$ in this paper denotes summation is done for all values of the index variable q in T_S.

$$L(T_S) = \sum_{\{q|q\in T_S\}} d(q). \quad (5)$$

Minimizing $L(T_S)$ at no-compression is equivalent to finding SPT, and minimizing $L(T_S)$ at full-compression is equivalent to finding MST.

2.2. Formulation of Our Problem

2.2.1. Assumption of Sensor Network and Convergecast

In a sensor network, all locations of sensor nodes are known. All nodes periodically generate a fixed-length packet which is convergecast to the root node R. During convergecast, packet compression occurs, and we have enough statistics about packet compression.

2.2.2. Definition of Problem

Assuming Subsection 2.2.1, we want to find the spanning tree T_S that minimizes total forwarding energy for a convergecast. To numerically define the problem, our goal is to find the T_S that has minimum tree length using definitions in (4) and (5).

2.3. Analysis of Our Metric $d(\cdot)$

The metric metrics $d(\cdot)$ and $\lambda(\cdot)$ are nonlinear. If we use a nonlinear operator, the link cost $C(i, j)$ contributes differently to the tree length. This influences the preferred type of spanning trees. If we apply $d(\cdot)$, nodes close to R prefer short links, and nodes distant from R choose straight paths to R. If we use $\lambda(\cdot)$, nodes prefer short links and do not mind choosing very long-hop paths.

The nonlinear operator $d(\cdot)$ is very difficult to handle. We found a strange characteristic about $d(\cdot)$. Under $d(\cdot)$, a node can be more distant from R than its descendant. We also found that there is no greedy solution, and are strongly convinced that there is no polynomial optimum solution. Therefore, we decided to rely on heuristics.

Figure 2 can be used to explain why there is no greedy solution. **Figure 2(a)** describes a given four-node network. There are four usable wireless links in **Figure 2(a)**, and their link costs are written by them. The node R is the root node of the spanning tree. **Figures 2(b)** and **2(c)** illustrate two spanning trees. **Figure 2(b)** shows the spanning tree solved by a greedy solution. In **Figures 2(b)** and **2(c)**, node distance is recorded in each node circle.

The greedy solution chooses a link that minimally increases the tree length if the link addition still keeps the sub-tree connected. The sub-tree grows with the addition of selected links. The greedy method firstly chooses $l(R,A)$. The selected links, especially $l(R,A)$, are never removed from the spanning tree. If we add three node distances, the length of the greedy spanning tree T_b becomes 196. In contrast, the length of the minimum tree T_c in **Figure 2(c)** is 189.4. $L(T_c)$ is shorter by 6.6 than $L(T_b)$. **Figure 2** shows the optimum tree does not always include greedy selection.

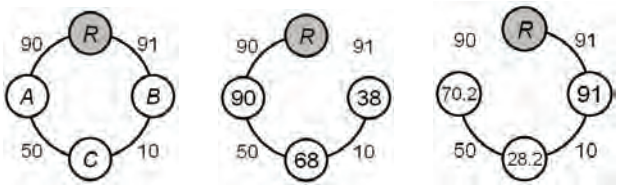
We choose one-time compression $d(\cdot)$ instead of continuous compression $\lambda(\cdot)$ for two reasons. Firstly, $d(\cdot)$ is very simple. Because of simplicity, we can not only calculate the best estimate compression ratio $\hat{f}$ from statistics easily, but also obtain the core operation expressed in (7) with $\theta(1)$ -complexity. Secondly, the $\lambda(\cdot)$ model only covers partial ways of compression. $\lambda(\cdot)$ cannot describe a typical nonzero compression. Compression ratio generally decreases as compression repeats. Nonzero length is common after infinite compression. $\lambda(\cdot)$ cannot cover this type of compression.

3. Packet Compression Ratio Dependent Spanning Tree (CST)

In this section we define CST in two steps and explain its operation. Step A calculates the best estimate $\hat{f}$ of f using statistics. Step B establishes a spanning tree with calculated $\hat{f}$. Step B has two sub-steps. Step B.1 generates an initial spanning tree and Step B.2 describes how to transform to a shorter one.

3.1. Step A: Calculation of the Best Estimate $\hat{f}$ of f

To deduce the best estimate $\hat{f}$, we need statistical data about the average packet length for every active link in a spanning tree. As our major target is to find out the optimal spanning trees with obtained $\hat{f}$, we leave this part to the Appendix.



(a) (b) (c)

Figure 2. Evidence that shows CST solution cannot be greedy. (a) A given network. f=0.8 ; (b) A tree T_b by greedy method, $L(T_b)$=196 ; (c) A tree T_c by optimum method, $L(T_c)$=189.4.

3.2. Step B: Tree Establishment

This step finds the local shortest spanning tree with $\hat{f}$ obtained in Step A. We heuristically generate a CST in two sub-steps. An initial tree is built up in a top-down process (from *R* to leaf nodes) in Step B.1, and is reconfigured in a bottom-up process (from leaf nodes to *R*) to a shorter one in Step B.2.

3.2.1. Step B.1: Initial Tree Setup

Step B.1 defines how to establish an initial tree. An initial tree is built by a combined algorithm of Dijkstra's SPT [10] and Prim's MST [11]. These two famous algorithms are very alike in definition. They are both greedy and satisfy the greedy properties mentioned in Section 2.3. Dijkstra chooses a link that connects a node whose distance to *R* is minimum, and Prim picks up the link whose link cost is minimum. Because both protocols always maintain the working sub-tree as connected, merging two algorithms is straightforward.

Our algorithm uses the Dijkstra's SPT algorithm until it generates an n_S-node sub-tree including *R*. Then we use the Prim's MST algorithm until a complete *n*-node spanning tree is made. n_S is calculated from (6). In (6), $\lceil x \rceil$ represents the integer closest to *x* and not smaller than *x*. We fix *W* in (6) as 8 based on simulation with various *n*-node sensor networks and *f* values.

$$n_S = \lceil (1 - f^W) \cdot n \rceil \quad W = 8, \quad 0 \le f \le 1. \tag{6}$$

We can make two comments on the initial tree. First, *W* has a very large value 8. This makes n_S very close to *n* in (6) for most values of *f*. This means the initial tree is very similar to SPT. Second, we use the SPT algorithm in establishing a spanning tree near *R*. It is because the node, having many descendants, prefers SPT in selecting its parent node.

We explain the reason as follows. Suppose a node *m* is looking for a better parent node. *m*'s current parent node is $p_1(m)$, and there is a candidate m_i in **Figure 3**. *m* decides its parent node as the one that produces less tree length. Assume *m*'s uplink $l(m, p_1(m))$ and $d(m_i)$ are short, and $l(m,m_i)$ and $d(p_1(m))$ are long. For *m*, the link cost of its first hop contributes 100% to the tree length, and the link costs of later hops are added as a ratio of $(1-f)$. However, for *m*'s descendants, the costs of all up-links above *m* are reflected fairly with a ratio of $(1-f)$. If *m* is a leaf node, it prefers $p_1(m)$ because the first hop to *R* is more important than later hops. If *m* has many descendants, *m* has to consider them too. For the sake of its descendants, *m* chooses m_i as a parent node sacrificing itself.

3.2.2. Step B.2: Spanning Tree Establishment

Step B.2 reduces tree length from the initial tree established according to Step B.1. We are going to explain the core operation of the algorithm that decides *m*'s best uplink as shown in **Figure 3**. The node *m* is called a 'designate node.' Throughout the operation, a designate node *m* prepares: 1) 10 closest nodes representing m_i $(1 \le i \le 10)$, 2) $d(m)$, and 3) the number of *m*'s descendent nodes denoted by $h(m)$. A list *Q* is maintained that sorts all nodes in reverse order of $d(\cdot)$.

Figure 3(a) is the current spanning tree T_O, and **Figure 3(b)** is a candidate of a new spanning tree T_{mi} for one of 10 m_i's. $p_1(m)$ and *m*'s descendants are excluded from m_i. For T_O and all candidate spanning trees T_{mi}, we calculate $L(T_O)$ and $L(T_{mi})$ and choose a tree with the least tree length. If the chosen parent node is different from $p_1(m)$, we reconfigure the spanning tree according to **Figure 3(b)** and update influenced variables.

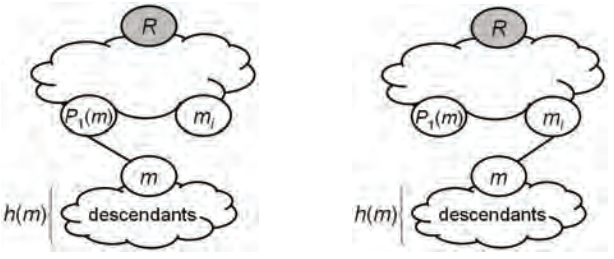
The Tree improvement L_{mi} defined by $(L(T_{mi}) - L(T_O))$ indicates relatively improved tree length with a new parent node m_i. Positive L_{mi} means the new tree is better. If we apply our metric $d(\cdot)$, L_{mi} becomes (7). If we replace $d(\cdot)$ with $\delta(\cdot)$, (7) changes to (12). Note (12) has $\theta(1)$ -complexity because it only includes known terms including $\delta(\cdot)|_{T_O}$.

So far we have explained how to improve the tree for a designate node. We will now give comments on how to assign designate nodes. A designate node *m* is chosen from *Q* by the reverse order of $d(m)$. After performing all jobs mentioned above, the algorithm updated the queue *Q* to choose the next designate node m_n as the one whose $d(m_n)$ is the smallest but larger than *dm* which is $d(m)$ before reconfiguration. We call a cycle every related operation during which *Q* is totally scanned once for choosing designate nodes. Cycles repeat until no more improvement is possible. We observed through simulation that calculation usually finishes at the second or the third cycle. In this way, we achieve the locally best CST.

The CST algorithm for a cycle is summarized below.

1) Prepare m_i, $d(m)$, $h(m)$ for every *m* and establish a queue *Q* in reverse order of $d(\cdot)$. Let $dm = \infty$.

2) Check whether *m*' exists whose $d(m')$ is the largest but less than *dm*.



(a) (b)

Figure 3. Spanning tree before and after reconfiguration. (a) Tree T_O before reconfiguration; (b) Tree T_{mi} after reconfiguration.

(If *m*' does not exist,) the current cycle is completed.

(If *m*' exists,) assign *m*' as a new designate node *m* and update *dm* to the value of *dm*'.

3) For every allowed m_i, calculate L_{mi} using (12). Check whether a positive L_{mi} exists.

(If a positive L_{mi} exists,)

a) *m*'s new parent node is assigned by m_j which generates the largest positive L_{mj} among m_i.

b) Reconfigure the tree as drawn in **Figure 3(b)**.

c) Update $d(\cdot)$ and $h(\cdot)$ at the nodes $p_1(m)$, m_j , *m* and all of *m*'s descendants.

d) Re-sort the queue *Q* in reverse order of $d(\cdot)$.

4) Go back to 2.

Now we define L_{mi} as a $\theta(1)$ -complexity expression. Reconfiguration in **Figure 3** changes node distance for node *m* and *m*'s descendants only. To visualize the dependency between a spanning tree and distance, the used tree is specified after $d(\cdot)$ or $\delta(\cdot)$. *m*'s descendant nodes vary in distance after reconfiguration by $((1-f)\cdot\delta(m)|_{T_{mi}} - (1-f)\cdot\delta(m)|_{T_O})$. If we denote *m*'s descendent node set by *p_*(*m*), the above discussion is mathematically expressed as (7).

$$\begin{aligned} L_{mi} &= L(T_{mi}) - L(T_O) \\ &= \sum_{\{q|q\in\{m\cup p\_(m)\}\}} d(q)|_{T_{mi}} - \sum_{\{q|q\in\{m\cup p\_(m)\}\}} d(q)|_{T_O} \\ &= d(m)|_{T_{mi}} + (1-f)\cdot h(m)\cdot\delta(m)|_{T_{mi}} \\ &\quad - (d(m)|_{T_O} + (1-f)\cdot h(m)\cdot\delta(m)|_{T_O}). \end{aligned} \tag{7}$$

The complex nonlinear operator $d(\cdot)$ in (7) can be replaced by $\delta(m)$ using (8) and (9).

$$d(m)|_{T_{mi}} = (1-f)\cdot\delta(m)|_{T_{mi}} + f\cdot C(m, m_i), \tag{8}$$

$$d(m)|_{T_O} = (1-f)\cdot\delta(m)|_{T_O} + f\cdot C(m, p_1(m)). \tag{9}$$

Because *Q* maintains $d(m)$ in the tree T_O, we remove $\delta(m)|_{T_{mi}}$ using **Figure 3(b)**.

$$\delta(m)|_{T_{mi}} = \delta(m_i)|_{T_O} + C(m, m_i). \tag{10}$$

Replacing $d(m)$ in (7) with $\delta(m)$ using (8) and (9), and removing $\delta(m)|_{T_{mi}}$ using (10), L_{mi} is expressed in terms of $\delta(\cdot)$ in (11). This is finally transformed to (12) to denote L_{mi} with $d(\cdot)$ using (9).

$$\begin{aligned} L_{mi} &= (1-f)\cdot(1+h(m))\cdot((\delta(m_i)-\delta(m))|_{T_O} \\ &\quad + C(m, m_i)) + f\cdot(C(m, m_i) - C(m, p_1(m))) \end{aligned} \tag{11}$$

$$\begin{aligned} &= (1+h(m))\cdot\{(d(m_i)-d(m))|_{T_O} \\ &\quad - f\cdot C(m_i, p_1(m_i)) + f\cdot C(m, p_1(m)) \\ &\quad + (1-f)\cdot C(m, m_i)\} + f\cdot(C(m, m_i) \\ &\quad - C(m, p_1(m))). \end{aligned} \tag{12}$$

We hope CST be better than or equal to MST and SPT for any f ($0 \le f \le 1$). To do so, CST at least has to satisfy the next two properties. As two proofs are very similar, we only prove Property 1.

***Property* 1** If *f* is 1, CST is equivalent to MST.

If *f* is 1, the CST algorithm assigns MST as the initial tree at Step A. MST is optimum at f =1. As the initial tree is already optimum, finding a positive L_{mi} for any designate node *m* is impossible. This leads to no reconfiguration at Step B.2.

***Property* 2** If *f* is 0, CST is equivalent to SPT.

4. Simulation and Analysis

We investigated the tree lengths of CST, SPT and MST for various *f*s according to (4) and (5). We have 25-node, 100-node and 400-node sensor networks with nodes located randomly in a unit-length square sensor field. We also select the root node randomly. We prepare 100 kinds of sensor networks for each condition and deduce the average tree length to draw curves. We basically use the link cost $C(i, j)$ as a square of distance from *i* to *j*, and we consider the case of the fourth power of distance additionally because the transmission loss model is proportional to l^2~l^4 with the wireless transmission distance *l*.

Figure 4 shows five CSTs with varying *f* from 0 to 1 in a 35-node sensor network with fixed root node *R* at the center top position denoted by big dots in the sensor field. CSTs in **Figures 4(a)** and **4(e)** are equivalent to SPT and MST respectively. We can observe that paths along SPT have a tendency to be straight toward *R* in **Figure 4(a)**, and SPT links are longer than MST links in **Figure 4(e)**. When *f* increases from 0 to 0.75, we notice only a few changes. Most changes occur in the range of $0.75 \le f \le 1$.

Figures 5 to **8** draw the tree lengths using (5) for the entire range of f ($0 \le f \le 1$). **Figures 5** to **7** indicate the tree lengths of SPT, MST and CST when the numbers of nodes are 25, 100 and 400 respectively. All curves in these figures decrease monotonously with increasing *f* because the tree length is short with a high *f* value due to (4). CST is always the shortest for all *f*s, and SPT and MST curves cross each other at *f* value of a little more than 0.9. The difference in tree lengths between SPT and MST cases are large at a large sensor network and a sparse sensor network.

We additionally examined the case of the fourth power of distance instead of square distance assigned to the link cost $C(i, j)$ in **Figure 8**. As tree length in **Figure 8** drops sharply as *f* increases, we use the log scale in the *y*-axis. Except when using log or nonlog scales, **Figures 5** to **8** are similar in curve pattern.

Table 1 shows tree lengths and their averages for SPT and MST relative to CST for various *f*s at 20% intervals. 'Relative' means the results are obtained after being divided by CST length for the same condition. A special *f*

(a) (b) (c) (d) (e)

Figure 4. Changes in CSTs with varying *f*. (a) 0%; (b) 25%; (c) 50%; (d)75%; (e)100%.

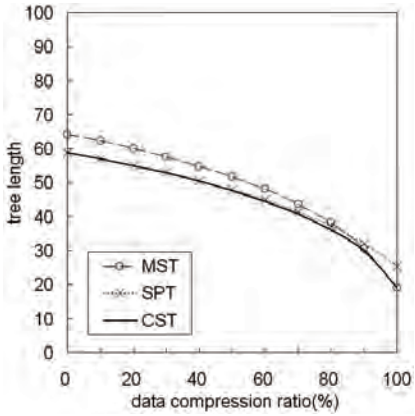


Figure 5. Tree lengths of MST, SPT and CST with varying *f* in 25-node sensor networks.

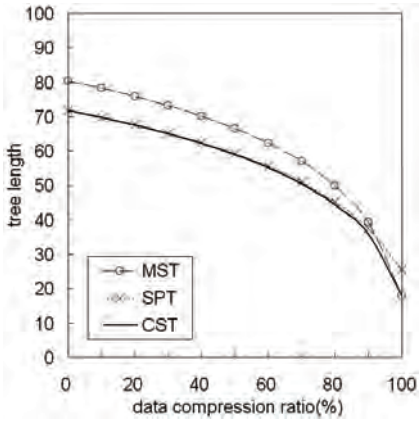


Figure 6. Tree lengths of MST, SPT and CST with varying *f* in 100-node sensor networks.

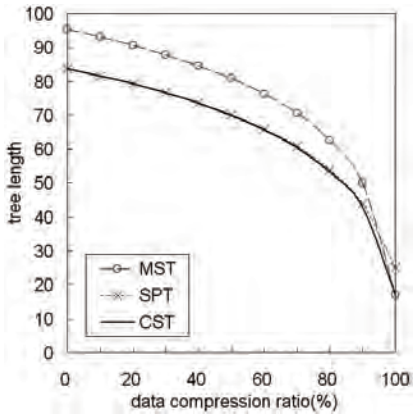


Figure 7. Tree lengths of MST, SPT and CST with varying *f* in 400-node sensor networks.

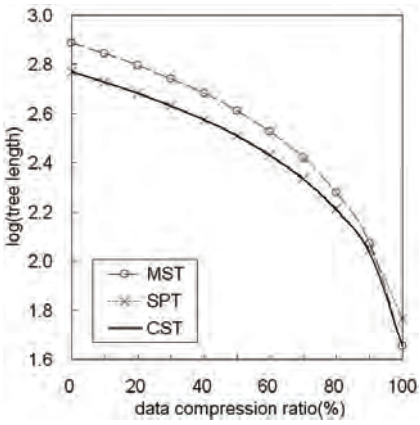


Figure 8. Tree lengths of MST, SPT and CST with varying *f* in 100-node sensor networks. (*C*(*i*, *j*) is re-defined as fourth power of normal transmission distance.)

Table 1. Relative tree length of CST, SPT and MST in 100-node sensor networks.

Routing tree \ *f* (%)	0	20	40	60	80	93	100	Average
CST	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
SPT	100.0	100.1	100.5	101.5	104.2	110.3	140.7	107.8
MST	148.0	146.0	143.9	138.4	128.5	111.4	100.0	134.1
MSPT*	100.0	100.1	100.5	101.5	104.2	110.3	100.0	102.4

* MSPT is the tree from SPT and MST which produces the shorter tree length.

value of 93% is added at which the SPT curve and MST curve coincide. **Table 1** introduces MSPT in the last row. MSTP is defined as the shorter tree from MST and SPT for each given condition.

The far right column in **Table 1** shows averages of the relative tree lengths on a line except for the value at f = 93%. The average MSPTs include this value specially because MSPT has a peak value at f = 93%. According to **Table 1**, CST is 34.1% and 7.8% shorter than MST and SPT respectively in terms of tree length. CST still outperforms MSPT by 2.4%.

5. Conclusions

In convergecast sensor networks, many routing schemes have proposed packet compression to save energy. The Shortest Path spanning Tree (SPT) and the Minimum Spanning Tree (MST) are only popularly used because they are optimal at no-compression and full-compression respectively. We proposed two important concepts. One is a simple one-time compression model. The other is a new routing method called packet Compression ratio dependent Spanning Tree (CST).

CST is superior to SPT and MST for three reasons. Firstly, CST provides us the best estimate $\hat{f}$ of compression ratio f using collected statistics. Secondly, we can calculate CST easily using a one-time compression model. Thirdly, CST outperforms CST, MST and SPT at all considered sensor network environments and is optimum at no-compression and full-compression. Simulation shows CST outperforms MST and SPT by 34.1% and 7.8% respectively in averaging over many compression ratio fs. These observations confirm that CST is very useful in convergecast sensor networks.

6. References

[1] J. N. Al-Karaki and A. E. Kamal, "Routing Techniques in Wireless Sensor Networks: A Survey," *IEEE Wireless Communications*, Vol. 11, No. 6, December 2004, pp. 6-28.

[2] J. N. Al-Karaki, R. Ul-Mustafa and A. E. Kamal, "Data Aggregation in Wireless Sensor Networks—Exact and Approximate Algorithms," *Proceedings of IEEE Workshop on High Performance Switching and Routing* (*HPSR*), Phoenix, April 2004, pp. 18-21.

[3] I. Demirkol, C. Ersoy and F. Alagoz, "MAC Protocols for Wireless Sensor Networks: A Survey," *IEEE Communication Magazine*, Vol. 44, No. 4, April 2006, pp. 115-121.

[4] R. Vidhyapriya and P. T. Vanathi, "Conserving Energy in Wireless Sensor Networks," *IEEE Potentials*, Vol. 26, No. 5, 2007, pp. 37-42.

[5] C.-K. Liang, Y.-J. Huang and J.-D. Lin, "An Energy Efficient Routing Scheme in Wireless Sensor Networks," 22*nd International Conference on Advanced Information Networking and Applications* (*AINAW*), Okinawa, March 2008, pp. 916-921.

[6] L. K. Lawrence, "Sensor and Data Fusion Concept and Applications," 2nd Edition, SPIE Optical Engineering Press, Bellingham, 1999.

[7] S. Upadhyayula and S. K. S. Gupta, "Spanning Tree based Algorithms for Low Latency and Energy Efficient Data Aggregation Enhanced Convergecast (DAC) in Wireless Sensor Networks," *Ad Hoc Networks*, Vol. 5, No. 5, July 2007, pp. 626-648.

[8] H. Luo, J. Luo, Y. Liu and S. K. Das, "Adaptive Data Fusion for Energy Efficient Routing in Wireless Sensor Networks," *IEEE Transactions on Mobile Computing*, Vol. 55, No. 10, October 2006, pp. 1286-1299.

[9] H. Luo, J. Luo, Y. Liu and S. K. Das, "Routing Correlated Data with Fusion Cost in Wireless Sensor Networks," *IEEE Transactions on Mobile Computing*, Vol. 5, No. 11, November 2006, pp. 1620-1632.

[10] H. C. Thomas, E. L. Charles, L. R. Ronald and S. Clifford, "Introduction to Algorithms," 2nd Edition, MIT press, Cambridge, 2001.

[11] R. C. Prim, "Shortest Connection Networks and some Generalizations," *Bell System Technical Journal*, Vol. 36, 1957, pp. 1389-1401.

Appendix: Step A—Calculation of the Best Estimate $\hat{f}$ of Compression Ratio *f*

This step demonstrates how to calculate the best estimate $\hat{f}$ of compression ratio *f* with a given spanning tree T_S from collected statistics. We call the quadratic modeling error M_E by adding the square of link estimate error $L_E(l(i, j))$ of our one-time compression model throughout all links in T_S shown in (13). We can find the only $\hat{f}$ that minimizes $M_E(f)$ in (14).

$$M_E(f) = \sum_{\{l(i,j)|l(i,j)\in T_S\}} L_E(l(i, j))^2. \quad (13)$$

$$M_E(\hat{f}) = \min\{M_E(f)\} \le M_E(f). \quad (14)$$

The link estimate error $L_E(l(i, j))$ is defined in (15). To perform convergecast, every node forwards packets along T_S, compressing the received packets with the compression ratio *f*. To calculate the best estimate $\hat{f}$, we use statistics to provide the average message length E_{ij} on a link $l(i, j)$ in T_S. If we apply our one-time compression model, we can build up the one-time compression model length W_{ij} for all active links $l(i, j)$ according to the procedure mentioned in Subsection 2.1 and **Figure 1**. Utilizing the property that W_{ij} is a linear equation of *f*, we can obtain every a_{ij} and b_{ij} for every active links $l(i, j)$ in (16).

$$L_E(l(i, j)) = | E_{ij} - W_{ij} |, \quad (15)$$

$$W_{ij} = a_{ij} \cdot f + b_{ij}, \qquad a_{ij}, b_{ij} \text{ are constants.} \quad (16)$$

Combining the four definitions (13) to (16), we get (17). Equation (17) turns to a simple quadratic equation of *f* in (18) for the constants α, β, and γ.

$$\begin{aligned} M_E(f) &= \sum_{\{(i,j)|l(i,j)\in T_S\}} \{a_{ij} \cdot f + (b_{ij} - E_{ij})\}^2 \\ &= \alpha \cdot f^2 - \beta \cdot f + \gamma, \end{aligned} \quad (17)$$

$$\alpha(\neq 0), \beta, \gamma \text{ are constants.} \quad (18)$$

The quadratic equation in (18) has the minimum value at the origin. Because a parabolic and its axis cross at the origin, the axis of the parabolic guarantees the minimum modeling error $M_E(f)$. The axis corresponds to

$$\hat{f} = \frac{\beta}{2\alpha} = \frac{\sum_{\{(i,j)|l(i,j)\in T_S\}} (E_{ij} - b_{ij})}{\sum_{\{(i,j)|l(i,j)\in T_S\}} a_{ij}}. \quad (19)$$

All E_{ij}, a_{ij}, and b_{ij} are available, thus we can get $\hat{f}$.

Wireless Sensor Network, 2010, 2, 512-519
doi:10.4236/wsn.2010.27063 Published Online July 2010 (http://www.SciRP.org/journal/wsn)

Passive Loss Inference in Wireless Sensor Networks Using EM Algorithm*

Yu Yang[1,2], Zhulin An[1,2], Yongjun Xu[1], Xiaowei Li[1], Canfeng Chen[3]
[1]*Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China*
[2]*Graduate University of Chinese Academy of Sciences, Beijing, China*
[3]*Nokia Research Center, Beijing, China*
E-mail: {*yuyang, anzhulin, xyj, lxw*}*@ict.ac.cn, canfeng-david.chen@nokia.com*
Received April 20, 2010*; revised May* 4, 2010*; accepted May* 20, 2010

Abstract

Wireless Sensor Networks (WSNs) are mainly deployed for data acquisition, thus, the network performance can be passively measured by exploiting whether application data from various sensor nodes reach the sink. In this paper, therefore, we take into account the unique data aggregation communication paradigm of WSNs and model the problem of link loss rates inference as a Maximum-Likelihood Estimation problem. And we propose an inference algorithm based on the standard Expectation-Maximization (EM) techniques. Our algorithm is applicable not only to periodic data collection scenarios but to event detection scenarios. Finally, we validate the algorithm through simulations and it exhibits good performance and scalability.

Keywords: Wireless Sensor Networks, Passive Measurement, Network Tomography, Data Aggregation, EM Algorithm

1. Introduction

Recent deployments of Wireless Sensor Networks (WSNs) indicate that energy-efficient mechanisms of performance measurement are needed. The unattended nature and complexity of WSNs require that the network managers are given updated indications on the network health, for instance, the state of network links and nodes, after deployment [1]. Such information can provide early warnings of system failures, help in incremental deployment of nodes, or tuning network algorithms [1-3]. However, WSNs measurement is not a trivial task because sensor nodes have limited resources (bandwidth and power). Considering further the distributed nature of the algorithm, the large number of the nodes, and the interaction of the nodes with the environment, leads to the fact that WSNs are very hard to measure. The most commonly used approach for evaluating network performance is active measurement, which collects statistics from internal nodes directly or even injects probe messages into the network to aid in performance evaluation. However, it is usually impractical to rely on the use of active measurement in WSNs, which is not scalable or bandwidth-efficient. On the other hand, WSNs are mainly deployed for data acquisition, thus the network performance can be passively measured by exploiting whether application data from various sensor nodes reach the sink.

The typical mode of communication in WSNs is from multiple data sources to one sink, rather than communication between any pair of nodes. Thus it usually involves tree-based topology rooted at the sink. Furthermore, since the data being collected by multiple nodes are based on common phenomena, there is likely to be some redundancy in the data being communicated. Moreover, for many node designs, the wireless transceiver requires the largest share of the overall power budget. Thus data aggregation has been put forward as a particularly useful communication paradigm for WSNs [4]. The idea of data aggregation is to combine the data coming from different sources, to eliminate redundancy, to minimize the number of transmissions and thus to save energy. In the process of data aggregation, sensor nodes in the network attempts to forward the sensing data they have collected back to the sink via a data aggregation tree. When an intermediate node in the aggregation tree receives data from multiple source nodes, it checks the contents of incoming data, combines them by eliminating redundant information and then forwards the aggregated packet to

*Supported by the National High Technology Research and Development Program of China (No. 2007AA12Z321) and the National Natural Science Foundation of China (No. 60772070; 60873244).

its parent [1,4].

Specifically, WSNs can be used advantageously for periodic data collection or events detection scenarios [5]. Periodic data collection is required for operations such as tracking of the material flows, health monitoring of equipment/process. In these scenarios, all the nodes in the network are typically organized into one aggregation tree, through which nodes periodically send application data to the sink. On the other hand, WSNs are more often used in events detection scenarios, where sensor nodes are deployed to detect and classify rare and random events, such as alarm and fault detection notifications. **Figure 1** depicts an example of data aggregation communication paradigm in events detection scenarios. In this example, two interested events happened in the deployment area. Then sensor nodes within a distance S (called the event range) and other related nodes construct two aggregation trees, through which the information of the two events collected by the sensor nodes is aggregated and sent to the sink.

In view of the unique data aggregation communication paradigm of WSNs, this study is the first to develop the network model to represent the network topology composed of multiple aggregation trees. Based on the network model, we further model the problem of link loss rates inference as a Maximum-Likelihood Estimation (MLE) problem and an inference algorithm using the standard Expectation-Maximization (EM) techniques [6] is proposed. Compared with those in wired communication networks, links in WSNs are prone to suffer high packet loss rates, which will result in unreliable and incomplete data [1]. It would thus be useful to identify links with high loss rates (lossy links) at run-time. The presented algorithm passively monitors the application traffic between sensor nodes and the sink, and then uses network tomography technology [7,8] to locate lossy links. Our algorithm handles well topology changes and hence is applicable not only to periodic data collection scenarios but to events detection scenarios.

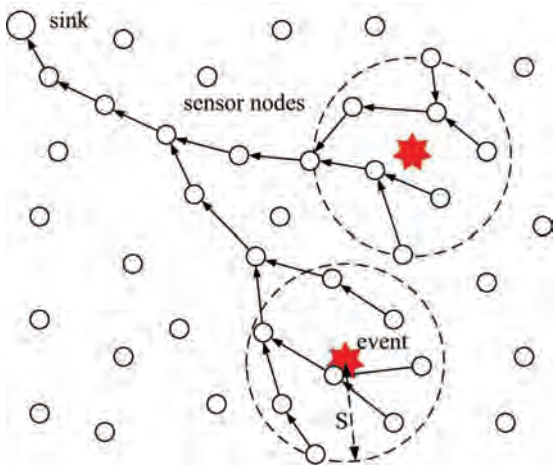


Figure 1. Data aggregation communication paradigm in events detection scenarios: An example.

The rest of this paper is organized as follows. Section 2 focuses on the related work. The inference models are presented in Section 3 and the inference algorithm is proposed in Section 4 based on the models. Then the simulation is shown in Section 5. Finally, Section 6 concludes the paper.

2. Related Work

WSNs measurement recently has received considerable attention from the research community. Existing work can be classified into two categories: active and passive measurement. Active measurement researches mainly rely on proactive approaches [9-13]. The basic idea underlying these approaches is that sensor nodes periodically report the internal status information of themselves to the sink, such as residual energy, node failures, link status, neighbor list, and the like. However, regularly sending status information from sensor nodes to the sink requires significant communication overheads, which would inevitably speed up the depletion of energy. Furthermore, contrary to wired networks where network measurement information can be delivered reliably, reports sent from sensor nodes also suffer losses. The sink has therefore no guarantee that it will receive up-to-date information.

Recently, some researches have shown that application data that sensor nodes send to the sink can be used to measure the network itself. In 2004, G. Hartl *et al*. considered firstly applying network tomography in WSNs based on data aggregation paradigm [14]. They formulated loss inference as an MLE problem and presented an inference algorithm based on the standard EM algorithm. However, their approach requires two restrictive assumptions: first, there is only one aggregation tree in the network and second, the aggregation tree remains static for the entire loss rate inference process. Obviously, these assumptions limit the potential applications of their algorithm in practice.

In 2005, Mao *et al*. employed the factor graph approach to solve the link loss inference problem [15]. In 2007, Li *et al*. formulated the problem of link loss rate estimation as a Bayesian inference problem and a Gibbs sampling algorithm was proposed [16]. In 2008, E. Shakshuki *et al.* presented an inference mechanism adopting iterative computation under Markov Chain [17]. However, these inference methods also rely on the above two assumptions. On the other hand, there are also some other researches on the loss inference problem in WSNs [18-20]. However, these approaches can only used in the situation where sensor nodes report application data to the sink through multi-hop paths separately. Thus they can not be applied to data aggregation paradigm.

3. Inference Models

In this section, we develop the models which are the basis for the inference. We first model the network and then formulate the loss performance.

3.1. Network Model

Let $N = (V(N), L(N))$ denote a network with sets of nodes $V(N)$ and the set of links $L(N)$. The sink is denoted by s and is assumed to have greater resources available than the other nodes. $(i, j) \in L(N)$ denotes a directed link from node i to node j in the network. Let Ψ denote a set of aggregation trees embedded in N, *i.e.*, $\forall T \in \Psi$, $V(T) \subseteq V(N)$ and $L(T) \subseteq L(N)$. Note that (i, j) can appear in more than one tree. Thus for $\forall (i, j) \in L(N)$, we denote $\Psi_{i,j} \subseteq \Psi$ the set of trees which include link (i, j). Let $(q \to p)_T$ denote the path from node q to node p in T and $L((q \to p)_T)$ denote the set of links on the path. The set of children of node i in T is denoted by $d(i, T) = \{k \in V(T) \mid (k, i) \in L(T)\}$. Define the set of leaf nodes in T by $R(T)$, *i.e.*, $R(T) = \{k \in V(T) \mid d(k, T) = \varnothing\}$. For $\forall k \in V(T) \backslash R(T)$, let $T(k)$ denote the subtree within T rooted at k. Then $R(k, T) = R(T) \cap V(T(k))$ is defined as the set of leaf nodes which are descended from k in T. Also, we denote the set of all of the leaf nodes in the network by $R = \bigcup_{T \in \Psi} R(T)$.

3.2. Loss Model

For each link an independent Bernoulli loss process is assumed, which is a reasonable assumption [21]. Thus $\forall (i, j) \in L(N)$ can be associated with a transmission rate $\alpha_{i,j} \in (0,1)$. Let $\alpha_{i,j}$ be the probability that packets sent from node i to node j are received successfully by j. Accordingly, the loss rate of (i, j) is $\bar{\alpha}_{i,j} = 1 - \alpha_{i,j}$. The flow of the data through an aggregation tree T therefore can be modeled by a stochastic process $X_T = (X_{p,q,T})_{p,q \in V(T)}$, where $X_{p,q,T} \in \{0,1\}$. $X_{p,q,T} = 1$ means that the data sent from node q were successfully received by intermediate node p in the path $(q \to s)_T$. Similarly, $X_{p,q,T} = 0$ means that the data sent from q did not successfully reach p. Let $\alpha = (\alpha_{i,j})_{(i,j) \in L(N)}$ denote the set of transmission rates for all links.

We assume that information about which leaf nodes' data is present in the aggregated data must also be bundled into the aggregated packet. Thus for $\forall T \in \Psi$, consider the collection of data by the sink to be an experiment. Each round of data collection will be considered a trial within this experiment. Suppose each leaf node tries to send data in each round. The outcome of each trial will be a record of which leaf nodes the sink received data from in that round. It is worth noting that the inference method in [14] needs the record about all nodes in the network, including leaf nodes and intermediate nodes. Therefore our method is more practical than that in [14].

In terms of the stochastic process X_T defined above, each trial outcome is a random value $X_{R(T)} = (X_{s,k,T})_{k \in R(T)} \in \Omega_{R(T)} = \{0,1\}^{|R(T)|}$. Thus the overall outcome of Ψ can be denoted by $X_R = \bigcup_{T \in \Psi} X_{R(T)}$. Let $\Omega_{R(T)}(i)$ be the set of outcomes $X_{R(T)}$ in which there is at least one node in $R(i,T)$ whose data were successfully received by the sink, *i.e.*,

$$\Omega_{R(T)}(i) = \left\{ x_{R(T)} \in \Omega_{R(T)} : \vee_{k \in R(i,T)} X_{s,k,T} = 1 \right\}. \quad (1)$$

Let n_T denote the data collection rounds of T. The probability of the n_T independent observations of $X_{R(T)}$ is then

$$p(X_{R(T)}; \alpha) = \prod_{m=1}^{n_T} p(x_{R(T)}^m; \alpha) \quad (2)$$

and the probability of the observations of X_R is

$$p(X_R; \alpha) = \prod_{T \in \Psi} p(X_{R(T)}; \alpha). \quad (3)$$

$p(X_R; \alpha)$ is the likelihood function, that is, the probability that we observe the data X_R given the link transmission rates α. Therefore the problem of inferring link loss rates from measurements at the sink can be formulated as an MLE problem. That is, our goal is to estimate α by $\hat{\alpha}$ such that $\hat{\alpha}$ maximizes the likelihood of observing the outcomes of X_R, *i.e.*,

$$\hat{\alpha} = \arg\max_{\alpha} p(X_R; \alpha). \quad (4)$$

However, X_R is not sufficient to obtain a direct expression for $p(X_R; \alpha)$ and then the above equation can not be solved directly. Fortunately, the standard EM techniques can be applied to the data taken from all of the trees to efficiently solve the problem. The basic idea is that rather than performing a complicated maximization, we "augment" the observable data with unobservable data so that the resulting likelihood has a simpler form.

4. Inference Algorithm

In this section, we further formulate the loss inference problem as an MLE problem with complete observations and then efficiently solve it using the EM algorithm. We begin by presenting some brief background information and then apply the EM algorithm to the loss inference.

4.1. Background

The EM Algorithm is a general method for finding MLE of parameters in statistical models, where the model depends on missing data [6]. Although a problem at first sight may not appear to be an incomplete-data one (as it is in our case), there may be much to be gained computation-wise by artificially formulating it as such to facilitate MLE. For many statistical problems, the complete-

data likelihood has a nice form.

To be more specific, the EM algorithm attempts to find an estimate, $\hat{\alpha}$, of parameter α from the complete data X_c, *i.e.*,

$$\hat{\alpha} = \arg\max_{\alpha} p(X_c;\alpha). \quad (5)$$

Beginning with an arbitrary initial assignment, $\alpha^{(0)}$, the EM algorithm is iterative and alternates until convergence. On each iteration, there are two steps—called the Expectation (E) step and the Maximization (M) step. Hence the name EM algorithm is given to this class of techniques. The E-Step computes the conditional expected value of the complete data likelihood given the observed data, under the probability law induced by the current estimates of α, *i.e.*,

$$Q(\alpha \mid \hat{\alpha}^{(t)}) = \mathrm{E}_{\hat{\alpha}^{(t)}}[p(X_c;\alpha) \mid X_{obs}], \quad (6)$$

where $\hat{\alpha}^{(t)}$ is the current estimates of α and X_{obs} is the observable data. In the M-Step, the expected complete data likelihood function is maximized with respect to α to obtain the new estimates, *i.e.*,

$$\hat{\alpha}^{(t+1)} = \arg\max_{\alpha} Q(\alpha \mid \hat{\alpha}^{(t)}). \quad (7)$$

4.2. Application to Loss Inference

In our loss inference problem, the observable data, X_{obs}, is X_R. Following the method in the above section, we augment the actual observations with the unobservable observations at the interior nodes. To be more specific, for $\forall T \in \Psi$, it is assumed that for $\forall (i, j) \in L(T)$, we can observe at node j whether the packets sent from node i successfully reach j in every round of data collection. In terms of the stochastic process X_T defined in Subsection 3.2, the complete data of T are $X_{c(T)} = (X_{j,i,T})_{(i,j)\in L(T)}$. Base on the independent Bernoulli distribution assumption in Subsection 3.2, therefore, it is easy to realize that the likelihood function of $X_{c(T)}$ can be written as

$$p(X_{c(T)};\alpha) = \prod_{(i,j)\in L(T)} \alpha_{i,j}^{n_{j,i,T}} \cdot \bar{\alpha}_{i,j}^{n_T - n_{j,i,T}}, \quad (8)$$

where $n_{j,i,T}$ is the number of those outcomes of $X_{j,i,T}$ whose value is equal to 1. Similarly to (3), the likelihood function of the complete data of Ψ, $X_c = (X_{c(T)})_{T\in\Psi}$, is

$$p(X_c;\alpha) = \prod_{T\in\Psi} p(X_{c(T)};\alpha). \quad (9)$$

It is convenient to work with the log-likelihood function, $\mathcal{L}(X_c;\alpha) = \log p(X_c;\alpha)$, which is given by

$$\mathcal{L}(X_c;\alpha) = \sum_{(i,j)\in L(N)} \Big(\sum_{T\in\Psi_{i,j}} n_{j,i,T} \log \alpha_{i,j} + \Big(\sum_{T\in\Psi_{i,j}} n_T - \sum_{T\in\Psi_{i,j}} n_{j,i,T} \Big) \log \bar{\alpha}_{i,j} \Big). \quad (10)$$

It can be seen from Subsection 4.1 that the key of the EM algorithm is to compute the conditional expectation of the complete data likelihood. Based on (6) and (10), we can obtain

$$\begin{aligned} Q(\alpha \mid \hat{\alpha}^{(t)}) &= \mathrm{E}_{\hat{\alpha}^{(t)}}[\mathcal{L}(X_c;\alpha) \mid X_R] \\ &= \sum_{(i,j)\in L(N)} \Big(\sum_{T\in\Psi_{i,j}} \hat{n}_{j,i,T} \log \alpha_{i,j} \\ &\quad + \Big(\sum_{T\in\Psi_{i,j}} n_T - \sum_{T\in\Psi_{i,j}} \hat{n}_{j,i,T} \Big) \log \bar{\alpha}_{i,j} \Big), \end{aligned} \quad (11)$$

where $\hat{n}_{j,i,T}$ is the conditional expectation of $n_{j,i,T}$ given the observable data $X_{R(T)}$ under the probability law induced by $\hat{\alpha}^{(t)}$, *i.e.*,

$$\hat{n}_{j,i,T} = \mathrm{E}_{\hat{\alpha}^{(t)}}[n_{j,i,T} \mid X_{R(T)}]. \quad (12)$$

Remember that

$$n_{j,i,T} = \sum_{m=1}^{n_T} x_{j,i,T}^{m}. \quad (13)$$

Thus we have

$$\begin{aligned} \hat{n}_{j,i,T} &= \sum_{m=1}^{n_T} \mathrm{P}_{\hat{\alpha}^{(t)}}[X_{j,i,T} = 1 \mid X_{R(T)} = x_{R(T)}^{m}] \\ &= \sum_{x_{R(T)}\in\Omega_{R(T)}} n_T(x_{R(T)}) \mathrm{P}_{\hat{\alpha}^{(t)}} \\ &\quad [X_{j,i,T} = 1 \mid X_{R(T)} = x_{R(T)}], \end{aligned} \quad (14)$$

where $n_T(x_{R(T)})$ denotes the number of collection rounds for which the outcome $x_{R(T)}$ is obtained. Then the problem is turned into the computation of the conditional probability, $\mathrm{P}_{\hat{\alpha}^{(t)}}[X_{j,i,T} = 1 \mid X_{R(T)} = x_{R(T)}]$. To facilitate the computation, we divide the set of outcomes $x_{R(T)}$ into two groups:

1) $x_{R(T)} \in \Omega_{R(T)}(i)$

This group of outcomes implies that there is at least one node in $R(i,T)$ whose data are aggregated at node i and then successfully received by the sink through path $(i \to s)_T$. This clearly indicates that node j successfully received data from node i, and hence

$$\mathrm{P}_{\hat{\alpha}^{(t)}}[X_{j,i,T} = 1 \mid X_{R(T)} = x_{R(T)}, x_{R(T)} \in \Omega_{R(T)}(i)] = 1. \quad (15)$$

2) $x_{R(T)} \in \Omega_{R(T)} \backslash \Omega_{R(T)}(i)$

This group of outcomes suggests that there is none of nodes in $R(i,T)$ whose data successfully reached the sink. Obviously, this does not indicate whether node j successfully received data from node i. Thus we divide this group of outcomes into two categories and handle them accordingly (for clarity, let $X_{\bar{\Omega}}$ denote the situation $\{X_{j,i,T} = 1 | X_{R(T)} = x_{R(T)}, x_{R(T)} \in \Omega_{R(T)} \backslash \Omega_{R(T)}(i)\}$):

a. There is at least one node in $R(i,T)$ whose data reach node i, *i.e.*, $\vee_{k\in R(i,T)} X_{i,k,T} = 1$. In this case, the data should reach node j and then be lost on path $(j \to s)_T$. Thus,

$$\begin{aligned} &\mathrm{P}_{\hat{\alpha}^{(t)}}[X_{\bar{\Omega}} \mid \vee_{k\in R(i,T)} X_{i,k,T} = 1] \\ &= \hat{\alpha}_{i,j}^{(t)} \cdot (1 - \prod_{(p,q)\in L((j\to s)_T)} \hat{\alpha}_{p,q}^{(t)}) . \end{aligned} \tag{16}$$

b. There is none of nodes in $R(i,T)$ whose data reach node i, *i.e.*, $\vee_{k\in R(i,T)} X_{i,k,T} = 0$. In this case, the value of $X_{j,i,T}$ is independent of $X_{R(T)}$ and depends only on the performance of link (i,j). Thus

$$\mathrm{P}_{\hat{\alpha}^{(t)}}[X_{\bar{\Omega}} \mid \vee_{k\in R(i,T)} X_{i,k,T} = 0] = \hat{\alpha}_{i,j}^{(t)}. \tag{17}$$

For $\forall i \in V(T)\backslash R(T)$, define $\beta(i,T)$ to be the probability that there is at least one node in $R(i,T)$ whose data reach node i, *i.e.*,

$$\beta(i,T) = \mathrm{P}_{\hat{\alpha}^{(t)}}[\vee_{k\in R(i,T)} X_{i,k,T} = 1]. \tag{18}$$

Based on the formula of total probability, therefore, for $x_{R(T)} \in \Omega_{R(T)}\backslash\Omega_{R(T)}(i)$, we obtain

$$\begin{aligned} \mathrm{P}_{\hat{\alpha}^{(t)}}[X_{\bar{\Omega}}] &= \mathrm{P}_{\hat{\alpha}^{(t)}}[X_{\bar{\Omega}} \mid \bigvee_{k\in R(i,T)} X_{i,k,T} = 1] \cdot \beta(i,T) \\ &+ \mathrm{P}_{\hat{\alpha}^{(t)}}[X_{\bar{\Omega}} \mid \bigvee_{k\in R(i,T)} X_{i,k,T} = 0] \cdot \bar{\beta}(i,T). \end{aligned} \tag{19}$$

$\bar{\beta}(i,T) = 1 - \beta(i,T)$ is the probability that i does not receive data from any node in $R(i,T)$. This means that i does not receive data from any leaf node descended from all of i's children, namely from $R(q,T)$, $q \in d(i,T)$. For any $R(q)$, there are two reasons that i does not receive data from it, one is that q does not receive data from any node in $R(q)$, the other is that q receives data from $R(q)$, but the data are lost on link (q,i). Thus for $\forall i \in V(T)\backslash R(T)$, the following recursive equation can be obtained,

$$\bar{\beta}(i,T) = \prod_{q\in d(i,T)} (\bar{\beta}(q,T) + \beta(q,T) \cdot (1 - \hat{\alpha}_{q,i}^{(t)})). \tag{20}$$

Using (20) we can recursively compute $\beta(i,T)$ for $\forall i \in V(T)$ based on $\hat{\alpha}^{(t)}$ (for $\forall i \in R(T)$, $\beta(i,T) = 1$).

While combining(14), (15), and (19), we can work out $\hat{n}_{j,i,T}$ and then use (11) to compute the conditional expectation of the complete data likelihood, $Q(\alpha \mid \hat{\alpha}^{(t)})$.

For the M-Step of the EM algorithm, maximization of $Q(\alpha \mid \hat{\alpha}^{(t)})$ is trivial, as the stationary point conditions

$$\frac{\partial Q(\alpha \mid \hat{\alpha}^{(t)})}{\partial \alpha_{i,j}} = 0 , \qquad (i,j) \in L(N). \tag{21}$$

immediately yield

$$\hat{\alpha}_{i,j}^{(t+1)} = \frac{\sum_{T\in\Psi_{i,j}} \hat{n}_{j,i,T}}{\sum_{T\in\Psi_{i,j}} n_T} , \qquad (i,j) \in L(N). \tag{22}$$

4.3. Algorithm Description

The algorithm starts with an arbitrary initial assignment of link transmission rates, $\hat{\alpha}^{(0)}$. Simulations suggest that the values that the algorithm converges to are independent of initial values, which is a property of the EM algorithm. Then the E-step and M-step are iterated until the inferred transmission rate of each link changes by less than a termination criterion between consecutive iterations. Finally, the algorithm use a threshold t_l to decide whether a link is lossy. That is, for $\forall(i,j) \in L(N)$, if $\hat{\alpha}_{i,j}$ lies below t_l, (i,j) is added to the solution set of lossy links, $\mathcal{X}$, which originally is empty. The algorithm is presented in **Table 1**.

5. Simulation

5.1. Simulation Setup

The algorithm is simulated over OMNeT++ network simulator. Random networks consisting of 500, 1000 and 2000 nodes (N500, N1000 and N2000) are generated. Furthermore, two different distributions are used for assigning loss rates to links. The first loss distribution (LD1) is a random selection model, where the intended link loss rate is selected randomly in the interval [0.01, 0.4]. In the second loss distribution (LD2), link transmission rates are drawn from a distribution with probability density function $f(\alpha) = \lambda\alpha^{\lambda-1}$, for $0 < \alpha < 1$ parameterized by $\lambda > 1$. The expected value of this random variable is $\lambda/(1+\lambda)$. In our simulations, λ is chosen as 4 so that the expected link loss rate is 0.2.

Table 1. The description of the EM algorithm.

//Input:
// Network model $N = (V, L)$
// Observations at the sink, X_R
// Data collection round of each tree, *i.e.*, $(n_T)_{T\in\Psi}$
//Output:
// The set of lossy links $\mathcal{X}$

1. ***Initialization***. Select the initial link transmission rates $\hat{\alpha}^{(0)}$;
2. ***Expectation***. Given the current estimate $\hat{\alpha}^{(t)}$, compute the conditional expectation of the log-likelihood given the observable data X_R under the probability law induced by $\hat{\alpha}^{(t)}$ using (11) ~ (20);
3. ***Maximization***. The conditional expectation is maximized to obtain the new estimates $\hat{\alpha}^{(t+1)}$, which is given by (22);
4. ***Iteration***. Iterate steps 2 and 3 until a termination criterion is satisfied. Set $\hat{\alpha} = \hat{\alpha}^{(l)}$, where l is the terminal number of iterations;
5. ***Decision.*** for $\forall(i,j) \in L(N)$
   if $\hat{\alpha}_{i,j} < t_l$ then add (i,j) to $\mathcal{X}$

We say a link is lossy if its transmission rate lies below 0.8 (*i.e.*, its loss rate is above 0.2); otherwise, it is normal. The termination criterion of the algorithm is chosen as 0.0001. Moreover, the performance of the algorithm is evaluated using two metrics: the detection rate (DR), which is the percentage of links that are correctly inferred as lossy, and the false positive rate (FPR), which is the percentage of links that are normal but are inferred as lossy. With $\mathcal{F}$ denoting the set of the actual lossy links and $\mathcal{X}$ the set of links identified as lossy by the algorithm, these two rates are given by:

$$\text{DR} = |\mathcal{F} \bigcap \mathcal{X}| / \mathcal{F}; \text{FPR} = |\mathcal{X} \setminus \mathcal{F}| / \mathcal{X}.$$

5.2. Simulations in Periodic Data Collection Scenarios

The algorithm is first evaluated in periodic data collection scenarios. In this group of simulations, one random tree topology is generated in every network. The branching ratio at each non-leaf node is randomly chosen between 1 and an upper bound of 10. The data collection rounds in each simulation are chosen as 200 and the algorithm is performed every 20 rounds to observe its convergent tendency.

Figure 2 and **Figure 3** plot DR and FPR of the algorithm with LD1 and LD2 respectively. We observe that the algorithm is quite insensitive to different loss distributions. We also note that the algorithm shows good scalability. As the network size increases, the algorithm has high DR and low FPR and the accuracy slightly decreases. Furthermore, we observe the accuracy increases as the data collection rounds increases and the algorithm converges fast: when the data collection rounds reach 100, DR is about 0.9 and FPR is about 0.1.

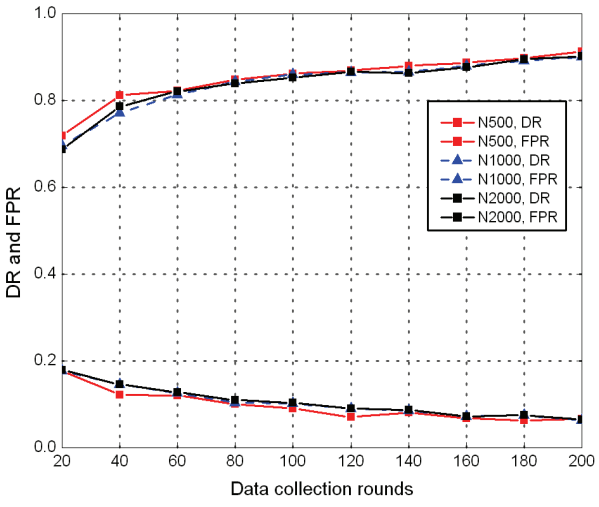


Figure 2. DR and FPR with LD1.

Figure 4 shows how accurate the inference is when the links are rank ordered based on our "confidence" in the inference. We quantify the confidence as the inferred loss rates of the links. The links in N1000 with LD2 are considered in the order of decreasing inferred values (the data collection rounds are 100). We plot 3 curves: the true number of lossy links in the set of links considered up to that point, the number of correct inferences, and the number of false positives. We note that the confidence rating assigned by the algorithm works very well. There are zero false positives for the top 112 rank ordered links. Moreover, each of the first 346 true lossy links in the rank ordered list is correctly identified as lossy (*i.e.*, none of these true lossy links is "missed"). These results suggest that the confidence estimate of the algorithm can be used to rank the order of the inferred lossy links so that the top few inferences are (almost) perfectly accurate. This is likely to be useful in a practical setting where we may want to identify at least a small number of lossy links with certainty so that corrective action can be taken.

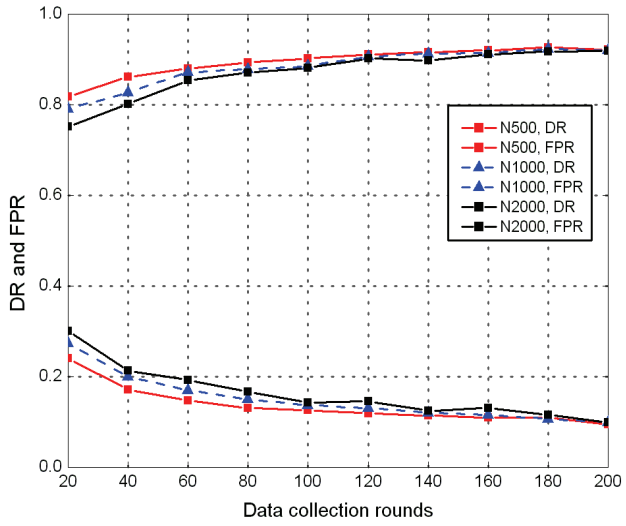


Figure 3. DR and FPR with LD2.

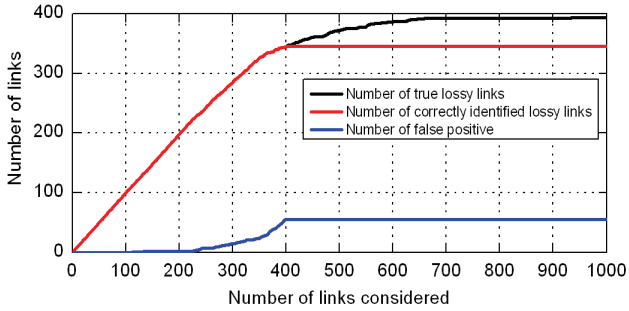


Figure 4. The links are rank ordered based on inference confidence (N1000, LD2, 100 collection rounds).

5.3. Simulations in Events Detection Scenarios

In this section, the algorithm is evaluated in events detection scenarios. The experimental setup of this group of simulations is as follows. The networks are randomly placed in a square area of D size and the sink is located at the upper left corner of the area. The communication radius of sensor nodes is set as 0.05D. In each experiment, 100 events randomly happen one after another in the area with the event range S. For each event, a data aggregation tree is constructed using the greedy incremental tree algorithm proposed in [22] to transmit the event information to the sink. See **Figure 1**. After data collection, the algorithm is performed on the observations of all trees to obtain the combined results.

The simulations are divided into two groups. The first group of simulations is used to evaluate the convergence of the algorithm in events detection scenarios. In this group the data collection rounds of each aggregation tree are selected randomly in [0, 50], [50, 100] or [100, 150] and the event ranges are set as 0.1D. The second group is used to investigate the performance of our algorithm with different scales of aggregation trees in events detection scenarios. In this group the event ranges vary between 0.05D, 0.1D and 0.15D while the data collection rounds of each aggregation tree are uniformly distributed in [50, 100].

Figure 5 and **Figure 6** depict the simulation results of the first group of experiments. The figures suggest that the algorithm is also robust against different loss distributions and scales well. Furthermore, the algorithm shows good convergence: even through the collection rounds of each tree are less than 50, the algorithm still has high DR and low FPR. The underlying reason for this success can be attributed to the fact that for the links that are contained by more than one trees, the algorithm can combine more observations and therefore the results are more accurate.

The simulation results of the second group of experiments are illustrated in **Figure 7** and **Figure 8**. We also note that the algorithm performs similarly under different

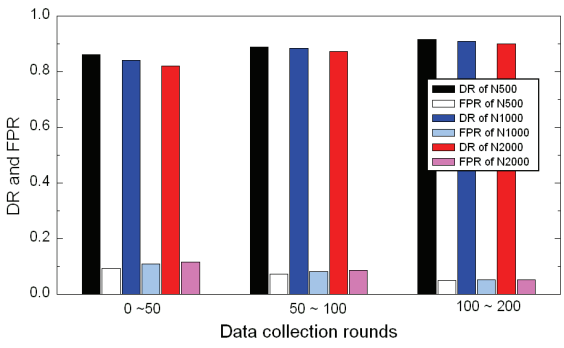


Figure 5. DR and FPR with different data collection rounds of each aggregation tree (LD1).

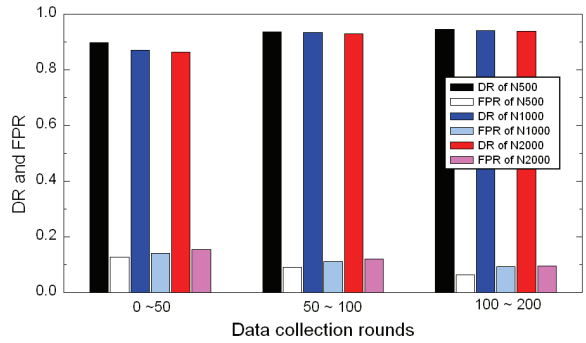


Figure 6. DR and FPR with different data collection rounds of each aggregation tree (LD2).

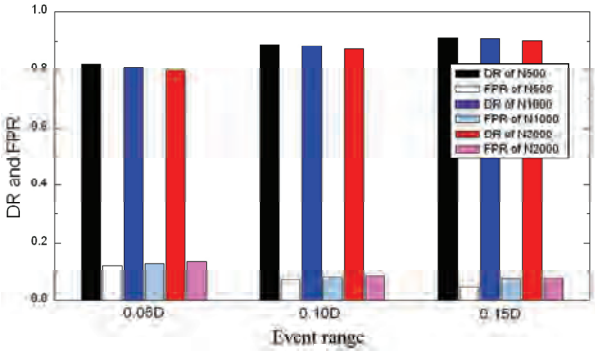


Figure 7. DR and FPR with different event ranges (LD1).

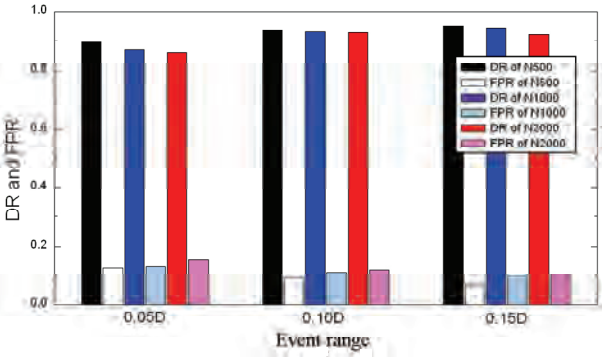


Figure 8. DR and FPR with different event ranges (LD2).

network scales and loss distributions. Moreover, we observe that its accuracy increases as evnt range, S, increases. The number of the links that shared by multiple trees will increase when S increase. Thus the algorithm can perform more accurately by combining more observations.

6. Conclusions

In this paper we consider the problem of inferring link loss rates from the application traffic between sensor nodes and the sink taken from a collection of aggregation

trees. Based on the data aggregation communication paradigm, we propose a practical loss performance inference algorithm using the standard Expectation-Maximization (EM) techniques. Our algorithm simply observes the application traffic of the network and handles well topology changes. Therefore, our approach is applicable not only to periodic data collection scenarios but also to events detection scenarios. Via simulations varying between different network scales, application scenarios, loss distributions, it can be observed that the algorithm converges fast and exhibits good scalability and stability.

7. References

[1] C. G. Vehbi and P. H. Gerhard, "Industrial Wireless SenSor Networks: Challenges, Design Principles, and Technical Approaches," *IEEE Transactions on Industrial Electronics*, Vol. 56, No. 10, October 2009, pp. 4258-4265.

[2] J. A. Stankovic, "When Sensor and Actuator Networks Cover the World," *ETRI Journal*, Vol. 30, No. 5, October 2008, pp. 627-633.

[3] A. Willig, "Recent and Emerging Topics in Wireless Industrial Communications: A Selection," *IEEE Transactions on Industrial Informatics*, Vol. 4, No. 2, May 2008, pp. 102-124.

[4] E. Fasolo, M. Rossi, J. Widmer and M. Zorzi, "In-Network Aggregation Techniques for Wireless Sensor Networks: A Survey," *IEEE Wireless Communications*, Vol. 14, No. 2, 2007, pp. 70-86.

[5] K. S. Low, W. N. N. Win and M. J. Er, "Wireless Sensor Networks for Industrial Environments," *Proceedings of International Conference on Computational Intelligence for Modelling, Control and International Conference on Automation and Intelligent Agents, Web Technologies and Internet Commerce*, Vienna, Vol. 2, 2005, pp. 271-276.

[6] G. J. McLachlan and T. Krishnan, "The EM Algorithm and Extensions," 2nd Edition, John Wiley and Sons, Inc, New York, 2008.

[7] R. Castro, M. Coates, G. Liang, R. Nowak and B. Yu, "Network Tomography: Recent Developments," Statistical Science, Vol. 19, No. 3, 2004, pp. 499-517.

[8] B. Tian, D. Nick, P. Francesco Lo and T. Don, "Network Tomography on General Topologies," *ACM SIGMETRICS Performance Evaluation Review*, Vol. 30, No. 1, 2002, pp. 21-30.

[9] S. Rost and H. Balakrishnan, "Memento: A Health Monitoring System for Wireless Sensor Networks," *Proceedings of* 3*rd Annual IEEE Communications Society on Sensor and Ad Hoc Communications and Networks*, Santa Clara, Vol. 3, 2006, pp. 575-584.

[10] X. Meng, T. Nandagopal, L. Li and S. Lu, "Contour Maps: Monitoring and Diagnosis in Sensor Networks," *Computer Networks*, Vol. 50, No. 15, 2006, pp. 2820-2838.

[11] N. Ramanathan, K. Chang, R. Kapur, L. Girod and E. Kohler, "Sympathy for the Sensor Network Debugger," *Proceedings of ACM Conference on Embedded Networked Sensor Systems* (*SenSys*), San Diego, 2005, pp. 255-267.

[12] S. Gupta, R. Zheng and A. M. K. Cheng, "ANDES: An Anomaly Detection System for Wireless Sensor Networks," *Proceedings of IEEE International Conference on Mobile Adhoc and Sensor Systems* (*MASS*), Pisa, 2007, pp. 1-9.

[13] A. Meier, M. Motani, S. Hu and K. Simon, "DiMo: Distributed Node Monitoring in Wireless Sensor Networks," *Proceedings of International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, Vancouver, 2008, pp. 117-121.

[14] G. Hartl and B. Li, "Loss Inference in Wireless Sensor Networks Based on Data Aggregation," *Proceedings of International Symposium on Information Processing in Sensor Networks* (*IPSN*), Berkeley, 2004, pp. 396-404.

[15] Y. Mao, F. R. Kschischang, B. Li and S. Pasupathy, "A Factor Graph Approach to Link Loss Monitoring in Wireless Sensor Networks," *IEEE Journal on Selected Areas in Communications*, Vol. 23, No. 4, 2005, pp. 820-829.

[16] Y. Li, W. Cai, G. Tian and W. Wang, "Loss Tomography in Wireless Sensor Network Using Gibbs Sampling," *Proceedings of European Conference on Wireless Sensor Networks*, Delft, 2007, pp. 150-162.

[17] E. Shakshuki and X. Xing, "A Fault Inference Mechanism in Sensor Networks Using Markov Chain," *Proceedings of the* 22*nd International Conference on Advanced Information Networking and Applications*, GinoWan, 2008, pp. 628-635.

[18] H. X. Nguyen and P. Thiran, "Using End-To-End Data to Infer Lossy Links in Sensor Networks," *Proceedings of IEEE International Conference on Computer Communications* (*INFOCOM*), Barcelona, 2006, pp. 2205-2216.

[19] K. Liu, M. Li, Y. Liu, M. Li and Z. Guo, "Passive DiaGnosis for Wireless Sensor Networks," *Proceedings of ACM Conference on Embedded Network Sensor Systems* (*SenSys*), New York, 2008, pp. 113-126.

[20] B. Wang, W. Wei, W. Zeng and R. P. Krishna, "Fault Localization Using Passive End-to-End Measurement and Sequential Testing for Wireless Sensor Networks," *Proceedings of IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, Rome, 2009, pp. 225-234.

[21] A. Woo, T. Tong and D. Culler, "Taming the Underlying Challenges of Reliable Multi-Hop Routing in Sensor NetWorks," *Proceedings of ACM Conference on Embedded Networked Sensor Systems*, Los Angeles, 2003, pp. 14-17.

[22] B. Krishnamachari, D. Estrin and S. Wicker, "The Impact of Data Aggregation in Wireless Sensor Networks," *Proceedings of the* 22*nd International Conference on Distributed Computing Systems*, Vienna, 2002, pp. 575-578.

Wireless Sensor Network, **2010, 2, 520-527**
doi:10.4236/wsn.2010.27064 Published Online July 2010 (http://www.SciRP.org/journal/wsn)

Developing a Multi-Layer Strategy for Securing Control Systems of Oil Refineries

Musaria K. Mahmood[1], Fawzi M. Al-Naima[2]
[1]*Department of Electrical Engineering, College of Engineering, Tikrit University, Tikrit, Iraq*
[2]*Department of Computer Engineering, College of Engineering, Nahrain University, Baghdad, Iraq*
E-mail: *Musariaoja@yahoo.com, fawzi.alnaima@ieee.org*
Received May 12, 2010; *revised May* 21, 2010; *accepted May* 27, 2010

Abstract

The energy industry and in particular the Oil Refineries are extremely important elements in Iraq's infrastructure. A terrorist attack on one oil refinery will have a catastrophic impact on oil production and the whole economy. It can also cause serious damage to the environment and even losses of human lives. The security of information systems and industrial control systems such as Supervisory Control and Data Acquisition (SCADA) systems and Distributed Control System (DCS) used in the oil industry is a major part of infrastructure protection strategy. This paper describes an attempt to use several security procedures to design a secure, robust system for the SCADA and DCS systems currently in use in the North Oil Refinery in the city of Baiji located in northern Iraq.

Keywords: DCS, SCADA, Security, Encryption, Internet, Public Key, DMZ, Data Security

1. Introduction

The increased use of computer network in control systems and the use of internet as communication backbone have brought benefits in communication and for the process control in general. The ability to share information, making decision concerning production stoking and distribution has been greatly improved [1]. New information technologies have recently been introduced for control system based on open standards like Visual Basic, Java, Open Database Connectivity (ODBC), Ethernet communication and finally the use of internet based control system. Internet or web based control system uses some internet protocols in the application layer to monitor and control plants at a distance. Older control systems were based on closed protocols implemented by vendors of SCADA, DCS, and Programmable Logic Controller (PLC) systems. However, the new common technology and the use of internet introduced many challenges in the area of security, such as cyber threats [2].

The North Oil Refinery in Baiji is a large industrial complex located on various sites scattered around the city of Baiji, north of Baghdad. This complex comprises four basic big refineries: 1) Salah Aldine-1, 2) Salah Aldine-2, 3) North, and 4) Chemical Products. All these refineries are functioning under the same direction and are technically sported by unified sections. Each refinery unit has many DCS systems controlling several chemical processes. These processes are independent of each other and are situated at different geographical locations. Each DCS acts as standalone control system with its private communication network and local control room (LCR).

DCS systems in operation are based on new information technology (IT) with open standards such as TCP/IP protocols using fiber optic Ethernet for data transmission between field and LCR. The refineries direction begins a new project to connect all DCS systems to a Central Control Room (CCR) via intranet. Local server of each DCS will be connected to system server located at the CCR. System server will be connected to the Internet as part of web based control system of the refineries.

Multi-layered security system is suggested for Baiji Oil Refineries. This strategy of security will be based on the "ring of defense" around each local network of each DCS and also around the overall corporate intranet (which will be similar to SCADA system) [2].

The suggested security system can also be used for other SCADA systems in general like those used in Electrical Power Plants, Water Purification and many other Oil Industries.

2. Networking DCS's

The model to be adopted in this work is to consider each

DCS system as a node or "station" of a global SCADA system in the perspective of grouping all LCR of all DCS's of the North Refineries in the CCR. CCR will be the central control of the overall SCADA system. Each DCS will keep its local server, local database, and local LCR, and will be connected to the CCR via appropriate communication links.

The connection of local servers to the system server (or web server) of SCADA system located at the CCR is made via corporate intranet as depicted in **Figure 1**. Communication links between LCR sites and CCR can be accomplished by fiber optics, coaxial cable or wireless links regarding several factors related to topology, geography, and required bandwidth. Reliability, availability and redundancy are major factors for the design of network topology. System database and all local databases use the same standard SQL database. The system database is located near the system server at the interface zone between corporate intranet and the internet. This location is chosen at the border of the system to minimize the risk of unauthorized intruder. The topology definition and the reliability of the communication network are not a part of this work.

3. Security Strategy

The similarity of the new SCADA system and the ordinary IT system leads to the adoption of the same security approach used in computer communication network. The principal of networks separation into "rusted network" and "non-trusted network" fit perfectly for securing SCADA systems [3]. The ring of defense (multi-layer security system) will be implemented using various techniques, which are cited as the principal countermeasure for securing SCADA [4]. The goal of such model will be the repetition of countermeasures to enhance the global system security as shown in **Figure 2**.

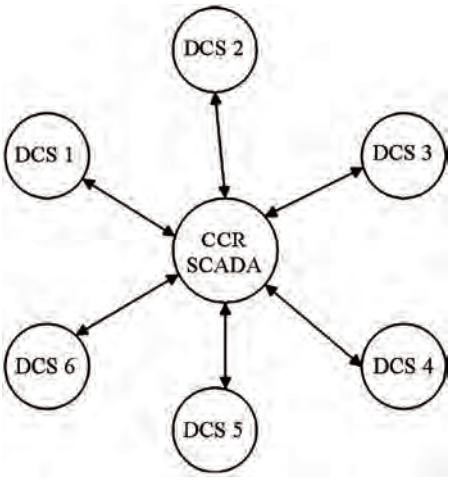


Figure 1. Networking DCS's.

Each layer will acts against one or more of possible security challenges. There exist many cyber threats facing control systems as cited in **Table 1** [5]. Each security layer will be part of the defense on depth against threats.

The possible attack will be considered from the external non trusted network (the Internet) in the direction of the internal trusted network (the corporate intranet).

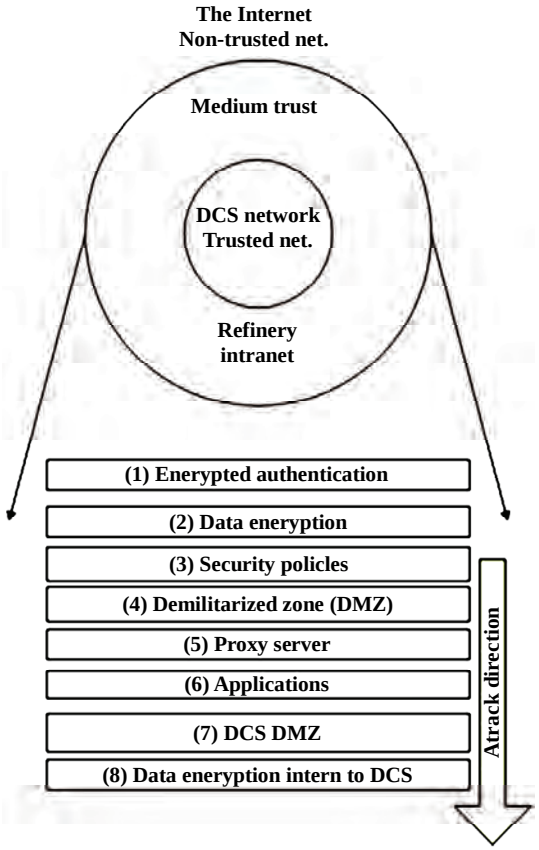


Figure 2. Ring of defense.

Table 1. Correlation of security threats and properties.

Properties	Cyber threats	Importance
Message confidentiality	Eavesdropping	Low
	Traffic analysis	Low
Message integrity	Message modification	High
	False message injection	High
Message freshness	Message replay	High
Availability	Denial-of-service	Middle
	Malicious codes	Middle
Source authentication	Masquerade	High
	Unauthorized access	Middle

The compatibility of countermeasures like encryption standards, firewall, antivirus, intrusion detection techniques will be accomplished by the use of standardized techniques. Each DCS system can be isolated immediately from the SCADA network and then be controlled locally if needed. The connection or the disabling of connection of one DCS can be made locally or from the CCR by some fixed authorization protocol. The connection of DCS system to the CCR server and then to the internet will be subject to several layers of security defense rings as illustrated in **Figure 3**. Between each DCS system and the corporate intranet network a demilitarized zone (DMZ) is used where the local server and database of DCS reside. The global SCADA system is connected to the internet via another DMZ. To login from browser to the SCADA intranet an original process of encrypted authentication and digital signature is implemented.

3.1. Encrypted Authentication

It is the first security stage for an external client. An original authentication process is developed based on asymmetric key encryption algorithm as shown in **Figure 4**. The RSA algorithm is used to encrypt the password two times for the implementation of authentication and the secrecy of password [3,6].

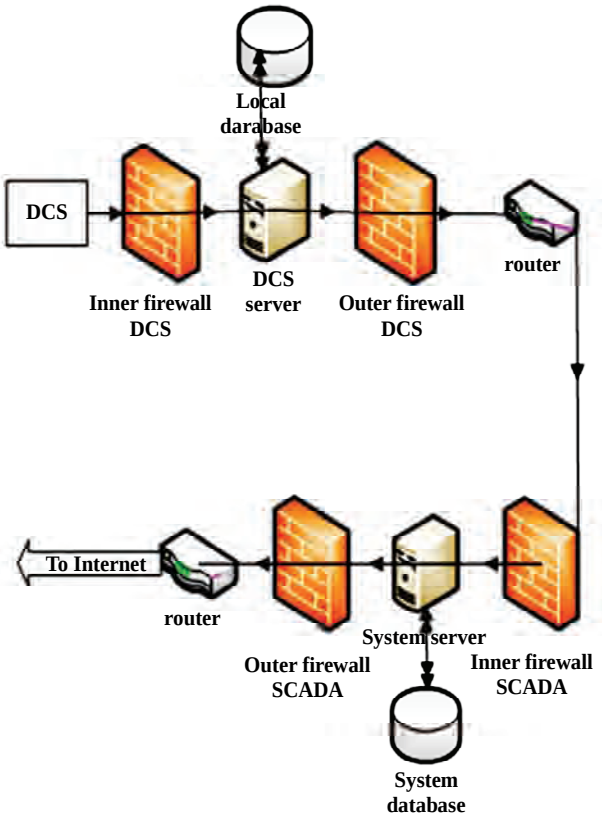


Figure 3. Security layers from DCS to internet.

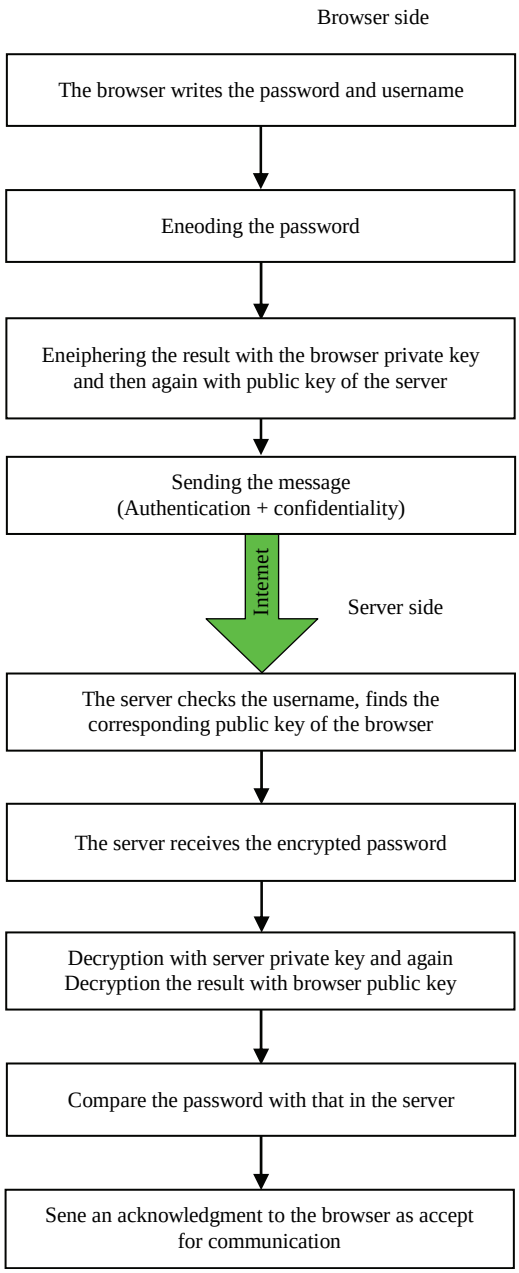


Figure 4. Authentication process.

The implementation of public key algorithm was made with two key groups (e_1,d_1), and (e_2,d_2), where e_1 is the public key of the first key group, d_1 is the private of browser (personal private key), e_2 is the second public key, and d_2 is the private key of server.

The method of defining these keys, and the encryption algorithm is as follows:

Choose two large prime numbers (p and q), let:

$$n = p \times q \quad (1)$$

The function $\beta(n)$ is the number of numbers less than (n) with no factors in common with (n).

Choose an integer number e, ($e < n$) relatively prime to $\beta(n)$. Find second integer (d) such that

$\{e \times d \bmod \beta(n) = 1\}$

Then, the Public Key = (e, n)

The Private Key = (d, n)

The encryption function of RSA for a plain text (M) is (C) such that:

$$C = M^{d} \bmod n \quad (2)$$

Then, the recovered plain text (M) is:

$$M = C^{e} \bmod n \quad (3)$$

If (e) is used instead of (d) for the encryption of equation (2), then (d) will be used for the decryption in Equation (3).

The password will be encoded using ASCII code or other encoding system. Each byte will be encrypted two times: the first with (d_1) and the second with (e_2) This will ensure both secrecy and authentication of the password. At the server side, the reverse process is executed. The received cipher password will be first decrypted using (d_2) and then (e_1) in order to get the original plain text password.

The result will be compared to usernames table in the server. If matching is achieved then the server will open a communication channel and begin responding to browser requests.

3.2. Data Encryption

To protect SCADA system already connected to the internet and its data from unauthorized accesses many algorithms for data protection exist. In this work two encryption algorithms are used to encrypt data.

1) *Encryption*-1(*security layer* 2): This encryption algorithm used to encrypt data over the Internet. We propose to use the Secure And Fast Encryption Routine (SAFER+), which is one of the known symmetric key algorithms that accomplishes requirements for real time data encryption. The SAFER+ was designed by James Messey for Cylink Corporation in 1998 as the new algorithm of SAFER family (safer-64, safer-128...) [7]. It was one of the candidates of the Advanced Encryption Standard (AES) chosen for its good hardware-software tradeoff orientation, simplicity, high throughput compared to other algorithms, and low memory requirement [8], [9]. Also, this algorithm brought attention recently by the use in as security measure in Bluetooth and wireless communication [10,11].

A detailed analysis of maximum bit rate for one DCS proves that SAFER+ fits well for the encryption of data over Internet to get the required real data transmission. SAFER+ is published with three options, 128 bits, 192, bits and, 256 bits key lengths. All three options are used to encrypt 16-byte plaintext. The plaintext block then passes through R rounds of encryption where R is determined by the key length chosen for encryption in the following manner:

- If key length = 128 bits then R = 8 rounds.
- If key length = 192 bits then R = 12 rounds.
- If key length = 256 bits then R = 16 rounds.

Our choice was the adoption of SAFER+ with 128 bits key length as other key lengths were found with certain key weaknesses [12]. Also, by increasing key length more computation time is needed related to rounds number which will affect the assumption of real data transmission in control system.

This algorithm will be implemented as hardware in the server side and portable software in the client side. If there is wireless link between two sites interior to the corporate intranet, SAFER+ must be used with hardware encryption decryption processes. SAFER+ algorithm is divided into three blocks: the key scheduling, encryption process, and decryption process. Programming is accomplished using Turbo Pascal.

2) *Encryption*-2 (*security layer* 8): This algorithm is used to encrypt data over the network of one DCS. This algorithm is not a part of this work but we can use the American Gas Association (AGA) standard which is proven to be a good choice for data over DCS network when dealing with protocols such as MODBUS or DNP3 [13]. In this work we found that securing data over the DCS network has no significant impact to enhance the security of overall system because all DCS system are well physically protected in isolated areas and we take care about data security when data circulate outside DCS network by other countermeasures. This encryption procedure is mentioned here merely for other case studies with other compromises and challenges related to other situations.

3.3. SAFER+ Encryption/Decryption

Giving the 16-byte key (128 bits), SAFER+ begin by calculating a set of 17 keys each with same length 16 bytes. The calculation uses sample arithmetic and logic functions like bit rotation, bit-by-bit exclusive-or of bytes, modulo 256 addition of bytes, and selection byte process.

The 17 sets of keys are used in the encryption rounds. Two keys are used for each round. Round (i) uses key K_{2i-1} and K_{2i}. At the end of 8 rounds key K_{17} is used for the output operation which is the output transformation. The output transformation uses bit-by-bit exclusive-or of bytes and modulo 256 byte addition as shown in **Figure 5**. At the reception the reverse process is used for decrypting the cipher text. Beginning by the input transformation, Key K_{17} is used. The input transformation uses same functions of the output transformation but with modulo 256 subtraction of bytes instead of addition of bytes. Each encryption round (i) begin by the bit-by-bit Xor of bytes, and modulo 256 addition of bytes for the

key K_{2i-1} to the input 16-byte of the round **Figure 6**. The 16-bytes result are then fed to a layer of nonlinear function. The value x of byte j is converted to 45^x mod 257 for j = 1, 4, 5, 8, 9, 12, 13, and 16 (with the convention that when x = 128, then 45^{128} mod 257 = 256 is represented by a 0). The value x of byte j is converted to $\log_{45}(x)$ for j = 2, 3, 6, 7, 10, 11, 14, and 15 (with convention that when x = 0, then $\log_{45}0$ = 128). The output of the nonlinear layer is then subject to same addition and Xor operation similar to the first block with key K_{2i}. At the end of round (i), a block of matrix multiplication is used. The 16 bytes are multiplied by matrix T in mod 256 arithmetic. T is a 16 × 16 predefined matrix.

The operations in the decryption round are simply conducted in reverse order to the operations from the encryption round.

The first operation in the decryption round (i), is to post multiply the 16-byte round input by matrix T^{-1}, which is the modulo 256 inverse of T to give the 16-byte result (S). The first round sub key $K_{16-2i+2}$, is then "subtracted" from (S) in the manner that the round sub key bytes 1, 4, 5, 8, 9, 12, 13, and 16 are subtracted modulo 256 from the corresponding bytes of (S), while round sub key bytes 2, 3, 6, 7, 10, 11, 14, and 15 are added bit-by-bit modulo 2 to the corresponding bytes of (S). The 16-byte result is then processed nonlinearly in the manner that the value x of byte j is converted to $\log_{45}$(x) for bytes j = 1, 4, 5, 8, 9, 12, 13, and 16 (with the convention that when x = 0, $\log_{45}0$ = 128). For j = 2, 3, 6, 7, 10, 11, 14, and 15, the value x is converted to 45^x mod 257 (with the convention that when x = 128, 45^{128} mod 257 = 256 round sub key $K_{16-2i+1}$, is then "subtracted" from the 16-byte result in the manner that the round sub key bytes 1, 4, 5, 8, 9, 12, 13, and 16 are added bit-by-bit modulo 2 to the corresponding input bytes. Sub key bytes 2, 3, 6, 7, 10, 11, 14, and 15 are subtracted modulo 256 from the corresponding input bytes to produce the 16-byte output of the round.

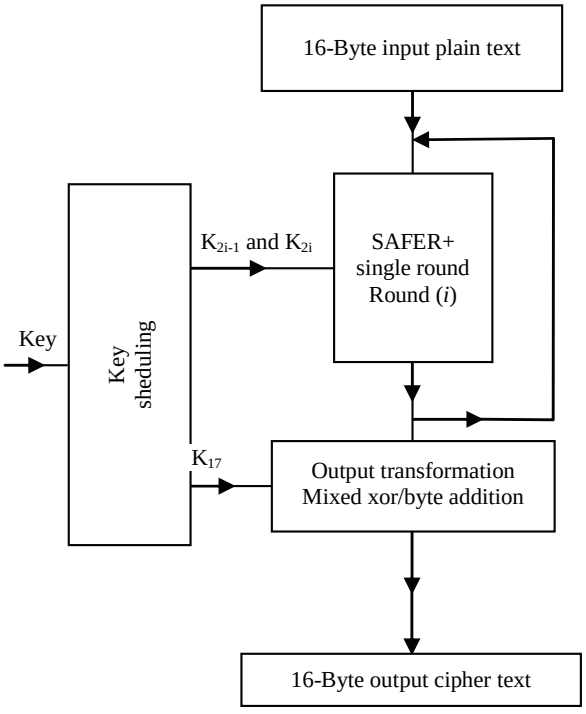


Figure 5. SAFER+ encryption process.

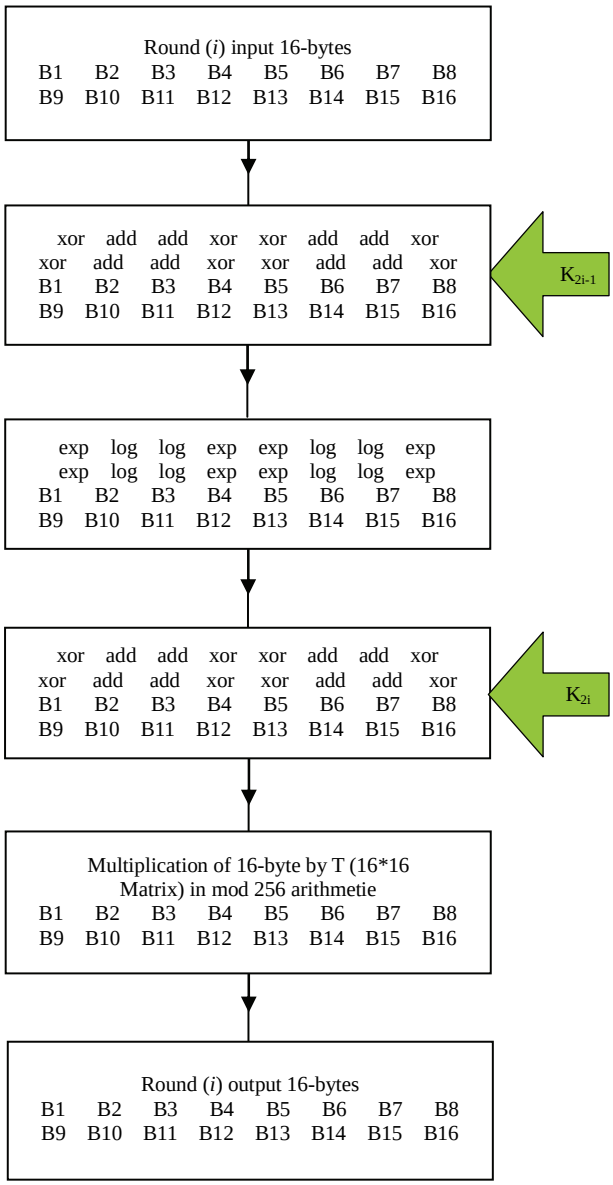


Figure 6. Encryption round (*i*).

3.4. Demilitarized Zone (DMZ)

DMZ is a good technique for securing communication network based on the principal of "networks separation strategy", between trusted network (like DCS LAN) and

another network with less level of trust [1,3]. In the DMZ the server and database reside in safe place, see **Figure 3**. Proxy server and application action like antivirus are also deployed in the DMZ. In this work a double protection is found by the use of two DMZs. The first is as a security interface between SCADA system and the external non-trusted Internet (security layer 4), while the second is between the DCS trusted network and the less trusted corporate intranet for SCADA (security layer 7). DMZ is a zone between an inner firewall and an outer firewall. This firewall, properly configured, can protect passwords, IP addresses, and files. Firewall acts as a filter that permits the data to enter from certain ports and blocks others. At the DMZ output a router is used as border router to route information to correct destination.

3.5. Security Policies

Security policies must be fixed by the security committee of the enterprise. Correct procedure of effective policies can reduce violation of security rules. Training personnel at the vocation of security can increase the defense in depth strategy. Developing documentation and well defining the access authorization will be a very active point. The choice of password must be as random as possible with at least ten characters, including symbols and numbers.

In this work all the required information of the group of DCS in the North Oil Refinery will be available in the system server in the master DMZ (between corporate network and the internet).

4. Simulation Results

RSA and SAFER+ encryption algorithms are implemented in Turbo Pascal language. As example to show how the encrypted authentication will be calculated. Let the password be chosen as the random word 'playmyaudio'. Encoding this password with simple code from 01 for (a) to 26 for (z), will give:

$$M = 16, 12, 1, 25, 13, 25, 1, 21, 4, 9, 15.$$

C will be composed by the same number of 11 bytes. Consider for our example the following values for the keys are taken:

$$e_1 = 13,\ d_1 = 53,\ e_2 = 37,\ d_2 = 17,\ n = 77$$

Using equation (2) twice with browser private key d_1 and then with server public key e_2 the cipher text will be:

$$(16^{53} \bmod 77)^{37} \bmod 77 = 60$$

$$(12^{53} \bmod 77)^{37} \bmod 77 = 45$$

For the same for other bytes, we have:

$$C = 60, 45, 1, 58, 13, 58, 1, 21, 37, 53, 15.$$

At the server, the received cipher text will be converted back to the original password using Equation (3) twice:

$$(60^{13} \bmod 77)^{17} \bmod 77 = 16$$

$$(45^{13} \bmod 77)^{17} \bmod 77 = 12$$

So on for other bytes to get the original password:

$$M = 16, 12, 1, 25, 13, 25, 1, 21, 4, 9, 15.$$

The use of double RSA encryption procedure will gives both authentication and secrecy.

SAFER+ algorithm uses matrix to represent the 17-sub keys (17 × 16 matrix).given the secrete key (K_1) as series of 16 bytes, the 16 other keys, each of 16 bytes will be generated. The calculation of each round is made using iterative process and many preprogrammed procedure (for Xor, Exp, log..).

As an example for the encryption/decryption process, let the 16 byte user selected input key (secrete key) be: 41, 35, 190, 132, 225, 108, 214, 174, 82, 144, 73, 241, 241, 187, 233, 235.

After the execution of key schedule procedure of **Figure 7**, for 128 bit key we have the (17 × 16) matrix which represents the 17 sub keys each of length 16 bytes (128 bits). Each key is represented by a row, where row (1) represents K_1 (the secrete key), to row (17) which represents K_{17}, this will give the matrix (K) shown in **Figure 8**. The resulting 17 sub keys and the plain text are used to generate the cipher text as presented in **Figure 5**.

The plain text block input (16 byte plain text) = 179, 166, 219, 60, 135, 12, 62, 153, 36, 94, 13, 28, 6, 183, 71, 222.

The resulting cipher text will be = 224, 31, 182, 10, 12, 255, 84, 70, 127, 13, 89, 249, 9, 57, 165, 220.

The predefined bias matrix (B) is given as (16 × 16) matrix as input to generate sub key matrix [7]. This procedure will be used to encrypt input data by block of 16 bytes at a time. The schedule procedure will be executed next time when the secrete key is changed to find the new matrix (K).

5. Conclusions

The use of open standard improves the control operations by improving the possibility of interconnecting many systems from different vendors together without restrictions in term of standards. Action for the security of SCADA system must be up to date regarding the continued advances in information technology. Care must be taken for the use of latest version of antivirus, and intrusion detection programs. Symmetric encryption algorithms can be used to encrypt data over internet for SCADA system if the required bit rate is accomplished. SAFER+ algorithm can accomplish the real data transmission requirement. Using SAFER+ give us the possibility to encrypt data by software or hardware implementations.

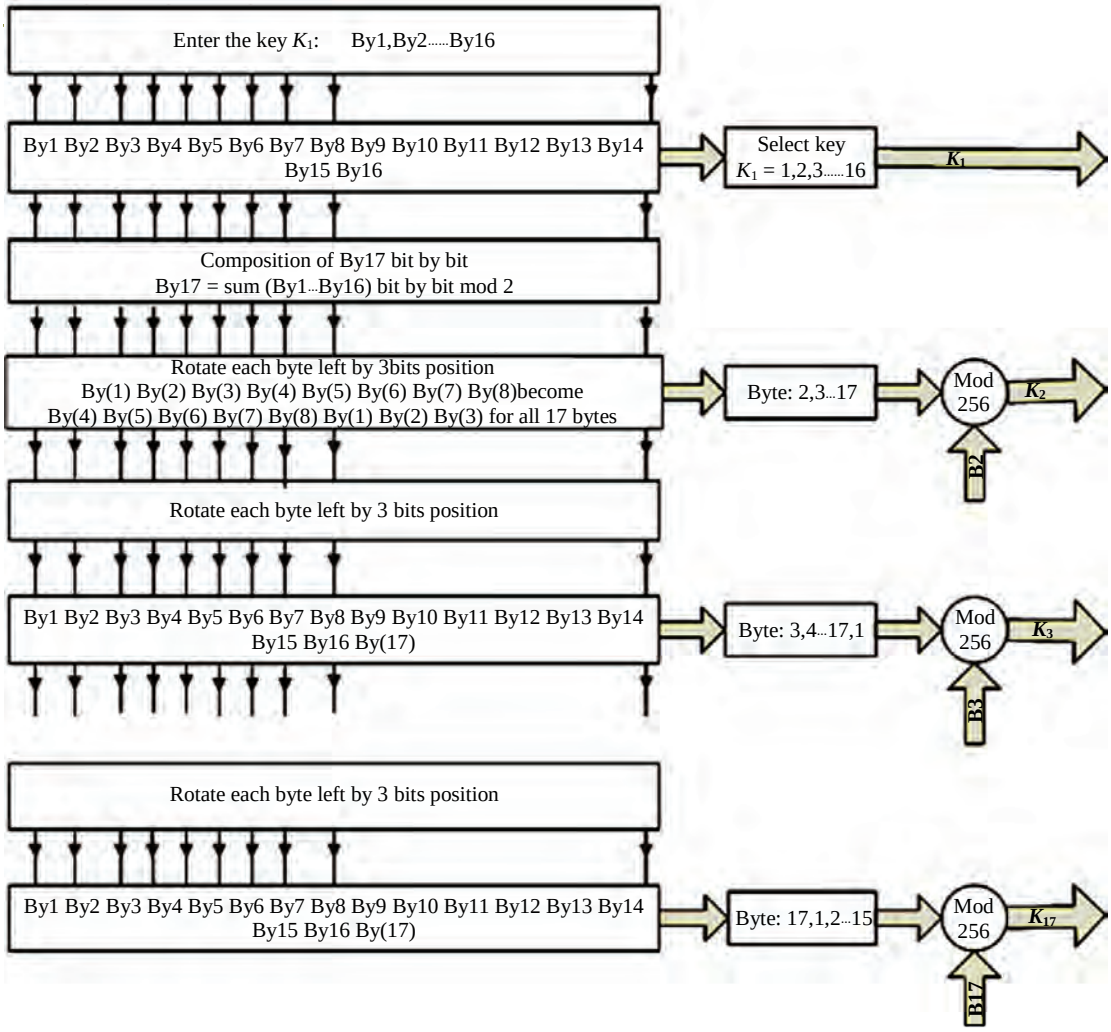


Figure 7. Sub keys generation process.

$$K = \begin{bmatrix}
41 & 35 & 190 & 132 & 225 & 108 & 214 & 174 & 82 & 144 & 73 & 241 & 241 & 187 & 233 & 135 \\
95 & 140 & 213 & 201 & 6 & 109 & 133 & 156 & 73 & 129 & 66 & 88 & 55 & 119 & 11 & 35 \\
155 & 204 & 34 & 225 & 28 & 64 & 236 & 49 & 74 & 22 & 114 & 92 & 224 & 214 & 2 & 135 \\
147 & 134 & 176 & 54 & 199 & 141 & 87 & 219 & 38 & 162 & 98 & 167 & 109 & 138 & 186 & 230 \\
123 & 29 & 255 & 9 & 250 & 122 & 240 & 218 & 65 & 124 & 92 & 57 & 59 & 43 & 149 & 127 \\
96 & 204 & 15 & 93 & 122 & 189 & 245 & 243 & 244 & 52 & 219 & 76 & 177 & 210 & 163 & 209 \\
56 & 190 & 201 & 32 & 12 & 248 & 157 & 109 & 168 & 81 & 214 & 221 & 102 & 105 & 53 & 81 \\
15 & 26 & 46 & 250 & 110 & 124 & 137 & 222 & 74 & 13 & 5 & 12 & 134 & 18 & 149 & 185 \\
207 & 61 & 251 & 224 & 179 & 66 & 183 & 96 & 253 & 60 & 37 & 78 & 211 & 15 & 222 & 9 \\
68 & 215 & 94 & 56 & 94 & 49 & 35 & 230 & 120 & 133 & 111 & 195 & 97 & 68 & 203 & 173 \\
78 & 156 & 190 & 181 & 130 & 222 & 6 & 159 & 38 & 59 & 53 & 238 & 123 & 180 & 138 & 107 \\
221 & 238 & 152 & 211 & 241 & 232 & 248 & 255 & 101 & 167 & 37 & 36 & 134 & 238 & 244 & 243 \\
55 & 111 & 165 & 66 & 105 & 237 & 214 & 179 & 86 & 233 & 14 & 214 & 53 & 115 & 165 & 201 \\
34 & 65 & 73 & 224 & 185 & 205 & 107 & 140 & 123 & 117 & 55 & 254 & 4 & 179 & 82 & 236 \\
212 & 162 & 91 & 17 & 41 & 175 & 56 & 251 & 163 & 238 & 13 & 249 & 50 & 54 & 180 & 74 \\
51 & 1 & 59 & 215 & 18 & 174 & 202 & 253 & 151 & 91 & 101 & 89 & 167 & 98 & 148 & 104 \\
127 & 111 & 186 & 111 & 62 & 132 & 35 & 230 & 184 & 23 & 199 & 252 & 186 & 75 & 227 & 149
\end{bmatrix}$$

Figure 8. Sub key matrix (*K*).

The use of computer network protection can be very useful in SCADA like DMZ, and network segmentation from the trust point of view. Layered strategy is a good method to secure SCADA, DCS and control systems in general. The use of different layered actions against cyber threats makes the defense in depth active. Authentication is one of most used and active countermeasure for security in control systems. The use of RSA public key algorithm for encrypt password will ensure security and authenticity.

6. References

[1] R. L. Krutz, "Securing SCADA Systems," Wesley Publishing, Inc., Indianapolis, 2006.

[2] J. Pollet, "Developing a Solid SCADA Security Strategy," *Proceedings of the Second ISA/IEEE Sensors for Industry Conference*, Houston, November 2002, pp. 148-156.

[3] M. Bishop, "Introduction to Computer Security," Addison-Wesley, Boston, 2005.

[4] J. Nordlander, "What is Special about SCADA System Cyber Security," Master Dissertation, Royal Institute of Technology, Sweden, 2009.

[5] H. Lim, S. Hong, M. S. Choi, S. J. Lee, T. W. Kim, S. W. Lee and B. N. Ha, "Security Protocols Against Cyber Attacks in the Distribution Automation System," *IEEE Transactions on Power Delivery*, Vol. 25, No. 1, 2010, pp. 448-455.

[6] T. Davis, "RSA Encryption," October 2003. http://www.geometer.org/mathcircles

[7] J. Messy, H. Kh and K. Kuregian, "Nomination of SAFER+ as a Candidate Algorithm for the AES," 1998. http://csrc.nist.gov/archive/aes/round1

[8] A. Schubert and W. Anheier, "Efficient VLSI Implementation of Modern Symmetric Block Ciphers," 6*th IEEE International Conference on Electronics, Circuit and Systems*, Pafos, Vol. 2, September 1999, pp. 757-760.

[9] S. Mukherjee, D. Ganguly and S. Naskar, "A New Generation Cryptographic Technique," *International Journal of Computer Theory and Engineering*, Vol. 1, No. 3, August 2009, pp. 284-287.

[10] I. S. Ashour, "Online Data and Voice Encryption System Based on FPGA," 24*th National Radio Science Conference*, Cairo, National Republican Senatorial Committee, 2007, pp. 1-7.

[11] D. Sharmila and R. Neelaveni, "Performance Analysis of SAFER+ and Triple DES Security Algorithms for Bluetooth Security System," *International Journal of Computer Science and Network Security*, Vol. 9, No. 2, 2009, pp. 74-87.

[12] J. Kelsey, B. Schneier and D. Wagner, "Key Schedule Weaknesses in SAFER+," *Second Advanced Encryption Standard Candidate Conference*, Rome, 1999, pp. 155-167.

[13] A. West, "Securing DNP3 and Modbus with AGA12-2J," *IEEE Power and Energy Society General Meeting*: *Conversion and Delivery of Electrical Energy in the* 21*st Century*, Pittsburgh, 20-24 July 2008, pp. 1-4.

Wireless Sensor Network, **2010, 2, 528-537**
doi:10.4236/wsn.2010.27065 Published Online July 2010 (http://www.SciRP.org/journal/wsn)

Contention-Based Beaconless Real-Time Routing Protocol for Wireless Sensor Networks*

Chao Huang, Guoli Wang
School of Information Science & Technology, Sun Yat-Sen University, Guangzhou, China
E-mail: *superchaohuang@gmail.com, isswgl@mail.sysu.edu.cn*
Received April 12, 2010; *revised May* 4, 2010; *accepted May* 17, 2010

Abstract

This paper presents a novel real-time routing protocol, called CBRR, with less energy consumption for wireless sensor networks (WSNs). End-to-End real-time requirements are fulfilled with speed or delay constraint at each hop through integrating the contention and neighbor table mechanisms. More precisely, CBRR maintains a neighbor table via the contention mechanism being dependent on wireless broadcast instead of beacons. Comprehensive simulations show that CBRR can not only achieve higher performance in static networks, but also work well for dynamic networks.

Keywords: Wireless Sensor Network, Real-Time Routing Protocol, Contention-Based Scheme, Beaconless

1. Introduction

WSNs are comprised of tiny, low power sensor nodes, which are densely deployed in a monitored area and collaborate to forward the sensed data to a base station through multiple hops. WSNs can be widely used in many applications, such as environmental monitoring, military surveillance, disaster recovery, and healthcare related applications [1]. However, many mission-critical applications require that WSNs can guarantee the satisfied quality of service (QoS), especially with real-time constraints. e.g., in forest fire detection, the abnormal temperature information should be delivered to a base station as soon as possible. Otherwise, the outdated data will be irrelevant and even have negative effects on the applications.

WSNs have particular features different from other wireless networks [2-4]. Firstly, each sensor node has severe resource constraints on bandwidth, memory, processing capability, and especially energy. Secondly, the number of sensor nodes is very large and the nodes are always densely deployed. Thirdly, it has been recognized that the network topologies of WSNs may change constantly due to sleep policy, node mobility, node failure, and so on. The features mentioned above will pose challenges to fulfill the real-time requirements.

Due to the high routing overhead and the poor scalability of global routing decisions, several real-time routing protocols [5-7] utilize the localized geographic information to heuristically find a satisfied path to a target. These protocols are mostly governed by the conventional geographic routing schemes, in which each node has to periodically broadcast beacons for obtaining the accurate information of its neighbors and to store the information in its neighbor table for routing decisions. Although the neighbors' information can be collected by broadcasting beacons, some drawbacks inevitably arise. Firstly, redundant beacons can lead to more energy consumption and higher communication cost. Secondly, only one neighbor or very small subset of neighbors can take part in routing decisions, but each node should store all of neighbors' information in a neighbor table. Therefore, the utilization of a neighbor table is much inefficient, and the unused neighbors' information still occupies some of the limited memory. Thirdly, when a network topology changes constantly, the collected neighbor information will be outdated quickly, which in turn leads to higher packet miss ratio, and nodes should broadcast beacons more frequently to update the neighbor tables. It inevitably incurs not only longer delay, but also consumes significant energy. Furthermore, frequently broadcasting beacons can result in more serious collisions thus higher packet miss ratio.

Contention-based beaconless scheme [8-12] has been proposed to handle with the issues abovementioned. The main idea is that when a node has a DATA packet to be

*This work is supported by Nature Science Foundation of China under Grant 60775055 and Guangdong Nature Science Foundation of China under Grant 06023190.

transmitted, it first broadcasts a contention request, and one neighbor with the shortest wait delay is selected as the most suitable next-hop forwarder. The contention-based scheme is completely stateless and reactive because each node can on-line select an available next-hop forwarder without any prior knowledge of its neighbors. This results in less energy consumption. Furthermore, it can be found that the current forwarder can obtain the information of the selected next-hop forwarder during a contention procedure. Although the information of the selected neighbor is not utilized in the scheme, it can benefit to future routing decisions. The character indicates that the approach for selecting the information of neighborhoods is energy efficient.

This paper presents a Contention-based Beaconless Real-time Routing protocol for WSNs, called CBRR. CBRR aims at fulfilling end-to-end real-time requirements with less energy consumption through integrating the contention and neighbor table mechanisms. The advantage of our approach is to remove the limitations of the beacon-based routing schemes, especially, in dynamic networks. Three remarkable aspects of CBRR are emphasized as follows. Firstly, the key to CBRR is that the information in a neighbor table is obtained through a contention mechanism without beacons, and CBRR employs the contention mechanism to select a new forwarder only when there is no an available neighbor, which meets the real-time requirements, in the neighbor table. Secondly, the neighbor table allows CBRR to select a next-hop forwarder by direct unicast or contention-based forwarding. As a consequence, the end-to-end real-time requirements can be fulfilled with speed or delay constraint at each hop. Thirdly, CBRR can collect the information from the two-hop neighbors during the wireless broadcasting, and the two-hop neighbor table can be helpful for further meeting the real-time requirements in the two-hop range.

The rest of the paper is organized as follows. Section 2 outlines the related works. Section 3 presents the proposed real-time routing protocol. Section 4 reports the experimental results.

2. Related Works

Recently, real-time routing protocol has aroused much research interests in WSNs. SAR [13] was the first QoS protocol for WSNs. It could find multiple paths between a source and a target with different priority in terms of energy efficiency and fault tolerance. EQR [14] tried to find a least-cost yet energy efficient path for real-time data, and to maximize the throughput for non-real-time data with a class-based queue model, simultaneously. However, both SAR and EQR were based on global routing decisions, which result in much higher routing overheads. Therefore, localized geographic routing decisions have been popular employed in a variety of real-time routing protocols. SPEED [5] was designed to provide soft end-to-end real-time guaranty with a desired delivery speed, $S_{setpoint}$, which could support the real-time communication between a source and a sink, through a combination of feedback control and non-deterministic geographic forwarding. Only the node, whose progress speed was higher than $S_{setpoint}$, could be selected as the next-hop forwarder. MMSPEED [6] was an extension to SPEED with additional service differentiations both in the timeliness and reliability domains. RPAR [7] fulfilled the end-to-end real-time requirements with low energy consumption by dynamically adapting transmission power and assigning one-hop velocity. Unfortunately, these localized routing protocols had to broadcast beacons to collect the neighbors' information and were inevitably suffered from the aforementioned drawbacks.

Contention-based beaconless scheme has been developed to address the issues aforementioned. IGF [8] only allowed the nodes within the 60-degree sector towards the sink to participate in the contention. The wait delay of each available node was determined by the combination of the progress distance to a sink, the remaining energy of the node and a random delay. GeRaF [9,10] used the busy tone to avoid collisions and replaced the wait delay function by time slots, which were assigned to different regions of forwarding areas. SGF [11] introduced a forwarding scheme through integrating the contention mechanism and gradient for unexpected node/link failures in highly dynamic networks. OGF [12] combined the contention mechanism with a neighbor table to forward packets in slowly dynamic networks. OGF was similar to our approach, but it did not consider the real-time constraints.

3. Main Results

Our design goal is to fulfill the end-to-end real-time requirements with less energy consumption for WSNs. Two versions of CBRR protocol are proposed, such as CBRR-OneHop and CBRR-TwoHop, which are based on one-hop neighbor table and two-hop neighbor table, respectively. CBRR-OneHop protocol is the foundation of CBRR, and CBRR-TwoHop protocol is an extension to CBRR-OneHop. Before describing CBRR protocol, we first introduce the relevant definitions and assumptions.

3.1. Definitions and Assumptions

Denote source node and sink node as $\boldsymbol{S}$ and $\boldsymbol{D}$. Let $d(i, j)$ and $Delay_i^j$ be the Euclidean distance and the delay between node i and node j, respectively. $Deadline(D)$ is the required deadline of a DATA packet for D and is

carried in the header of the DATA packet. R_c is the radio range.

Neighbor Set of Node *i*: NS_i is the set of nodes within the radio range of node *i*. Formally, $NS_i = \{j \mid d(i, j) \le R_c\}$.

Contention Candidate Set of Node *i*: CCS_i is the set of nodes which are in NS_i and closer to *D* than node *i*. Formally, $CCS_i = \{j \mid d(i,D) > d(j,D), j \in NS_i\}$.

Relay Speed: $Speed_i^j$ is the velocity between node *i* and *j*, which is the ratio of the distance to the delay between node *i* and *j*. Formally, $Speed_i^j = d(i, j) / Delay_i^j$.

Required Speed: $Speed_i^D$ is the velocity, which meets the real-time requirements. It is the ratio of the distance between node *i* and *D* to the remaining time of a DATA packet's deadline. Formally, $Speed_i^D = d(i,D) / (Deadline(D) - T_{now})$, where T_{now} is the current time.

Estimated Hop: Hop_i^D is the estimated hop number between node *i* and *D*, which is the ratio of the distance between node *i* and *D* to R_c. Formally, $Hop_i^D = d(i,D) / R_c$.

Required Delay: $Delay_i^D$ is the one-hop delay, which meets the real-time requirements. It is the ratio of the remaining time of the packet's deadline to the estimated hop. Formally, $Delay_i^D = (Deadline(D) - T_{now}) / Hop_i^D$.

Speed Constraint: Speed constraint is used for each node to find an available neighbor, whose relay speed is no less than the required speed, in its neighbor table. If having an available neighbor, the node can send the DATA packet to the neighbor by direct unicast. Otherwise, the node has to invoke a contention procedure to select a new next-hop forwarder.

Delay Constraint: Delay constraint is used for each node to determine whether its contention candidates can take part in the contention or not. If there is an available neighbor, whose one-hop delay is no less than the required delay, in the neighbor table of the contention candidate, this candidate can be admitted to participate in the contention. Otherwise, the candidate is forbidden to do that.

Assume that each node is aware of its position, and it has the same radio range and initial energy with a unique ID and an out-of-band busy tone.

3.2. CBRR-OneHop Protocol

CBRR-OneHop protocol is the foundation of CBRR, and it consists of five components, such as unicast forwarding policy, contention forwarding policy, contention function with wait delay, delay estimation and one-hop neighbor table, which is created or updated only during a contention procedure. When a source intends to send its first DATA packet, the node has to employ the contention forwarding policy to select a next-hop forwarder because the neighbor table of each node has not been created. After the first DATA packet is successfully forwarded to the sink, each node, which is on the previous forwarding path, has at least one neighbor in its one-hop neighbor table. Therefore, the first DATA packet can be regarded as a special packet for the path discovery. In the future forwarding, the one-hop neighbor table can play an important role to help with selecting an available next-hop forwarder, which meets the speed constraint or delay constraint, by the unicast forwarding policy or the contention forwarding policy.

- **Unicast Forwarding Policy**

As shown in **Figure 1**, the unicast forwarding policy can utilize the speed constraint to determine that a DATA packet is forwarded by direct unicast or invoking the contention forwarding policy. If the neighbor table has an available neighbor, which meets the speed constraint, the current forwarder selects the neighbor as the next-hop forwarder and unicasts the DATA packet to the neighbor directly. Otherwise, the current forwarder has to invoke the contention forwarding policy to select a new next-hop forwarder.

- **Contention Forwarding Policy**

If the unicast forwarding policy fails to unicast a DATA packet, the contention forwarding policy can employ ERTS-ECTS-DATA-ACK handshake to select a new next-hop forwarder. The delay constraint is used to determine whether a contention candidate can take part in the contention or not. ERTS is an extension to RTS with additional information including the positions of the current forwarder and the sink, and the required delay.

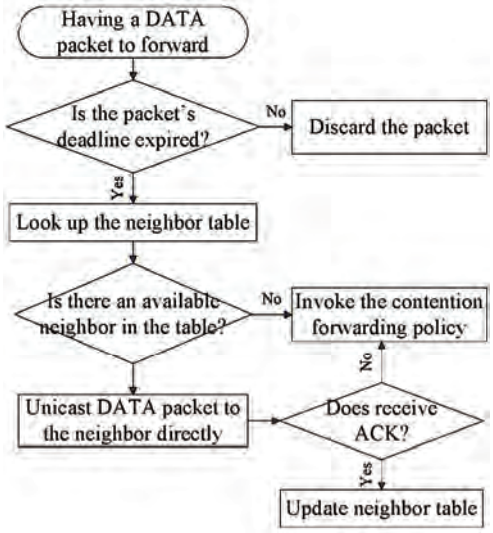


Figure 1. Unicast forwarding procedure.

ECTS is also an extension to CTS with additional position of CTS's sender. Busy tone is an out-of-band signal which is used to avoid multiple candidates taking part in the contention, simultaneously. The current forwarder and its contention candidates work as follows.

As shown in **Figure 2**, the current forwarder first broadcasts an ERTS packet to start a contention, and then waits for receiving an ECTS packet. If overhearing the ECTS packet, the current forwarder should send a busy tone immediately, and continues receiving the ECTS packet. After finishing receiving the ECTS, the current forwarder can unicast the DATA packet to the selected next-hop forwarder directly. When the current forwarder finishes receiving the ACK packet sent by the next-hop forwarder, it inserts the information of the new forwarder including the ID, position and one-hop delay, or updates the one-hop delay, in its neighbor table. If the current forwarder fails to select a next-hop forwarder, it just discards the DATA packet.

The contention candidate's contention procedure is shown in **Figure 3**. After receiving the ERTS packet, each contention candidate has to determine whether to participate in the contention or not according to its one-hop neighbor table. If its neighbor table has an available neighbor, which meets the speed constraint, the candidate can be assigned higher contention priority with short wait delay, T_{short}. If none of the neighbors in its one-hop neighbor table can meet the speed constraint, the candidate is forbidden to take part in the contention. Otherwise, if the neighbor table is empty, which means that the candidate has never been the forwarder, the candidate is also admitted to take part in the contention but with long wait delay, T_{long}. The candidate with the shortest wait delay can first respond an ECTS packet thus to win the qualification of the new forwarder. Other candidates, which receive the busy tone during their wait delay, must quit the contention immediately. When finishing receiving the DATA packet, the new forwarder should respond an ACK packet to the current forwarder.

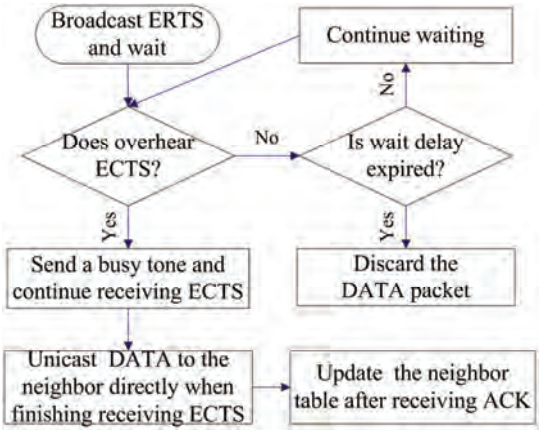


Figure 2. Current forwarder's working procedure.

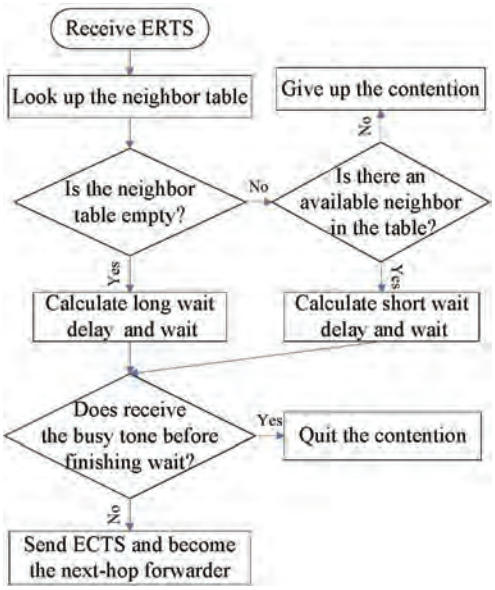


Figure 3. Contention candidate's contention procedure.

It needs to be pointed out that although the delay constraint never takes the wait delay into account, it does not affect the real-time performance because the selected neighbor is the most suitable forwarder, which has the shortest wait delay.

- **Contention Function and Wait Delay**

Contention function is the key factor to the contention-based beaconless scheme because it determines the wait delay of each contention candidate. Due to real-time constraint and energy efficiency, our contention function takes into account the combination of the progress distance toward a sink, the remaining energy of the node and the number of packets waiting in the output queue. For node i, its contention priority is computed as follows.

$$P_i = (q_i + 1)(\alpha d_i / R_c + \beta E_i / E_t + \gamma(1 - q_i / Q_t) + \eta r_i) \quad (1)$$

where P_i is the contention priority; d_i is the progress distance towards a sink; E_i is the remaining energy and E_t is the initial total energy; q_i is the number of packets waiting in the queue and Q_t is the total queue size; r_i is a random value between 0 and 1; α, β, γ and η are weights assigned to distance, energy, queue and random value, respectively, and meet $\alpha + \beta + \gamma + \eta = 1$.

Using the contention function, we can calculate the wait delay of T_{short} and T_{long} as follows.

$$T_{short} = SIFS + 0.5SIFS(1 - P_i) \quad (2)$$

$$T_{short} = SIFS + SIFS(1 - P_i) \quad (3)$$

where *SIFS* is Short Inter Frame Spacing and is defined as 10 μs in IEEE 802.11 standard [15]. From (2) and (3), it can be guaranteed that T_{short} is smaller than T_{long}.

- **Delay Estimation**

Delay estimation is responsible to calculate the one-

hop delay between two neighborhoods. The current one-hop delay is evaluated based on two timestamps. One is the time $T_{arriving}$ when the received DATA packet enters the tail of the output queue. Another is the time T_{ACK} when the node receives the ACK packet responded by the next-hop forwarder. The time interval between these two timestamps is employed to characterize the one-hope delay here.

The new one-hop delay can be calculated by the combination of the currently measured delay and the previous one-hop delay as follows.

$$Delay_{new}^{onehop} = \mu(T_{ACK} - T_{arriving}) + (1-\mu)Delay_{previous}^{onehop} \quad (4)$$

- **One-Hop Neighbor Table**

As shown as in **Table 1**, NeighborID and NeighborPosition are obtained through an ECTS packet; OneHopDelay can be computed from (4). Counter is used to record how many times a neighbor has been a next-hop forwarder.

One-hop Neighbor table is managed as follows.

1) If the newly selected forwarder is not in the neighbor table, the current forwarder has to add the information of the new forwarder in its neighbor table after receiving the ACK packet sent by the new forwarder. Otherwise, the current forwarder needs to update the OneHopDelay of the neighbor in its neighbor table.

2) If the Counter in the entry of a neighbor is up to MaxCount, the neighbor is forbidden to take part in the current forwarding, and its Counter should be cleared to zero. MaxCount is the maximal times that a node can be a next-hop forwarder. If the network topology is stable, a neighbor in the neighbor table can always be valid so as to be the next-hop forwarder continuously. In this case, the node not only consumes away its energy quickly, but also deprives other neighbors of being a next-hop forwarder. Therefore, MaxCount is used to balance the network load thus to prolong the lifetime of WSNs.

3.3. CBRR-TwoHop Protocol

CBRR-TwoHop protocol is an extension to CBRR-OneHop protocol. It can depend on wireless broadcast to collect the information of the two-hop neighborhoods for building two-hop neighbor table. With the help of two-hop neighbor table, CBRR-TwoHop is able to further meet the real-time requirements in the two-hop range.

- **Two-Hop Delay Estimation**

As shown in **Figure 4**, node A, B and C are regarded as the last-hop forwarder, the current forwarder and the next-hop forwarder, respectively. While node B unicasting a DATA packet to node C, node A can also overhear the DATA packet, which carries the position information of node C.

In order to obtain the two-hop delay, each node needs to create a Sent_DATA table for recording the DATA packets, which have been sent by the node successfully. As shown in **Table 2**, the record of a DATA packet includes the items of source's ID, sequence number and Received_Time, which is the timestamp that the DATA packet enters the tail of the output queue. With the help of the Sent_DATA table, the two-hop delay estimation between node A and C can be obtained as follows.

1) After receiving the ACK packet sent by node B, node A needs to add the relevant information of the DATA packet, which has been sent to node B previously, in its Sent_DATA table.

2) While node B unicasting a DATA packet to node C, node A can overhear the packet so as to look up its Sent_DATA table to see if A has sent this DATA packet previously. If true, the current two-hop delay between node A and C is approximated by the interval between the Received_Time in A's Sent_DATA table and the time T_{now} when node B finishes transmitting the DATA packet to node C. After estimating, node A needs to delete the entry of this DATA packet from its Sent_DATA table for reducing the storage overhead. However, if the DATA packet is not in the Sent_DATA table, node A has to ignore the packet and to do nothing.

We also compute the new two-hop delay by the combination of the currently measured delay and the previous two-hop delay as follows.

$$Delay_{new}^{twohop} = \mu(T_{now} - T_{\mathrm{Received\_Time}}) + (1-\mu)Delay_{previous}^{twohop} \quad (5)$$

- **Two-Hop Delay Estimation**

The two-hop neighbor table extends the one-hop neighbor table with additional information of the two-hop neighbor including ID, position and two-hop delay, as

Table 1. Structure of one-hop neighbor table.

NeighborID	NeighborPosition	OneHopDelay	Counter

Table 2. Structure of Sent_DATA table.

SourceID	SequenceNo.	Received_Time

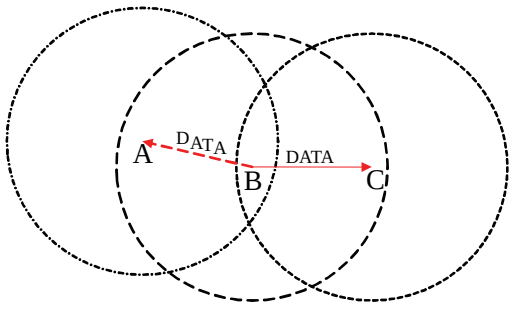


Figure 4. An example of wireless broadcast.

shown in **Table 3**. The ID and position of the two-hop neighbor are obtained through a DATA packet, and Two-Hop Delay can be computed from (5). Other items are the same as those in one-hop neighbor table.

It should be noted that if a node fails to select a next-hop forwarder, the node maybe encounters a void around it. In this case, the node will not unicast the received DATA packet, and its last-hop forwarder can never overhear the packet. As a result, the information of the two-hop neighbor should be null in the two-hop neighbor table of the last-hop forwarder. For avoiding more invalid contentions, if the two-hop table has more than two records, which have null information of the two-hop neighbor, the node maybe has a serious void problem in its two-hop range thus to be forbidden to take part in the latter contentions.

- **Forwarding policy**

The forwarding policy of CBRR-TwoHop is similar to that of CBRR-OneHop. The difference is that CBRR-TwoHop needs to employ the speed constraint between the two-hop neighborhoods. If there is a one-hop neighbor, which the two-hop relay speed between the current forwarder and the two-hop neighbor is no less than the required two-hop speed, in the two-hop neighbor table, the current forwarder can uincast the DATA packet to the one-hop neighbor directly. Otherwise, the current forwarder has to invoke the contention forwarding policy with the two-hop delay constraint to select a next-hop forwarder.

4. Experimental Studies

To validate the CBRR protocol proposed in the paper, CBRR-OneHop and CBRR-TwoHop are compared with SPEED. Experimental studies are conducted by means of J-Sim simulator [16], which is an open-source, component-based network simulation environment and is developed entirely in Java.

Table 3. Structure of two-hop neighbor table.

OneHop ID	OneHop Position	OneHop Delay	TwoHop ID	TwoHop Position	TwoHop Delay	Counter

Table 4. Parameters for experiments.

Parameter	Values	Parameter	Values
Network Size	200m×200m	MAC Layer	802.11
Node Number	200	Initial Energy	100 J
Radio Range	40 m	Bandwidth	2 Mbps
Packet Size	512 bytes	Send/Receive/Idle power(mW)	660/395/35

4.1. Simulation Setting

We randomly choose four source nodes in the left of network, and two sink nodes in the right of network. We test the above protocols in two network topologies:

- **Static network**, where the topology is invariable including packet generation rate and node density scenarios.
- **Dynamic network**, where the topology is changeable due to node mobility or sleep policy.

We choose constant bit rate (CBR) traffic and set CBR at 2 packets/s in all experiments except in the packet generation rate scenario. Unless specially noted, all parameters for experiments are set as shown in **Table 4**.

4.2. Performance in Static Networks

In static networks, the position of each node is not changeable. Therefore, we set the interval of beacon broadcast at 10 s in SPEED. Two scenarios are evaluated in the static networks including packet generation rate and packet size.

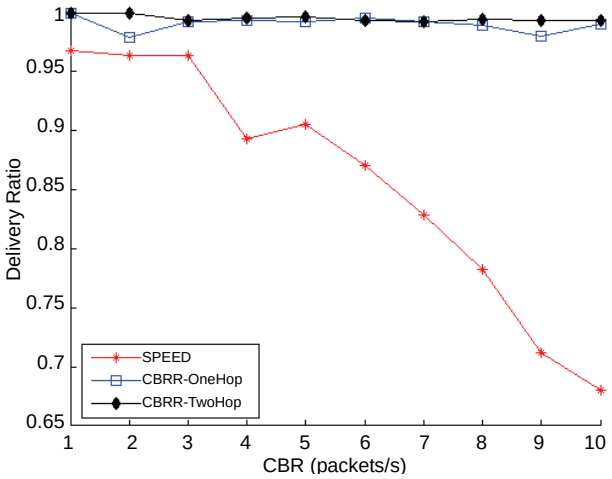
- **Impact of packet generation rate**

The comparative results between CBRR and SPEED are plotted in **Figure 5**. It can be seen that the both CBRR protocols achieve nearly 100% delivery ratio (**Figure 5(a)**) and stable end-to-end delay around 0.05s (**Figure 5(b)**). All these in CBRR contribute to the Routing/MAC cross-layer design, which can timely collect the state information of wireless channel thus to avoid more collisions during the forwarding procedures. In contrast, higher packet generate rate may bring forth more packet collisions thus lead to higher packet miss ratio (**Figure 5(a)**) and longer delay (**Figure 5(b)**) in SPEED. In **Figure 5(c)**, SPEED consumes the average energy about two times more than those of the two CBRR protocols due to its beacon broadcasting. It is worth noting that the average energy consumption of each protocol decreases slowly as the packet generation rate increasing. The reason is that when the packet generation rate is small, most nodes are always kept in idle state, whose energy consumption is the main part of the total consumed energy. However, more and more packets have been forwarded thus can lead to lower average energy consumption as the generate rate increasing. Furthermore, **Figure 5** illustrates that the overall performance of CBRR-TwoHop is little better than that of CBRR-OneHop because the two-hop neighbor table can be helpful for further meeting the real-time requirements in the two-hop range.

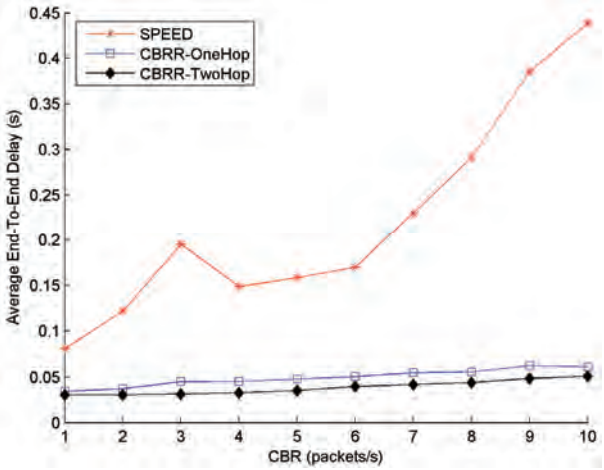
- **Impact of packet size**

Larger DATA packets may lead to higher probability of the collisions between a DATA packet and other packets, such as RTS/ERTS, CTS/ECTS, ACK or beacon. **Figure 6** illustrates the comparative results between

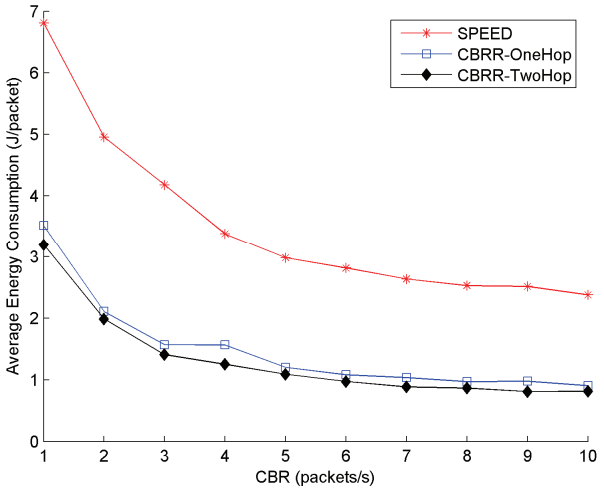
CBRR and SPEED in the packet size scenario. It can be observed in **Figure 6(a)** that the delivery ratio of SPEED drops quickly as the packet size increasing, but the packet size has very little impact on the two CBRR pro-



(a) Packet Delivery Ratio

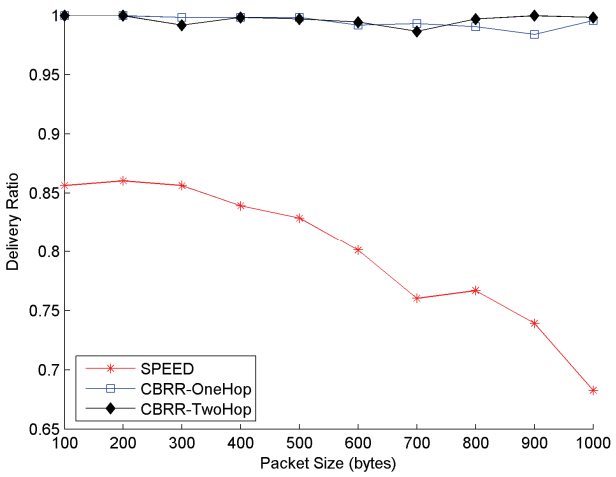


(b) Average End-to-End Delay

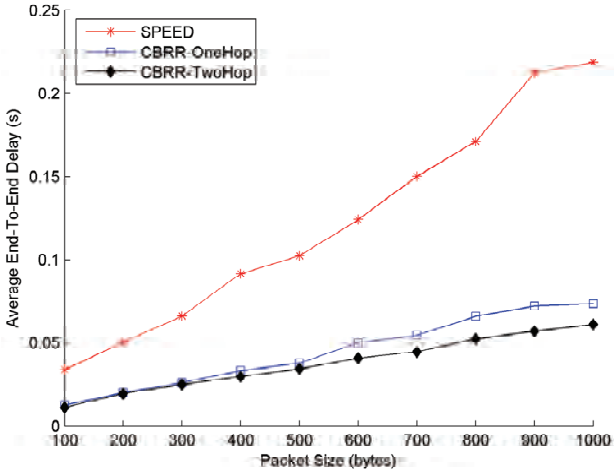


(c) Average Energy Consumption

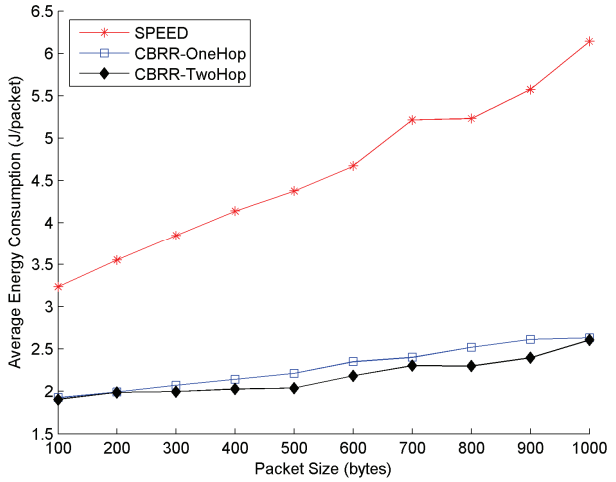
Figure 5. Impact of packet generation rate.



(a) Packet Delivery Ratio



(b) Average End-to-End Delay



(c) Average Energy Consumption

Figure 6. Impact of packet size.

tocols for their nearly 100% delivery ratio. The results may indicate that the collisions between DATA packets and beacons are much more severe than others between DATA and RTS/ERTS, CTS/ECTS or ACK packets. As a result, with larger packet size, more serious packet collisions can lead to longer end-to-end delay (**Figure 6(b)**) and more energy consumption (**Figure 6(c)**) than those of the two CBRR protocols.

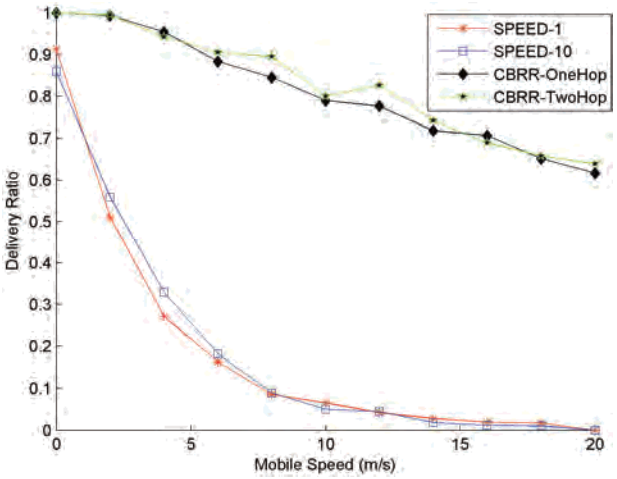
It can be viewed from above experimental results that the two CBRR protocols have much better performance than SPEED in the static networks. Furthermore, CBRR-TwoHop can outperform than other two protocols due to the help of the two-hop neighbor table, which can be helpful for further meeting the real-time requirements in the two-hop range. In addition, it can also be suggested that broadcasting beacons can aggravate the packet collisions thus to degrade the performance of the beacon-based routing protocols.

4.3. Performance in Dynamic Networks

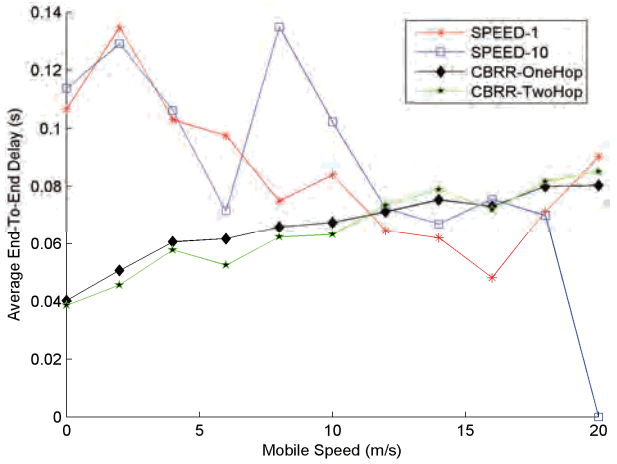
WSNs are highly dynamic networks and their topologies can change constantly due to node mobility, sleep policy, node failure, and so on. In the following experiments, we compare CBRR with SPEED in node mobility and sleep policy scenarios. In addition, we set the frequency of broadcasting beacons at 1 s and 10 s, respectively, for SPEED to timely collect fresh information of the neighborhoods.

- **Impact of node mobility**

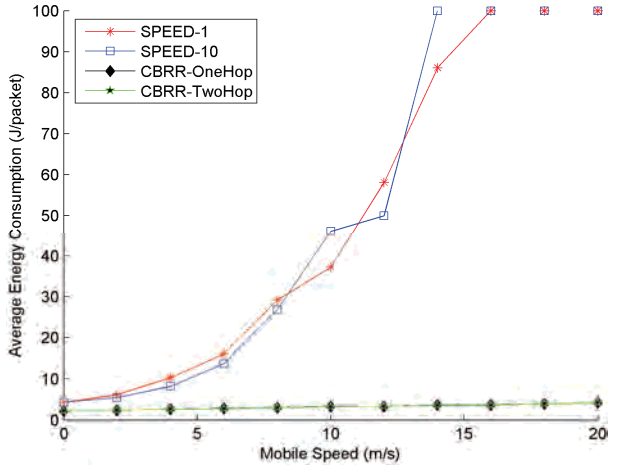
We adopt the Random Waypoint mobility model with zero pause time in the experiment. **Figure 7** plots the comparative results between the two CBRR protocols and SPEED in the node mobility scenario. **Figure 7(a)** illustrates that the delivery ratio of SPEED drops much more quickly, which shows that SPEED can hardly work when the mobile speed is higher than 10 m/s. This is because the information of the neighbor table is totally outdated thus to be useless for the forwarding. In contrast, although the two CBRR protocols drop more packets at higher mobile speed, they can still achieve about 60% delivery ratio at 20 m/s in contribution to the contention forwarding policy, which can on-line select a favorite next-hop forwarder. It can be observed in **Figure 7(b)** that the average end-to-end delay of SPEED becomes much instable for its extremely high packet miss ratio. However, at higher node mobility, the both CBRR protocols have little longer delay due to the failure of direct unicast. **Figure 7(c)** shows that the two CBRR protocols have very close performance, which consume far less energy than SPEED. It needs to point out that we set 100 as the maximum in **Figure 7(c)**, and the values of SPEED, which is plotted as 100, are actually more than 100. Furthermore, it can be observed in **Figure 7** that the performance of SPEED-1 is no better than that of SPEED-10, which suggests that higher frequency of beacon broadcasting can not improve the performance of SPEED in the node mobility scenario.



(a) Packet Delivery Ratio



(b) Average End-to-End Delay



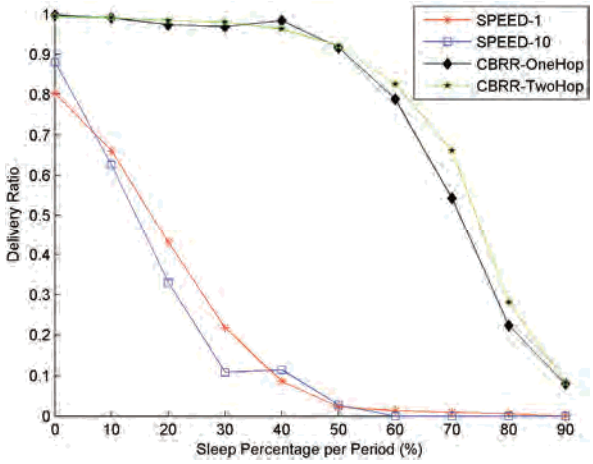
(c) Average Energy Consumption

Figure 7. Impact of node mobility.

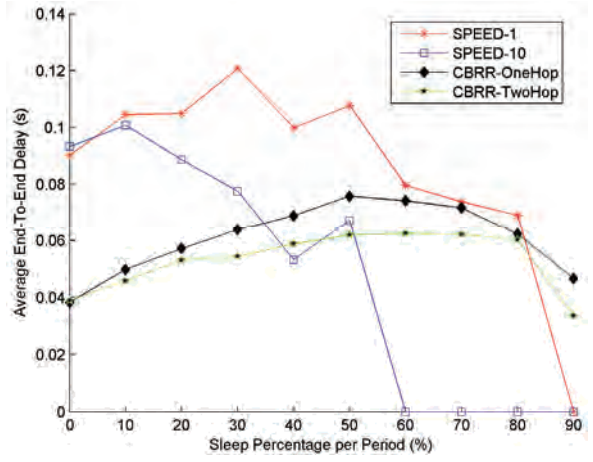
- **Impact of sleep policy**

In order to support energy conservation in WSNs, the most practical way is to use node sleep policy. Our sleep policy is designed as that the lifetime of each node is divided into several same periods, and each period includes an active sub-period and a sleep sub-period.

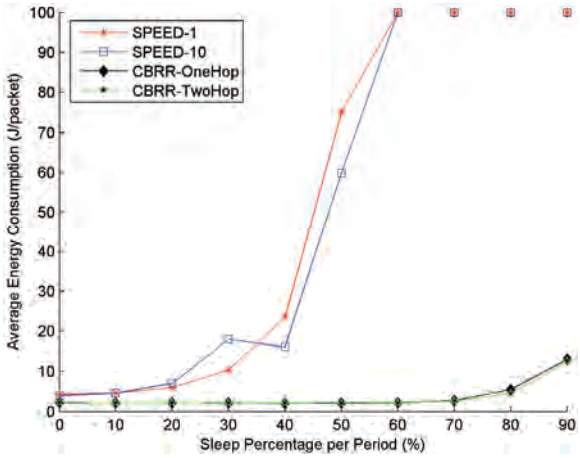
Figure 8 plots the comparative results between the two CBRR protocols and SPEED in the dynamic sleep policy scenario. Similar to the node mobility scenario, the sleep policy has far more negative impact on SPEED than those on the two CBRR protocols. The impact can be observed in **Figure 8(a)** that if the sleep percentage is larger than 50%, the delivery ratio of SPEED is approximate 0. The reason is that the next-hop forwarder, which is selected by SPEED in the neighbor table, would be usually in sleep state, and SPEED needs to retransmit more packets. However, the both CBRR protocols achieve more than 90% in delivery ratio at 50% sleep percentage. Although dropping more packets after 50%, the two CBRR protocols still outperforms SPEED very much. Other results are similar to those in the node mobility scenario as shown in **Figure 8(b)** and **Figure 8(c)**.



(a) Packet Delivery Ratio



(b) Average End-to-End Delay



(c) Average Energy Consumption

Figure 8. Impact of sleep policy.

It can be concluded from the above experimental results that CBRR is not only particularly suitable for the dynamic networks, but also has fair well performance than SPEED in the static scenarios with much less energy consumption. Furthermore, it can be observed that CBRR-TwoHop outperforms the other two protocols due to the help of the two-hop neighbor table. It also suggests that SPEED is totally not suitable for the dynamic networks, and increasing the frequency of beacon broadcasting can not improve but degrade the performance of the beacon-based routing protocols.

5. Conclusions

This paper presents a novel real-time routing protocol, CBRR, for WSNs. The point distinguishing our approach from the existing schemes is that CBRR collects the information of neighborhoods by the contention mechanism instead of beacons. This contribution can lead to more energy efficiency. In addition, it is notable that the end-to-end real-time requirements are well fulfilled with speed or delay constraint at each hop, which attributes to the combination of the contention and neighbor table mechanisms. The validity of CBRR is illustrated by means of experimental studies. It has been shown that CBRR can outperform SPEED in terms of delivery ratio, end-to-end delay and energy consumption, especially in dynamic networks.

Our future work is to conduct the theoretical analysis on the energy consumption of CBRR. It is also interested to investigate how to provide reliability in CBRR.

6. References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam and E. Cayirci, "Wireless Sensor Networks: A Survey," *Computer Networks*, Vol. 38, No. 4, March 2002, pp. 393-422.

[2] D. Z. Cheng and P. K. Varshney, "QoS Support in Wireless Sensor Networks: A Survey," *International Conference on Wireless Networks* (*ICWN*'04), LasVegas, Vol. 1, June 2004, pp. 227-233.

[3] Y. J. Li, C. S. Chen, Y. Q. Song and Z. Wang, "Real-time QoS Support In Wireless Sensor Networks: A Survey," *Proceedings of Seventh IFAC International Conference on Fieldbuses and Networks in Industrial and Embedded Systems*, Toulouse, November 2007, pp. 373-380.

[4] M. Younis, K. Akayya and A. Wadaa, "On Handling QoS Traffic in Wireless Sensor Networks," *Proceedings of the* 37*th Annual Hawaii International Conference on System Sciences*, Hawaii, January 2004, pp. 292-301.

[5] T. He, J. A. Stankovic, C. Y. Lu and T. F. Abdelzaher, "A Spatiotemporal Communication Protocol for Wireless Sensor Networks," *IEEE Transaction on Parallel Distributed Systems*, Vol. 16, No. 10, October 2005, pp. 995-1006,.

[6] E. Felemban, C. G. Lee and E. Ekici, "MMSPEED: Multipath Multi-SPEED Protocol for QoS Guarantee of Reliability and Timeliness in Wireless Sensor Networks," *IEEE Transactions on Mobile Computing*, Vol. 5, No. 6, June 2006, pp. 738-754.

[7] O. Z. Chipara, Z. M. He, G. L. Xing, Q. Chen and X. R. Wang, "Real-Time Power-Aware Routing in Sensor Networks," *Proceedings of* 14*th IEEE International Workshop on Quality of Service* (*IWQoS* 2006), New Haven, June 2006, pp. 83-92.

[8] T. He, B. M. Blum, Q. Cao, J. A. Stankovic, S. H. Son and T. F. Abdelzaher, "Robust and Timely Communication over Highly Dynamic Sensor Networks," *Real-Time Systems*, Vol. 37, No. 3, December 2007, pp. 261-289.

[9] M. Zorzi and R. R. Rao, "Geographic Random Forwarding (GeRaF) for Ad Hoc and Sensor Networks: Energy and Delay Performance," *IEEE Transactions on Mobile Computing*, Vol. 2, No. 4, October-December 2003, pp. 349-365.

[10] M. Zorzi and R. R. Rao, "Geographic Random Forwarding (GeRaF) for Ad Hoc and Sensor Networks: Multihop Performance," *IEEE Transactions on Mobile Computing*, Vol. 2, No. 4, October-December 2003, pp. 337-348.

[11] P. Huang, H. Y. Chen, G. L. Xing and Y. D. Tan, "SGF: A State-Free Gradient-Based Forwarding Protocol for Wireless Sensor Networks," *ACM Transaction on Sensor Networks*, Vol. 5, No. 2, March 2009, pp. 1-25.

[12] D. Z. Chen and P. K. Varshney, "On-Demand Geographic forwarding for Data Delivery in Wireless Sensor Networks," *Computer Communications*, Vol. 30, No. 14-15, October 2007, pp. 2954-2967.

[13] K. Sohrabi, J. Gao, V. Ailawadhi and G. J. Pottie, "Protocols for Self-Organization of a Wireless Sensor Network," *IEEE Personal Communications*, Vol. 7, No. 5, October 2000, pp.16-27.

[14] K. Akkaya and M. Younis, "Energy-Aware QoS Routing in Wireless Sensor Networks," *Cluster Computing*, Vol. 8, No. 2-3, July 2005, pp. 179-188.

[15] "IEEE Standard for Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, ISO/IEC 8802-11: 1999(E)," IEEE, August 1999.

[16] "DRCL J-Sim," 2005. http://j-sim.cs.uiuc.edu/index.html

Wireless Sensor Network, **2010, 2, 538-554**
doi:10.4236/wsn.2010.27066 Published Online July 2010 (http://www.SciRP.org/journal/wsn)

Scientific Research

Stable Sensor Network (SSN): A Dynamic Clustering Technique for Maximizing Stability in Wireless Sensor Networks

A. B. M. Alim Al Islam[1], Chowdhury Sayeed Hyder[2], Humayun Kabir[3], Mahmuda Naznin[3]
[1]*Purdue University, West Lafayette, Indiana, USA*
[2]*Michigan State University, Michigan, USA*
[3]*Bangladesh University of Engineering and Technology, Dhaka, Bangladesh*
E-mail: *razi_bd@yahoo.com, hydercho@msu.edu, mhkabir@cse.buet.ac.bd, mahmudanaznin@cse.buet.ac.bd*
Received April 26, 2010; *revised May* 12, 2010; *accepted May* 15, 2010

Abstract

Stability is one of the major concerns in advancement of Wireless Sensor Networks (WSN). A number of applications of WSN require guaranteed sensing, coverage and connectivity throughout its operational period. Death of the first node might cause instability in the network. Therefore, all of the sensor nodes in the network must be alive to achieve the goal during that period. One of the major obstacles to ensure these phenomena is unbalanced energy consumption rate. Different techniques have already been proposed to improve energy consumption rate such as clustering, efficient routing, and data aggregation. However, most of them do not consider the balanced energy consumption rate which is required to improve network stability. In this paper, we present a novel technique, Stable Sensor Network (SSN) to achieve balanced energy consumption rate using dynamic clustering to guarantee stability in WSN. Our technique is based on LEACH (Low-Energy Adaptive Clustering Hierarchy), which is one of the most widely deployed simple and effective clustering solutions for WSN. We present three heuristics to increase the time before the death of first sensor node in the network. We devise the algorithm of SSN based on those heuristics and also formulate its complete mathematical model. We verify the efficiency of SSN and correctness of the mathematical model by simulation results. Our simulation results show that SSN significantly improves network stability period compared to LEACH and its best variant.

Keywords: Network Stability Period, Clustering, Energy Consumption Rate

1. Introduction

With the emergence of highly dense fabrication technology and low production costs, wireless sensor networks (WSN) prove to be useful in myriad of diversified applications. In a typical WSN application, sensor nodes are scattered in a region from where they collect data to achieve certain goals. Data collection may be continuous, periodic or event based process. WSN must be very stable in some of its applications like security monitoring and motion tracking. Death of only one sensor node may disrupt coverage or connectivity and thus may reduce stability in this sort of applications. Therefore, all of the deployed sensor nodes in WSN must be active during operational lifetime. However, sensor nodes are generally equipped with one-time batteries and most of the batteries are of low energy. For this reason, each sensor node must efficiently use its available energy in order to improve the lifetime of WSN. Different techniques are used for efficient usage of this low available energy in a sensor node. Clustering is one of these most well-known techniques.

In some related research works, lifetime of WSN is considered as the time required for the last sensor node to die. Some other related research works also consider lifetime of WSN as the time required for half of the deployed sensor nodes to die. However, lifetime can be termed as stability period of WSN when it considers the time required for the first sensor node to die. There is an impact of efficient usage of available energy in a sensor node on stability period of WSN. This period is mainly

controlled by balanced energy consumption rate throughout the network. Therefore, we have to ensure balanced use of the available energy throughout the network along with the efficient use of the available energy in a sensor node to assure stability in WSN. In this paper, we propose a novel technique Stable Sensor Network (SSN), which ensures balanced use of the available energy throughout the network. It also achieves the efficient use of the available energy in a sensor node by exploiting clustering technique.

Clustering is a technique in which deployed sensor nodes are grouped into some clusters. Only one sensor node is solely responsible to communicate to the base station in a cluster. This sensor node is called cluster head and the remaining sensor nodes in the cluster are called followers. The followers collect data and send it to their corresponding cluster heads. The cluster heads aggregate its own data with the data received from its followers. Aggregated data is then sent to a sink to accomplish a specific goal. Cluster heads remain closer to their follower sensor nodes compared to the sink. It takes less energy to transmit data to the cluster head instead of the sink, which allows the sensor nodes to conserve more energy and live longer in WSN.

There are different clustering techniques already established for ad-hoc networks. However, those techniques cannot be directly used in WSN because of the fact that WSN imposes strict requirements on the energy efficiency than that ad-hoc networks do. As a result, many techniques have been proposed for clustering in WSN. Dynamic clustering techniques are more useful for WSN because of the dynamic variation in residual energies of the sensor nodes. LEACH [1] is one of the simple but popular dynamic clustering techniques used in WSN. LEACH rotates cluster headship very effectively among the sensor nodes of a network based only on some locally available information. However, LEACH does not consider the variation in residual energies of the sensor nodes when it selects the cluster heads. There are some already proposed modifications of LEACH to incorporate the variation. We consider this variation in more efficient way in SSN. We also incorporate the balanced use of residual energy in the network with the help of some heuristics in SSN. We formulate mathematical model for SSN. We prove that SSN has a significant improvement in network stability over LEACH and its best variant by simulation results.

In the next section, we briefly describe some related works. Then, we briefly describe the underlying approach of our technique along with its variants in Section 3. We present our novel clustering technique SSN with three heuristics and complete mathematical model in the next section. In Section 5, we evaluate SSN by simulation results. In last two sections, we conclude the paper shedding some lights on our future works.

2. Related Works

Several techniques have already been proposed to improve network lifetime in WSN. Clustering in one of the widely accepted techniques among them. Clustering is also used in wireless ad-hoc networks, mobile ad-hoc networks along with sensor networks. Several clustering techniques have already been introduced for partitioning nodes in these areas. Some of the early clustering techniques are—Hierarchical Clustering [2], Distributed Clustering Algorithm (DCA) [3], Spanning Tree (or BFS Tree) based Clustering [4], Clustering with On-Demand Routing [5], Clustering based on Degree or Lowest Identifier Heuristics [6], and Distributed and Energy-Efficient Clustering [7], Adaptive Power-Aware Clustering [8]. Some of the recently developed clustering techniques are PEGASIS (Power-Efficient Gathering in Sensor Information Systems) [9], Energy Efficient Clustering Routing [10], PEACH (Power Efficient And Adaptive Clustering Hierarchy) [11], Optimal Energy Aware Clustering [12], ACE (Algorithm For Cluster Establishment) [13], HEED (Hybrid Energy-Efficient Distributed Clustering) [14], PADCP (Power Aware Dynamic Clustering Protocol) [15], LEACH (Low- Energy Adaptive Clustering Hierarchy) [1], SEP (Stable Election Protocol) [16], and LEACH with Deterministic Cluster Head Selection [17].

In [9] PEGASIS introduces a near optimal chain-based protocol. Here, each node communicates only with a close neighbor and takes turns transmitting to the base station, thus reducing the amount of energy spent per round. It assumes that all nodes have global knowledge of the network and employ the greedy algorithm. It maps the problem of having close neighbors for all nodes to the traveling salesman problem. PEGASIS is a greedy chain protocol that is near optimal for a data-gathering problem in sensor networks. Greedy approach considers the physical distance only, ignoring the capability of a prospective node on the chain. Hence, a node with a shorter distance but less residual energy may be chosen in the chain and may die quickly.

In [10] a routing algorithm is proposed which combines hierarchical routing and geographical routing. The process of packet forwarding from the source nodes in the target region to the base station consists of two phases—inter-cluster routing and intra-cluster routing. For inter-cluster routing, a greedy algorithm is adopted to forward packets from the cluster heads of the target regions to the base station. For intra-cluster routing, a simple flooding is used to flood the packet inside the cluster when the number of intra-cluster nodes is less than a predetermined threshold. Otherwise, the recursive geographic forwarding approach is used to disseminate the packet inside target cluster, that is, the cluster head divides the target cluster into some sub-regions, creates the same number of new copies of the query packet, and then disseminates these copies to a central node in each sub reg-

ion. Like [9], it uses greedy algorithm based on the distance only but not on the capability or the residual energy. Although it deals with the optimal forwarding approach the criteria to choose the cluster heads optimally is not clearly explained.

PEACH [11] is a cluster formation technique based on overheard information from the sensor nodes. According to this approach, if a cluster head node becomes an intermediate node of a transmission, it first sets the sink node as its next hop. Then it sets a timer to receive and aggregate multiple packets from the nodes in the cluster set for a pre-specified time. It checks whether the distance between this node and the original destination node is shorter than that of between this node and already selected next hop node. If the distance is shorter, this node joins to the cluster of the original destination node and the next hop of this node is changed to the original destination node. PEACH is an adaptive clustering approach for multi-hop inter-cluster communication. However, it suffers from almost the same limitations of PEGASIS due to the choice of physical propinquity.

Optimal energy aware clustering [12] solves the balanced k-clustering problem optimally, where k signifies the number of master nodes that can be in the network. The algorithm is based on the minimum weight matching. It optimizes the sum of spatial distances between the member sensor nodes and the master nodes in the whole network. It effectively distributes the network load on all the masters and reduces the communication overhead and the energy dissipation. However, this research work does not consider of residual energy level while choosing a node as the master. Hence, the choice of the master or cluster head is far away from the optimal energy efficient distribution of the cluster heads.

ACE [13] is a distributed clustering algorithm which establishes clusters into two phases-spawning and migration. There are several iterations in each phase and the gap between two successive iterations follows uniform distribution. During the spawning phase, new clusters are formed in a self-elective manner. When a node decides to become a cluster head, it will broadcast a message to its neighbors to become its followers. During migration phase, existing clusters are maintained and rearanged, if required. Migration of an existing cluster is controlled by the cluster head. Each cluster head will periodically poll all of its followers to determine which could be the best candidate to elect as a new leader for the cluster. Current cluster head will promote the best candidate as the new cluster head and abdicate itself from its position. ACE results in uniform cluster formation with a packing efficiency close to hexagonal close-packing. However, ACE does not consider the residual energy of the nodes while selecting cluster heads. Hence, the clustering is far away from the optimal energy efficient.

HEED [14] introduces a distributed algorithm considering the residual energy of sensor nodes. It results in some clusters by uniformly distributing the cluster heads across the network. It periodically selects cluster heads according to a hybrid parameter which consists of a primary parameter, the residual energy of a node, and a secondary parameter, such as propinquity of a node to its neighbors or node degree. HEED converges in $O(1)$ iterations using low messaging overhead and achieves fairly uniform cluster head distribution across the network. However, it chooses the initial percentage of cluster heads randomly. This random choice remains as a severe limitation of this algorithm.

PADCP [15] uses several adaptive schemes like dynamic cluster range, dynamic transmission power and cluster head re-election to form clusters. In this approach, the sensor nodes are assumed to have the same transmission capability and the ability to adjust transmission power in five levels. PADCP has four major phases—neighbor information collection, cluster head election using a cost function, cluster formation using HEED, and cluster head re-election in case of residual energy lower than a pre defined threshold value. The mobility of the sensor nodes is considered in cluster formation. However, it suffers from the same randomly chosen initial probability limitations of HEED as it completely follows HEED algorithm for cluster formation in its third phase. Moreover, there is no suggestion about the optimal weights of the cost function used in cluster head selection and the threshold used in cluster head re-election.

LEACH [1] introduced a simple mechanism for localized coordination and control for cluster set-up and operation. It also introduces the randomized rotation of the cluster heads and the corresponding clusters. However, it does not consider the variation of the initial energy nor the residual energy of sensors during cluster head selection. SEP [16], a LEACH variant, modifies the equation of the threshold. However, it considers two types of nodes only, normal and advanced, instead of many types that can be encountered in the wireless sensor network after a significant amount of time of operation. Deterministic Cluster Head Selection [17], another variant of LEACH also modifies the threshold to accommodate the heterogeneity of residual energy based on some heuristics. LEACH-C, proposed by the same authors of LEACH in [18], is a centralized technique which selects the cluster heads based on their positions. It considers uniform distribution of the cluster heads based on their positions and the average residual energy in the network. They did not consider the relative residual energy in each sensor node. Adaptive Cluster Head Selection [19], a distributed clustering technique based on LEACH, considers the positions but not the relative residual energies of the sensor nodes.

There are a number of different research works that maximize network lifetime other than clustering. Lifetime is defined in a various ways in those works. In [20], functional lifetime is analyzed solving the linear progra-

ms only for simple and regular network topologies. Functional lifetime of a sensor network is defined as the maximum number of times a certain data collection task can be performed without the death of any sensor node. In [21], average network lifetime is maximized for a sensor network which is under physical node destruction by deriving deployment plan. In [22], α-lifetime of a wireless sensor network is maximized. α-lifetime is the time duration during which at least α portion of deployed sensor area is covered. In [23], a mathematical model is devised for sensor network, where data generation events are spatially and temporally independent. Based on the model, it also introduces a routing protocol for optimal average lifetime. In [24], a method is introduced using the k-shortest simple path algorithm and a dynamic programming method rooted in operational rate-distortion (RD) theory to increase the operational lifetime of a multi-hop 802.15.4 wireless sensor networks. In [25], sensor trees with desired properties are constructed from fusion center and then these sensor trees are scheduled to maximize network lifetime. It considers network lifetime as the time passed before the death of first node in the network. Load Balancing Protocol (LBP) [26] makes the number of live sensor nodes as large as possible by the enforcement of load balancing. Deterministic Energy-Efficient Protocol for Sensing (DEEPS) [27] allows higher energy consumption for sensors with higher total supply and minimizes energy consumption rate for low energy targets. Deterministic Energy-Efficient Protocol for Adjustable Range Sensing (ADEEPS) [28] is an extension of DEEPS. ADEEPS controls sensing range with the underlying approach of DEEPS. In [29], lifetime as time till the death of first node is improved by real time classifier using ART1 neural network model along with co-operative routing. In [30], network lifetime in terms of the death of first sensor node or the first failure of a transmission in the network is maximized by optimal sensor scheduling. It maps the problem to a stochastic shortest-path multi-armed bandit problem and thus chooses the sensor with the largest *Gittins* index for optimal transmission. In [31], Maximum Lifetime Data Aggregation (MLDA) problem is solved by selecting the best data aggregation tree using integer programming. It considers lifetime as the time during which information from all the sensors can be gathered to the base station. In [32], a combinative measurement is defined based on information utility, communication cost, and energy level. Weights of these factors are self-optimized using autonomic computing. In [33], average lifetime is maximized by reducing energy consumption through the enforcement of disjoint sets of sensor nodes. This approach maps Disjoint Set Cover problem to Maximum Flow Problem and then solves the Maximum Flow Problem by mixed integer programming. In [34], average lifetime is maximized by near optimal routing protocol which performs two shortest path computations to route a message. In [35], average lifetime is maximized by optimal routing through the formulation of linear programming problem. It considers both communication energy consumption rates and residual energy levels of two end nodes in the computation of link cost. In [36], lifetime of a fault tolerant sensor network in terms of death of first sensor node in the network is maximized by using multipath diversity and erasure codes. SPINDS [37] maximizes lifetime in terms of time till the failure of first Aggregation and Forwarding Node (AFN) in two steps. It formulates joint problem of energy provisioning and relay node placement into a mixed-integer nonlinear programming (MINLP) problem. Then it transforms MINLP problem into linear programming (LP) problem with maintaining all critical points in the search space. In [38], sensing ranges of sensor nodes are considered as adjustable. It finds maximum number of set covers and the sensing ranges of sensor nodes to achieve maximum lifetime in terms of time until BS detects the first failure. MLDR [39] attempts to improve network lifetime based on the death of first sensor node in the network by efficient routing using integer programming. This research work also uses data aggregation. In [40], network lifetime based on the death of first sensor node in the network is improved by distributed optimal routing technique using linear programming and sub-gradient algorithm.

A number of research works [20,25,29-31,36,39,40] attempts to improve network stability period by various techniques like—routing, scheduling, aggregation etc. However, in this paper we attempt to improve the network stability period using clustering as it can serve as a better platform for upper layer functionality such as broadcasting, aggregation etc. Our novel algorithm SSN exploits the underlying method of LEACH due to its wide acceptability. In experimental analysis, we compare SSN with LEACH and its best variant. For this reason, we describe LEACH and its variants in detail in the following section.

3. Leach

LEACH is a self-organizing and adaptive clustering protocol [1]. It dynamically creates clusters in order to distribute the energy load evenly among all of the sensor nodes. This algorithm needs time synchronization. Cluster heads are randomly rotated during each time interval. The resultant cluster heads directly communicate with the base station.

3.1. Mechanism

In LEACH, the lifetime of the network is divided into some discrete, disjoint time intervals. Each interval is again divided into some subintervals or rounds as shown in

Figure 1. Each subinterval begins with an advertisement phase followed by a cluster set up phase. In the advertisement phase, each node independently decides whether to become a cluster head or not. In the cluster set-up phase, the clusters are organized based on the decisions made in the advertisement phase. Then a steady-state phase follows. In this phase, the followers, *i.e.*, the sensor nodes except cluster heads, will send data to the corresponding cluster head. The cluster heads accumulate and compress the received data with its own data. Cluster heads send the compressed data to the base station. In order to minimize cluster establishment overhead, the duration of steady-state phase must be longer than that of cluster set-up phase.

At the very beginning of advertisement phase, each node decides whether it wants to become a cluster head for the current round. This decision is based on the suggested percentage of cluster heads for the network, which is set a priori. This decision also depends on the number of times the node has already been a cluster head. This decision is made by a node n choosing a random number between 0 and 1. If the number is less than a threshold $T(n)$, the node decides to become a cluster head. The threshold is calculated as follows:

$$T(n) = \begin{cases} \dfrac{P}{1 - P \times \left(r \bmod \dfrac{1}{P} \right)} & \text{if } n \in G \\ 0 & \text{otherwise} \end{cases}$$

where,

P = the percentage of nodes that can become cluster heads (e.g., P = 0.05);

$1/P$ = the number of subintervals in an interval;

r = the current subinterval;

G = the set of nodes that have not been cluster heads yet in the current interval.

Using this threshold, a node can be a cluster head in any one of $1/P$ subintervals in an interval. At the first subinterval of an interval (r = 0), each node has a probability P to become a cluster head. The nodes that are cluster heads in the first subinterval cannot be cluster heads in the next ($1/P - 1$) subintervals of the same interval. Thus the probability that the remaining nodes are becoming cluster heads is increasing. After the completion of $1/P$ subintervals, a new interval will start and all the nodes are again eligible to become cluster head.

Each node that has chosen itself as a cluster head in the current subinterval, broadcasts an advertisement message to the rest of the nodes. The non-cluster-head nodes will choose the cluster to which it will belong in this subinterval. This decision is based on the received signal strength of the advertised message. Assuming symmetric propagation channels, the cluster head whose advertisements have been heard with the largest signal strength will be selected by a non-cluster-head sensor node as its cluster head. In case of a tie, a cluster head is chosen randomly.

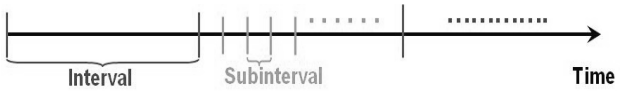


Figure 1. Discrete and disjoint intervals in the whole network lifetime; discrete and disjoint subintervals in an interval.

3.2. Mathematical Models

There are some incomplete mathematical models available on LEACH. In [18], a mathematical model is proposed to compute the total energy dissipation in the sensor network for the transmission of a frame. Taking the derivative of the total energy it finds the optimum number of clusters, k_{opt} as:

$$k_{opt} = \frac{\sqrt{N}}{\sqrt{2\pi}} \sqrt{\frac{\varepsilon_{fs}}{\varepsilon_{mp}}} \frac{M}{{d_{BS}}^2}$$

where, N is the total number of sensor nodes, M is the dimension of the sensor area, d_{BS} is the distance between cluster head and base station, ε_{fs} and ε_{mp} are the amplifier energies.

In [41], a mathematical model is proposed to compute the total energy consumption in the sensor network during a single round. Taking the derivative of the total energy it also finds the optimum number of clusters, k_{opt} as:

$$k_{opt} = \frac{\sqrt{N}}{\sqrt{\pi}} \sqrt{\frac{\varepsilon_{fs}}{\varepsilon_{mp}}} \frac{M}{{d_{BS}}^2}$$

In [42], a mathematical model is proposed to calculate the total energy consumption in the sensor network during a single round. It also finds the optimum desired cluster head probability, p_{opt} as:

$$p_{opt} = \frac{1}{2} \sqrt{\frac{\varepsilon_{fs}}{\lambda \left(\varepsilon_{mp} {d_{BS}}^4 - E_{elec} - E_{DA} \right)}}$$

where, λ is the intensity of homogeneous spatial Poisson process that indicates the sensor node density, E_{elec} is the electronic energy required for coding, modulation, filtering etc. and E_{DA} is the energy required for data aggregation.

However, the lifetime of a sensor node is directly the inverse of its long run rate or expected rate of energy consumption. Therefore, in order to elongate network lifetime, the long run rate of energy consumption must be given more importance than other metrices (for example, energy required to transmit one frame [41] or total energy consumption in an interval [42]). Moreover, none of these models consider the situation in which all the sensor nodes in the network can pick a random number higher than its respective threshold and become a temporary

follower. In this case, no sensor node will find any other node to choose as its cluster head under which it can keep its follower status. In this circumstance, every node changes its state from follower to cluster head, *i.e.*, all the sensor nodes will become a one-member cluster head. In [43,44], a complete mathematical model is proposed incorporating all these factors.

In [48], Heinzelman First Order Radio Model [21] is used as the energy model and Renewal Reward Process [26,48] is used as the underlying stochastic process to calculate long run rate of energy consumption. It defines the following parameters in the model:

1) P be the desired percentage of cluster heads,
2) s be the number of subintervals in an interval, therefore $s = 1/P$,
3) P_h be the probability of becoming cluster head of a follower node at the start of any subinterval,
4) P_h' be the probability of becoming cluster head of a cluster head node at the start of a subinterval in the next interval,
5) Φ_0 be the probability of becoming cluster head of a sensor node at the start of any subinterval,
6) $T(n)$ be the currently considered threshold value.
7) N be the total number of sensor nodes in the network.
8) $a \times b$ be the two dimensions of rectangular sensor area.

According to Renewal Reward Theorem, the rate of reward will be:

$$\lim_{t\to\infty} \frac{R(t)}{t} = \frac{E(R)}{E(X)} \tag{1}$$

where, R is reward and X is cycle length. It considers the energy consumed by the sensor as the reward and the difference between two consecutive subintervals in which a sensor node becomes cluster head as the cycle.

It considers different state transition diagrams for a sensor node between two states while changing the subinterval in an interval and between two states while changing the subinterval as well as the interval to compute $E(X)$. **Figure 2** shows those state transition diagrams.

Using these state transition diagrams, the probability of becoming a cluster head, Φ_0, at the start of any subinterval is calculated as follows:

$$\Phi_0 = \left(\frac{P_h}{s}\right) + P_h = (s+1) \times \left(\frac{P_h}{s}\right) \tag{2}$$

After a number of steps, the long run rate of energy consumption is calculated as:

$$\lim_{t\to\infty}\frac{R(t)}{t} = \frac{E(R)}{E(X)}$$

$$= (1-\Phi_0) \times \left[(E_{elec} k) + (\in_{amp\_F} k) \frac{1}{2} \left(\frac{ab}{\pi}\right)^2 \right.$$

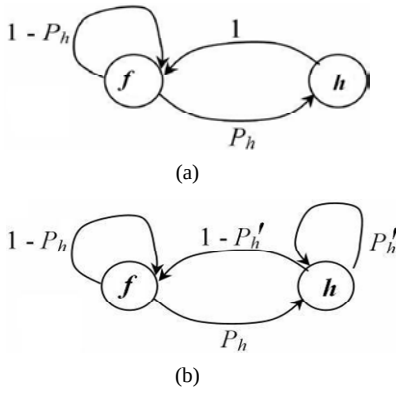


Figure 2. State transition of a node while (a) Changing subinterval without changing Interval, (b) Changing subinterval as well as the interval.

$$\left. \sum_{n=1}^{N} g(n)\left(\frac{1}{n} - \frac{1}{n+1}\right)\right] + \Phi_0 \sum_{i=0}^{N-1}\left[p(i) \times \sum_{n=1}^{N-i} q(i,n)\right] \tag{3}$$

where,

$$g(n) = \binom{N}{n} \Phi_0{}^n (1-\Phi_0)^{N-n} \frac{2n\pi}{ab},$$

$$p(i) = (2i+1)k(E_{elec} + E_{DA}) + (i+1) \in_{amp\_H} k \gamma d_{BS}{}^{\lambda},$$

$$q(i,n) = \binom{N-n}{i} h(r,n)^i \left(1 - h(r,n)\right)^{N-n-i}$$

$$\binom{N}{n} \Phi_0{}^n (1-\Phi_0)^{N-n},$$

and

$$h(r,n) = \frac{\pi r^2}{ab} p_a \left(1 - \frac{\pi r^2}{ab} p_a\right)^{n-1}.$$

Here, E_{elec} is energy required per bit to run the circuitry in transmitter or receiver, E_{DA} is the energy required for data aggregation, $\in_{amp\_F}$ is the energy constant for the radio transmission of a follower node, $\in_{amp\_H}$ is the energy constant for the radio transmission of a cluster head node, k is number of bits in a message, λ is the path loss exponent, d_{BS} is the distance between cluster head and base station, and p_a is the percentage of the circular area (centered at a follower and with radius equals distance to a cluster head) falls within the sensor area. As we cannot get any closed form for derivative of Equation 3, we can get the optimal percentage of cluster heads by plotting the value of long run rate of energy consumption from the equation.

3.3. Limitations

This algorithm introduced a fairly simple strategy whi-

ch is more efficient than the direct transmission and the minimum-transmission-energy (MTE) protocol that chooses the route to minimize the transmitter energy. However, it has some limitations:

1) LEACH always wants to achieve an even distribution of energy consumption which might not be rational. Residual energies in different nodes do not remain same after a significant amount of time of operation. Nodes with higher residual energy should get preference to be elected as cluster head. Otherwise, longer network stability as well as longer network lifetime cannot be ensured.

2) When the number of live nodes becomes small, the number of prospective cluster heads which is equal to the number of live nodes multiplied by desired percentage of heads will also become very small and in some cases it may become less than one. For example, if the initial number of sensor nodes is 100 and the desired percentage of heads P is 0.05 then the initial number of prospective heads is 100 × 0.05 = 5. However, with the same P when the number of live nodes becomes less than 20 the number of prospective heads will become less than one. Under this condition in most of the subintervals, none of the live sensor nodes can become a cluster head by choosing a random number which is less than the current threshold. In other words, there will be no cluster head available to the sensor nodes to which they can become followers. Rather, all the live sensors will force themselves to become a one member cluster head. In this particular case, the resultant cluster setting will behave like a setting which does not have any clustering. For this reason, no energy efficiency will be gained.

A number of variants have already been proposed for LEACH to overcome its limitations. Some of them are briefly summarized in the following section.

3.4. LEACH Variants

SEP [16] is variant of LEACH, which elects the clust- er heads based on weighted probabilities according to the residual energy of the sensor nodes. It assumes that a percentage of the sensor nodes are coming with higher energy resources and studies the impact of heterogeneity of nodes based on their energy levels. It follows the underlying synchronization approach used in LEACH. In addition, it considers the variation in the residual energy assuming two types of nodes—normal and advanced.

It assumes m fractions of the nodes are advanced nodes, which have α times energy than that of the normal nodes. As a result, it assumes $n(1 + \alpha\, m)$ number of virtual normal nodes in the network. It extends the number of subintervals from $1/P$ to $(1 + \alpha\, m)/P$ in an interval. The objective of this extension is to elect a normal node once and an advanced node $(1 + \alpha)$ times as the cluster head in an interval. The probability equation to become cluster head has been modified. In fact, two different equations are used for the normal and the advanced nodes. The weighted election probabilities for the normal and the advanced nodes are p_{nrm} and p_{adv} respectively. Their equations are as follows:

$$p_{nrm} = \frac{p_{opt}}{1+\alpha \times m}$$

and,

$$p_{adv} = \frac{p_{opt}}{1+\alpha \times m} \times (1+\alpha)$$

where, p_{opt} is the optimal probability of a node to become a cluster head. It also uses two different equations for the threshold. One for the normal nodes called $T(s_{nrm})$ and the other for the advanced nodes called $T(s_{adv})$. $T(s_{nrm})$ and $T(s_{adv})$ are calculated as follows:

$$T(s_{nrm}) = \begin{cases} \dfrac{p_{nrm}}{1 - p_{nrm} \times \left(r \mod \dfrac{1}{p_{nrm}} \right)} & \text{if } s_{nrm} \in G' \\ 0 & \text{otherwise} \end{cases}$$

and,

$$T(s_{adv}) = \begin{cases} \dfrac{p_{adv}}{1 - p_{adv} \times \left(r \mod \dfrac{1}{p_{adv}} \right)} & \text{if } s_{adv} \in G'' \\ 0 & \text{otherwise} \end{cases}$$

where, G' is the set of normal nodes that have not become cluster head yet within the last $1/p_{nrm}$ subintervals and G'' is the set of advanced nodes that have not become cluster head yet within the last $1/p_{adv}$ subintervals in an interval.

This works introduced the heterogeneity to LEACH in terms of two levels of residual energy. However, it has some limitations:

- In SEP, the percentage of cluster heads is optimized based on the energy consumption in an interval. However, this value should be optimized on the basis of the long run rate or expected rate of energy consumption for achieving the higher network stability period.
- SEP considers two types of nodes only in terms of residual energy. However, during the life cycle of the network the different levels of the residual energies may exist which will not be covered by only two types. More types of nodes are necessary to consider covering numerous residual energy levels in different nodes to achieve maximum network stability.
- It did not attempt any improvement to enhance the network stability.

Deterministic Cluster Head Selection [17] introduces the heterogeneity to LEACH in terms of residual energy. It considers the residual energies of the sensor nodes in order to manage rational power consumption throughout the network. It follows the underlying mechanism of LEACH exactly. It has changed the equation of the thresh-

old value only to incorporate the residual energy in cluster head selection process as follows:

$$T(n)_{new} = \frac{P}{1 - P \times \left(r \mod \frac{1}{P} \right)} \times \frac{E_{n\_current}}{E_{n\_\max}}$$

where, $E_{n\_current}$ is the current energy, E_{n_max} the initial energy of the node. The other parameters have the same definitions as of LEACH.

After a significant amount of time of operation, the residual energies of the sensors would become very low and then this threshold value will be very low. This can result in a situation where all the live sensors are one member cluster head. In this case the energy consumption rate will be very high. To break this stuck condition another modified equation of the threshold value has been proposed:

$$T(n)_{new} = \frac{P}{1 - P \times \left(r \mod \frac{1}{\mathrm{P}} \right)} \times \left[\frac{E_{n\_current}}{E_{n\_\max}} + \left(r_s div \frac{1}{P} \right) \times \left(1 - \frac{E_{n\_current}}{E_{n\_\max}} \right) \right]$$

where, r_s is the number of consecutive rounds in which a node has not been cluster head.

This works introduced the heterogeneity to LEACH in terms of different levels residual energy. However, it has some limitations:

- Deterministic Cluster Head Selection uses a random value for the percentage of heads parameter like LEACH. Therefore, it does not consider the optimal value of this parameter.
- It does not suggest any optimum value for r_s.
- It did not attempt any improvement to enhance the network stability.

LEACH-C [18] is a centralized technique to cluster sensor nodes based on their positions. In this approach, base station selects cluster heads to get uniformly distributed clusters. In LEACH-C, sensor nodes detect their current locations using GPS (Global Positioning System) receiver or any other technique. At the beginning of each interval, each node informs the base station its current location and residual energy level. After receiving the information from all the sensor nodes, base station computes the average residual energy in the network. It precludes those sensor nodes whose residual energy is below the average residual energy from attaining cluster headship. Base station selects the cluster heads from the remaining nodes using the simulated annealing algorithm [47]. Base station also selects corresponding followers for the clusters while selecting the clusters and cluster heads, and the base station broadcasts a message into the network informing these selections.

This algorithm minimizes the total sum of squared distances between all the non-cluster-head nodes and the corresponding closest cluster head node. Thus, it minimizes the amount of energy necessary to use to transmit data to the cluster head nodes by the non-cluster-head nodes. However, it suffers from the following limitations:

- In LEACH-C, the base station selects the cluster heads based on their positions and the average residual energy in the network. Like LEACH, the individual residual energy in each sensor node has little impact on the cluster head selection process in LEACH-C. This centralized algorithm also suffers from non-scalability.
- Incorporating GPS receiver or similar device in the sensor nodes increases sensor node cost.
- It did not attempt any improvement to enhance the network stability.

Adaptive Cluster Head Selection 19 assumes that a sensor node knows its distance from another sensor node by observing the signal strengths in the received messages. At first, this approach randomly selects cluster heads following LEACH. Next it reselects the cluster heads considering the distance between each cluster head and the sensor nodes farthest from the cluster heads. The reselection is done in order to distribute the cluster heads uniformly in the network. When a sensor node is selected as a cluster head by LEACH, it broadcasts an advertisement message to all other nodes. Other sensor nodes respond to the broadcast. From the received responses, a cluster head calculates its distance to its farthest follower node and its distance to the nearest cluster head of neighbor clusters. It subtracts the first distance from the later. Three cases may arise as follows:

Case 1: The result is positive.

Case 2: The result is negative.

Case 3: The result is zero.

In order to place the cluster head in an optimum location, the cluster head is moved to the direction of the closest head in Case 1 and to the direction of the farthest sensor node in Case 2. Cluster head position remains the same in Case 3.

This approach ensures uniform distribution of the cluster heads. However, it has the following limitations:

- It completely ignores the relative residual energy of each sensor node in the network while selecting the cluster heads. It also suffers from other LEACH limitations.
- In this work cluster head movement, if necessary, is not clearly defined.
- It did not attempt any improvement to enhance the network stability.

Therefore, none of the research works mentioned in this section makes any attempt to improve network stability. In the next section, we propose a novel technique, Stable Sensor Network (SSN) to improve this metric.

4. Stable Sensor Network (SSN)

In this section, we propose a new algorithm to cluster sensor nodes in a network to improve network stability in terms of death of the first sensor node. We follow the underlying approach of LEACH. In LEACH, each sensor node is given equal chance to get cluster headship and thus its lifetime depends only on its own residual energy. Therefore, a sensor node with low residual energy dies within a short period. However, there may be some other sensor nodes alive after its death. If that sensor node could exploit the residual energies of the live sensor nodes then it would live longer. Therefore, we should use the total residual energy of the network to increase the lifetime of the sensor node which dies first. We present three heuristics to achieve this goal. We illustrate our complete clustering algorithm SSN in details after describing those heuristics. We also adapt the mathematical model derived in [44] for SSN incorporating the heuristics accordingly.

4.1. Heuristics

We propose three heuristics for SSN. First two heuristics basically attempt to use the residual energy of the network in a sensor node. A sensor node needs residual energy status of other sensor nodes in the network for the second heuristic. It can get that information from unacknowledged broadcasts from other sensor nodes. Some of the information will not be available to the sensor node due to unacknowledged broadcasts. Third heuristic attempts to make up this missing information.

Heuristic 1: Energy consumption of a cluster head node is higher than that of a follower node. Therefore, sensor nodes with higher residual energy should be elected as cluster heads. In the original LEACH algorithm, if a node becomes cluster head in a subinterval it cannot become cluster head again in any of the subsequent subintervals of the same interval. However, if a sensor node with higher residual energy can become cluster head again in other subintervals in the same interval then a sensor node with lower energy can escape from being cluster head. In that case, the lifetime of this lower energy sensor node will increase by using residual energy of a higher energy sensor node. For this reason, we make the subintervals completely memory less and eliminate the use of the separate set of nodes that have not been cluster head yet in the current interval. With this modification, the probability of becoming cluster head of a sensor node in a subinterval does not depend on its status in the previous subintervals. This heuristic partially increase network stability period.

Heuristic 2: We can expect higher stability period of a sensor network if we increase the probability to become cluster heads for sensor nodes with higher residual energies. We should consider relative residual energy of a sensor node to determine whether it is with higher residual energy or not. For this reason, we judge the relative residual energy of a sensor node while selecting it as a cluster head. We map the relative residual energy of a sensor node in its threshold computation so that it keeps its expected value at the optimal percentage of cluster heads P. At the beginning of each subinterval, each node knows its own residual energy (E_{cur}) along with maximum (E_{cur_max}), minimum (E_{cur_min}), and average (E_{cur_avg}) residual energies of all sensor nodes alive in the network. Considering average residual energy (E_{cur_avg}) corresponds to expected percentage of cluster heads (P), we map E_{cur_min}, and E_{cur_max} to $(1 - P_{range})$ and $(1 + P_{range})$ respectively, where P_{range} is the minimum between P and $(1 - P)$. If $P \leq (1 - P)$, $(P - P_{range})$ becomes zero and if $P \geq (1 - P)$, $(P + P_{range})$ becomes one. This has been shown in **Figure 3**.

We define deviation from P for a sensor node based on the difference between its residual energy E_{cur} and the average residual energy E_{curr_avg} in the network. Hence, the deviation is:

$$\Delta P = P_{range} \times E_r \tag{4}$$

where,

$$E_r = \begin{cases} \dfrac{E_{cur} - E_{cur\_avg}}{E_{cur\_avg} - E_{cur\_\min}}, & if \;\; E_{cur} < E_{cur\_avg} \\ 0, & if \;\; E_{cur} = E_{cur\_avg} \\ \dfrac{E_{cur} - E_{cur\_avg}}{E_{cur\_\max} - E_{cur\_avg}}, & if \;\; E_{cur} > E_{cur\_avg} \end{cases}$$

In order to make the threshold value proportional to the residual energy of a sensor node, we assign threshold value equal to P plus ΔP, *i.e.*

$$T(n) = P + \Delta P \tag{5}$$

This heuristic along with the previous one enable a sensor node to become cluster head according to its relative residual energy in the network. A sensor node with higher residual energy is ensured to be more probable and a sensor node with lower residual energy is ensured to be less probable in the selection of cluster heads.

Heuristic 3: To apply the previous heuristic, a sensor node must know the maximum, minimum, and average residual energies of all sensor nodes alive in the network. To calculate these values it must know residual energies

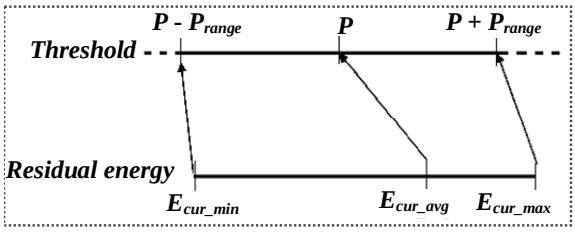


Figure 3. Distribution of Threshold Value according to Residual Energy.

of all other sensor nodes. Therefore, each node must broadcast its residual energy level. For guaranteed availability of this information, some acknowledgement based data transmission technique should be followed. However, this will incur a significant energy cost. For this reason, we make a trade-off between the accuracy of this information and the energy required to transmit them. We simply adopt one pass broadcast and to overcome the accuracy problem we multiply the threshold by the ratio between total number of deployed sensor nodes (N) and number of sensor nodes (N_{live}) from whom residual energy information is received. This will increase the probability of a sensor node to become cluster head when it finds a lower number of live sensor nodes in the network. This ultimately ensures the preservation of overall optimal percentage of cluster heads among those reachable sensor nodes irrespective of the statuses of the unreachable sensor nodes. This heuristic changes the equation of the threshold as the following way:

$$T(n) = \left(P + \Delta P\right) \times \frac{N}{N_{live}} \quad (6)$$

4.2. SSN Algorithm

We divide the lifetime of the network into some discrete and disjoint equal length intervals in SSN. Each interval has three consecutive phases—advertisement, cluster-setup, and steady-state phase. The algorithm depicted in **Figure 4** runs independently in each sensor node in each interval. The parameters are initialized at the start of the algorithm. E_{cur} is set to its current residual energy level. E_{cur_max}, E_{cur_min}, and E_{cur_avg} are set to its own current residual energy level, *i.e.*, equal to E_{cur}. The number of live sensor node, N_{live}, is set to one assuming it is the only live sensor node in the network. Advertisement, cluster-setup, and steady-state phases are executed as follows:

1) Advertisement Phase: During this phase, each node executes two parallel processes. In one process, each node waits for a uniformly distributed random amount of time and then broadcasts its current residual energy level. This random delay reduces the probability of collision. Another process receives the current residual energy levels of other sensor nodes. A sensor node may receive multiple copies of a current energy level advertisement me- ssage from the same sensor node due to multi-path effect. A receiver sensor node detects these duplicate receptions and ignores them. A receiver sensor node updates the parameters—E_{cur_max}, E_{cur_min}, E_{cur_avg}, and N_{live} using the fresh advertisement messages only.

2) Cluster Set-up Phase: In this phase, each sensor node independently decides whether to become a cluster head or not based on the information gathered in the advertisement phase. At first, it calculates the threshold $T(n)$ using Equation 6. Next, it picks a random number and compares the random number with the threshold. Three cases may arise as follows:

CASE 1: The random number is less than the threshold. In this case, the sensor node becomes a cluster head and broadcasts HEAD_EXPOSURE message.

CASE 2: The random number is not less than the threshold and it does not receive any HEAD_EXPOSURE message from other sensor nodes. In this case, the sensor node becomes a one member cluster head.

CASE 3: The random number is not less than the threshold and it receives one or more HEAD_EXPOSURE messages from other sensor nodes. In this case, the sensor node becomes a follower of the nearest cluster head and sends a FOLLOWER_ACCEPTANCE: Message to the nearest cluster head.

3) *Steady-state Phase*: In this phase, the followers send data to the corresponding cluster head. The cluster heads accumulate, aggregate, and compress the received data with its own data. Cluster heads send the aggregated and compressed data to the base station. The duration of steady-state phase is significantly longer than the summation of the durations of the advertisement and cluster-setup phases in order to minimize cluster establishment overhead.

4.3. Mathematical Model of SSN

The difference between underlying mode of operations of LEACH and SSN arises because of three new heuristics. The last two heuristics make change only in the threshold value ($T(n)$). This change merely affects the probability of becoming cluster head of a follower node at the start of any subinterval (P_h). Otherwise, there is no impact of these two heuristics on Equation 3, which is the latest mathematical formulation of LEACH. However, *heuristic* 1 of our new clustering algorithm makes the subinterval completely memory less. For this heuristic, the first state transition diagram of **Figure 2** is no longer applicable. However, Φ_0 is formulated form the weighted combination of two state transition diagrams of **Figure 2** in the mathematical model of LEACH. Therefore, the formulation of Φ_0 needs to be changed in the mathematical model of SSN. With the introduction of *heuristic* 1, any sensor node can become cluster head irrespective of its status in the previous sub interval. Therefore, the probability of becoming cluster head of a follower node at the start of any subinterval (P_h) will no longer differ from the probability of becoming cluster head of a sensor node at the start of any subinterval (Φ_0). As a result, formulation of Φ_0 in the changed mathematical model of SSN will be $\Phi_0 = P_h$. With this change, we can use Equation 3 as the mathematical model of SSN.

Algorithm SSN ()

Set the value of E_{cur}
Initialize E_{cur_max}, E_{cur_min}, and E_{cur_avg} to E_{cur},
Initialize N_{live} to 1
Advertisement(); broadcasts and receives current energy levels
Cluster_Set_Up(); generates the clusters
Steady_State(); receive and transmit data

Algorithm Advertisement()

Transmit_Current_Energy_Residual_Level()
Receive_Current_ Residual_Energy_Level()

Algorithm Transmit_Current_ Residual_Energy_Level()

Wait for a random time
Broadcast own current residual energy level

Algorithm Receive_Current_ Residual_Energy_ Level()

For each received current residual energy level, E'_{cur}
if E'_{cur} is not a repetition from an already received node
Update_Parameters(N_{live}, E'_{cur})
endif

Algorithm Update_Parameters(N_{live}, E'_{cur})

if $E'_{cur} > E_{cur\_max}$
$E_{cur\_max} = E'_{cur}$
endif
if $E'_{cur} < E_{cur\_min}$
$E_{cur\_min} = E'_{cur}$
endif

$$E_{cur\_avg} = \frac{N_{live} * E_{cur\_avg} + E'_{cur}}{N_{live} + 1}$$

Increment N_{live}

Algorithm Cluster_Set_Up ()

$$P_{range} = \max(P, 1-P)$$

if $E_{cur} < E_{cur\_avg}$

$$E_r = \frac{E_{cur} - E_{cur\_avg}}{E_{cur\_avg} - E_{cur\_\min}}$$

else if $E_{cur} > E_{cur\_avg}$

$$E_r = \frac{E_{cur} - E_{cur\_avg}}{E_{cur\_\max} - E_{cur\_avg}}$$

else
$E_r = 0$
endif

$$\Delta P = P_{range} \times E_r$$

$$T(n) = (P + \Delta P) \times \frac{N}{N_{live}}$$

Choose a random number r
if ($r < T(n)$) then
status=head
broadcast HEAD_EXPOSURE message
else
Receive HEAD_EXPOSURE messages from other sensor nodes
if (no HEAD_EXPOSURE message received) then
status=head
else
status=follower
send FOLLOWER_ACCEPTANCE message to nearest cluster head
endif
endif

Algorithm Steady_State ()

if *status=follower*
send self originated data to own cluster head
else
receive messages from own followers
aggregate and compress them with own message
send to base station
endif

Figure 4. Algorithms for Stable Sensor Network (SSN).

We compare these mathematical models from simulation results in the next section. We also analyze the efficiency of SSN in that section.

5. Simulation Results

We conduct our simulation runs on a randomly deployed wireless sensor network. Our simulation program is written in visual C++. In this section, we first describe our network settings along with various parameters used in energy rate calculation. Then, we compare mathematical models of SSN with that of LEACH. Finally, we evaluate network stability in SSN with that of LEACH and its high performance variant.

5.1. Network Settings

We use network settings as shown in **Figure 5** in our simulation runs. The network settings do not make any impractical assumption to simplify the analysis. The settings are as follows:

- The dimension of sensor area is 200 × 200 m^2.
- Total number of sensor nodes in the network is 100.
- The sensor nodes are randomly distributed over the sensor area.
- Each sensor node is initially equipped with a battery of 5 Joule.
- The base station is located at position (400 meter, 100 meter).

We use the following parameters in the simulation runs for verification of mathematical models and in the first runs of the subsequent analyses:

1) The amount of energy per bit to run sensor node circuitry, E_{elec} is 5×10^{-8};

2) The value of energy constant, ϵ_{amp}, for radio transmission, is 1×10^{-10};

3) The number of data packets generated during each subinterval by a sensor node is normally distributed in the range of [0, 50], with the value of mean equal to 25. We applied Box-Muller transformation [48] to achieve this normal distribution from the uniform distribution of the built-in *rand*() function in visual C++.

4) Each data unit contains 8 bits data;

5) The probability that a message successfully arrives at its destination is 90%.

5.2 Verification of Mathematical Model

We first plot the long run rate of energy consumption from Equation 3 versus the percentage of heads for a LEACH node in **Figure 6**. The value of the probability of becoming cluster head of a sensor node at the start of any subinterval Φ_0 must not exceed 1 and the value of Φ_0 can be computed from Equation 2. According to Equation 2, if the value of P exceeds 0.61 then the value of Φ_0 will exceed 1. In order to avoid this, we plot the graph against the percentage of cluster heads P up to 0.61. According to the graph:

1) Energy consumption rate initially decreases very sharply with the increase of the percentage of cluster heads.

2) There is an optimal point for which energy consumption rate is the lowest. After this point the energy consumption rate increases with the increase of the percentage of cluster heads. In our simulation runs this optimal point is (0.045, 0.000337).

We also plot the long run rate of energy consumption from Equation 3 versus the percentage of heads for a SSN node in **Figure 7**. Here, the probability of becoming cluster head of a sensor node at the start of any subinterval Φ_0 is no longer computed from Equation 2 as in SSN Φ_0 directly maps to P_h. Therefore, we plot the graph against the percentage of cluster heads P up to 1.

The graph in **Figure 7** exhibits almost the same trends found in the graph in **Figure 6**. Energy consumption rate initially decreases very sharply with the increase of the percentage of cluster heads and after an optimal point the energy consumption rate increases with the increase of the percentage of cluster heads. In **Figure 7**, the optimal point for SSN is (0.045, 0.000331) which gives lower long run rate of energy consumption than the optimal point for LEACH found in **Figure 6**. This improvement is only due to *heuristic* 1 as only this heuristic modifies the mathematical model.

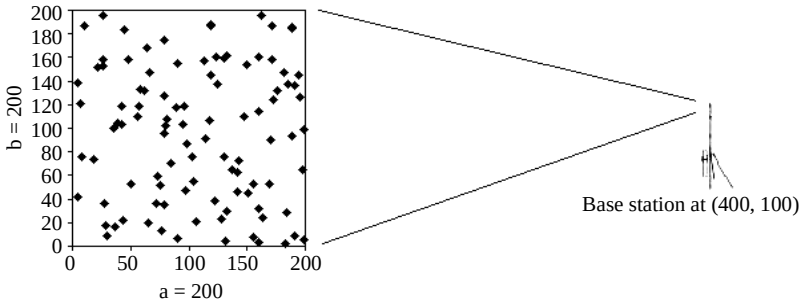


Figure 5. Network Settings: Uniformly distributed sensor nodes with a base station.

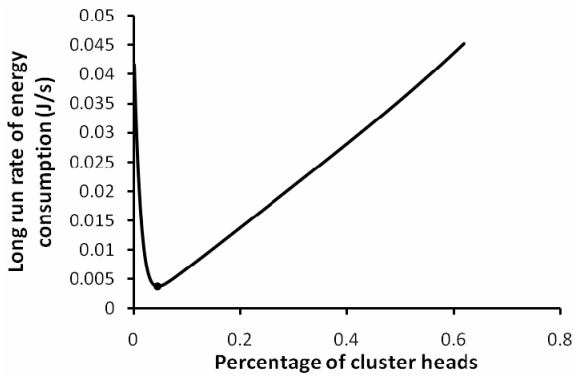


Figure 6. Long run rate of energy consumption against Different Percentage of Heads according to the Mathematical Model of LEACH.

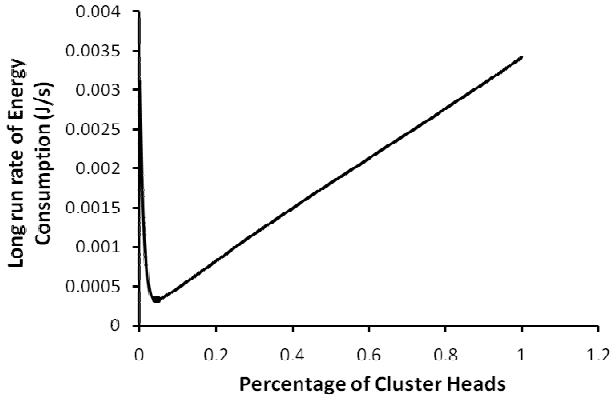


Figure 7. Long run rate of energy consumption against Different Percentage of Heads according to the Mathematical Model of SSN.

We evaluate network stability of SSN against that of LAECH and its best variant. The authors of Deterministic Cluster Head Selection [17] claimed that it improves the network stability period by 30% over LEACH whereas the authors of SEP [16] claimed that it does the improvement over LEACH by 26%. These two are the most improved LEACH variants claimed so far. For this reason, we take Deterministic Cluster Head Selection [17] as the best LEACH variant instead of SEP in our performance comparison. We already find $P = 0.045$ as the optimal percentage of cluster heads from the mathematical models of both LEACH and SSN. We use this value for all of the three techniques under evaluation. Our evaluation is based on three metrics:

1) Data rate of a sensor node,
2) Position of base station, and
3) Initial energy of a sensor node.

In the evaluation process, for each simulation run we take average of values found from fifty simulation passes. Each simulation run generates a point in a graph. We start with already described network settings for the first point in each of the graphs.

I. Data rate of a sensor node

In the initial network settings, number of data packets generated by a sensor node in a subinterval is normally distributed in the range of 0 to 50, with mean 25. We conduct fifteen simulation runs varying this range. We change the upper limit of the range from 50 packets with step of 10 packets in each simulation run. We plot the values of network stability periods in terms of FND in **Figure 8** and the values of HND in **Figure 9**. **Figure 9** indicates that HNDs of SSN, LEACH and LEACH variant are comparable. However, there is a significant steady improvement in FND for SSN over LEACH and its variant. We plot these improvements in **Figure 10**. The average improvement over LEACH and its variant is 53.42% and 35.62% accordingly.

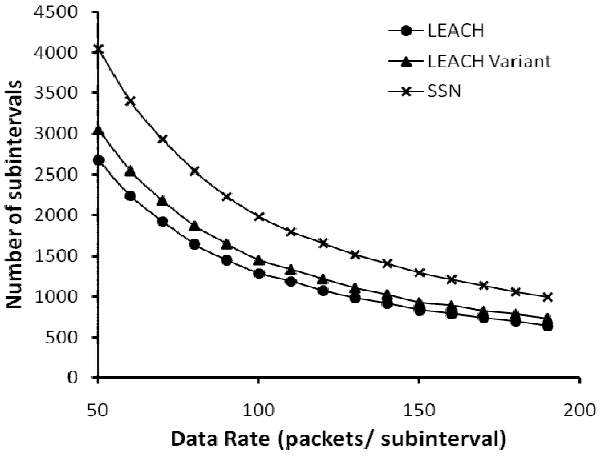


Figure 8. Network stability period in terms of First Node Dies (FND) for different data rates.

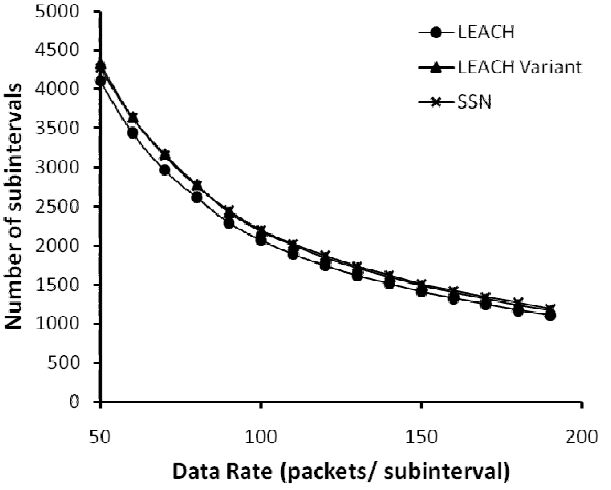


Figure 9. Half of the Nodes Die (HND) for different data rates.

II. Position of base station

In the initial network settings, base station is located at (400 m, 100 m). Therefore, distance of the base station from the center of network area is 300 meter. We conduct fifteen simulation runs varying this distance. We change the position of base station in first dimension from 400 meters with step of 25 meters in each simulation run. We plot the values of network stability periods in terms of FND in **Figure 11** and the values of HND in **Figure 12**. **Figure 12** indicates that HNDs of SSN, LEACH and LEACH variant are comparable. However, there is a significant steady improvement in FND for SSN over LEACH and its variant. We plot these improvements in **Figure 13**. The average improvement over LEACH and its variant is 48.55% and 30.22% accordingly.

III. Initial energy of a sensor node

In the initial network settings, initial energy of a sensor node is 5 Joule. We conduct fifteen simulation runs varying this initial energy. We change the initial energy of a sensor node from 5 Joule with step of 1 Joule in each simulation run. We plot the values of network stability periods in terms of FND in **Figure 14** and the values of

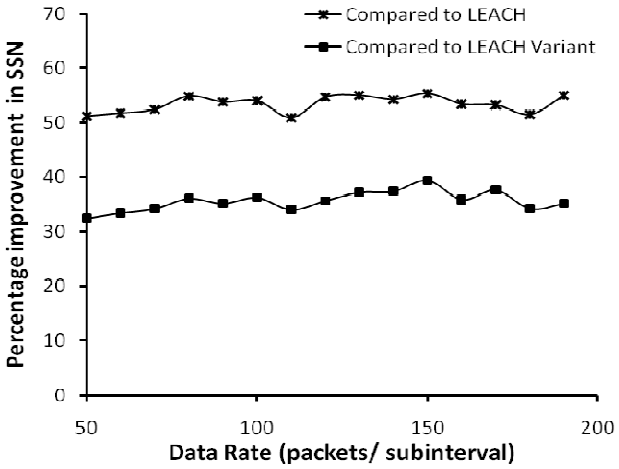


Figure 10. Improvement in network stability period in terms of First Node Dies (FND) in SSN for different data rates.

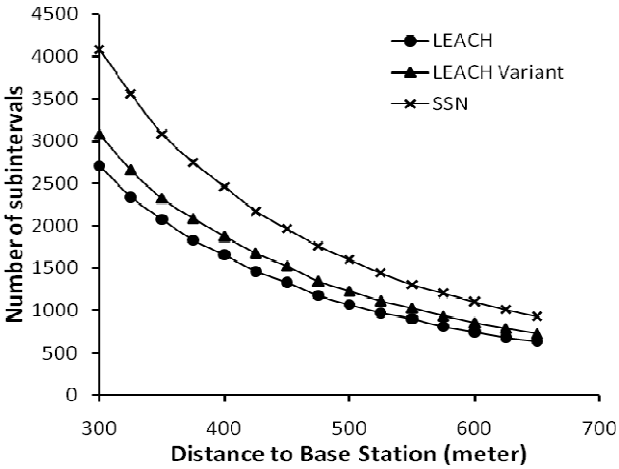


Figure 11. Network stability period in terms of First Node Dies (FND) for different positions of base station.

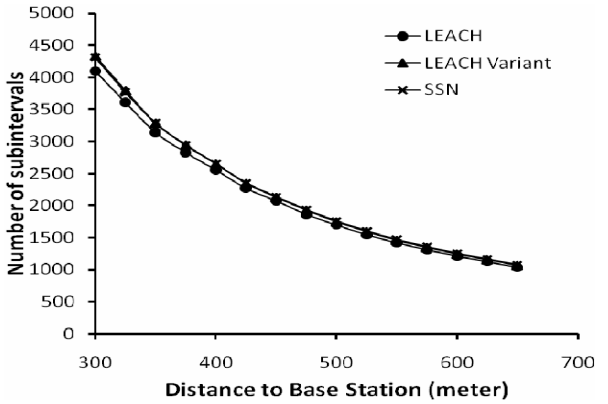


Figure 12. Half of the Nodes Die (HND) for different positions of base station.

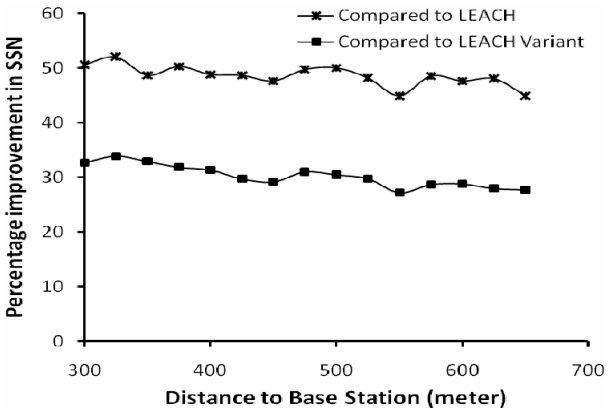


Figure 13. Improvement in network stability period in terms of First Node Dies (FND) in SSN for different positions of base station.

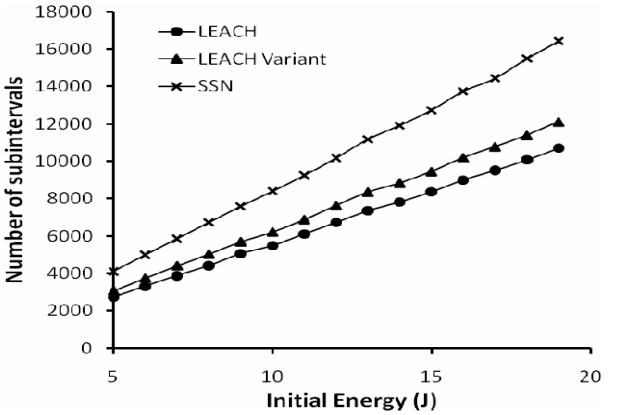


Figure 14. Network stability period in terms of First Node Dies (FND) for different initial energies of a sensor node.

HND in **Figure 15**.

Figrue 15 indicates that HNDs of SSN, LEACH and LEACH variant are comparable. However, there is a significant steady improvement in FND for SSN over LEACH and its variant. We plot these improvements in **Figrue 16**. The average improvement over LEACH and its variant is 52.04% and 34.25% accordingly.

These values clearly indicate that, SSN provides significantly higher time before first node dies in compareson to LEACH and its variant irrespective of data rate of sensor node or position of base station or initial energy of a sensor node.

6. Conclusions

We propose a novel self-organizing and adaptive clustering protocol SSN in this paper. We use three heuristics for SSN with proper justifications. We present its complete mathematical formulation with the help of that of LEACH. We evaluate its stability period with that of LEACH and its best variant. We get a significant steady improvement in the evaluation in different circumstances.

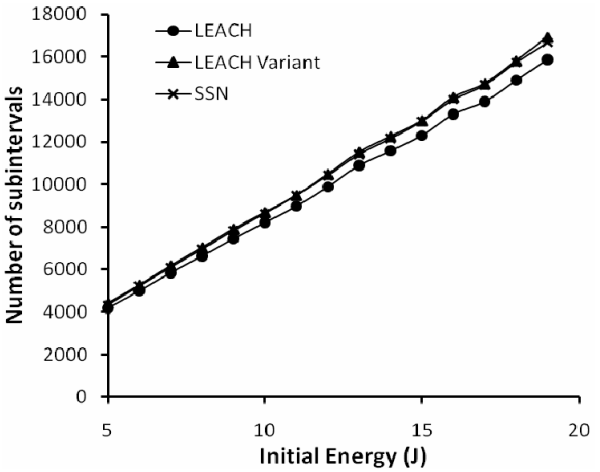


Figure 15. Half of the Nodes Die (HND) for different initial energies of a sensor node.

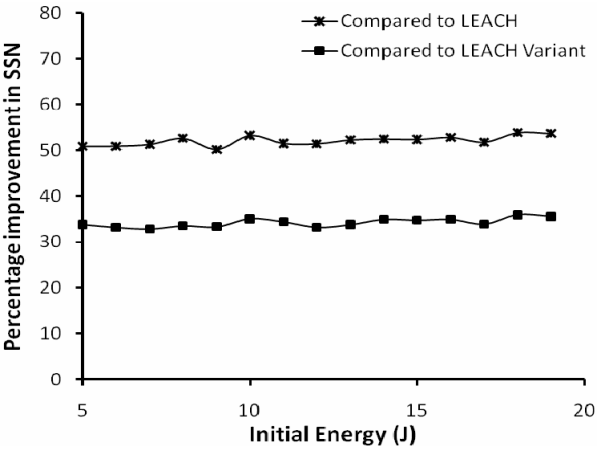


Figure 16. Improvement in network stability period in terms of First Node Dies (FND) in SSN for different initial energies of a sensor node.

7. Future Works

We propose SSN for homogeneous sensor nodes, *i.e.*, sensor nodes with similar transmission and sensing ranges. In our future works, we will attempt to enhance SSN for sensor nodes with different transmission and sensing ranges. We will also attempt to enhance SSN for multi radio sensor nodes, which are now emerging in the recent research works.

8. References

[1] W. B. Heinzelman, A. P. Chandrakasan and H. Balakrishnan, "Energy Efficient Communication Protocol for Wireless Microsensor Networks," *Proceedings of the Hawaii International Conference on System Sciences*, Maui, Vol. 2, January 2000, pp. 1-10.

[2] J. Han and K. Micheline, "Data Mining: Concepts and Techniques," 2nd Edition, Morgan Kauffman, Elsevier, San Francisco, 2001.

[3] S. Basagni, "Distributed Clustering Algorithm for Ad-Hoc Networks," *Proceedings of International Symposium on Parallel Architectures, Algorithms, and Networks* (*I-SPAN*), Perth, 1999, pp. 310-315.

[4] S. Banerjee and S. Khuller, "A Clustering Scheme for Hierarchical Control in Multi-Hop Wireless Networks," *Proceedings of IEEE INFOCOM*, Anchorage, 2001, pp. 1028-1037.

[5] M. Gerla, T. J. Kwon and G. Pei, "On Demand Routing in Large Ad Hoc Wireless Networks with Passive Clustering," *Proceedings of IEEE WCNC*, Chicago, Vol. 1, 2000, pp. 100-105.

[6] C. R. Lin and M. Gerla, "Adaptive Clustering for Mobile Wireless Networks," *IEEE Journal on Selected Areas in Communications*, Vol. 15, No. 7, 1997, pp. 1265-1275.

[7] J. Kamimura, N. Wakamiya and M. Murata, "Energy-Efficient Clustering Method for Data Gathering in Sensor Networks," *Proceedings of Workshop on Broadband Advanced Sensor Networks*, San Diego, Vol. 103, 2004, pp. 31-36.

[8] J. Leu, M. H. Tsai, T. C. Chiang and H. Y. M. Huang, "Adaptive Power Aware Clustering and Multicasting Protocol for Mobile Ad Hoc Networks," *Lecture Notes in Computer Science*, Vol. 4159, September 2006, pp. 331-340.

[9] S. Lindsey and C. S. Raghavendra, "PEGASIS: Power-Efficient Gathering in Sensor Information Systems," *IEEE Aerospace Conference Proceedings*, Big Sky, Vol. 3, 2002, pp. 1125-1130.

[10] L. Li, S. Dong and X. Wen, "An Energy Efficient Clus-

tering Routing Algorithm for Wireless Sensor Networks," *Journal of China Universities of Posts and Telecommunications*, Vol. 3, No. 13, September 2006, pp. 71-75.

[11] Y. Sangho, H. Junyoung, C. Yookun and H. Jiman, "PEACH: Power-Efficient and adaptive Clustering Hierarchy Protocol for Wireless Sensor Networks," *Computer Communications*, Vol. 30, No. 14-15, 2007, pp. 2842-2852.

[12] S. Ghiasi, A. Srivastava, X. Yang and M. Sarrafzadeh, "Optimal Energy Aware Clustering in Sensor Networks," *Sensors Journal*, Vol. 2, No. 7, 2002, pp. 258-269.

[13] H. Chan and A. Perrig, "ACE: An Emergent Algorithm for Highly Uniform Cluster Formation," *Lecture Notes in Computer Science*, Springer Berlin/Heidelberg, Vol. 2920, February 2004, pp. 154-171.

[14] O. Younis and S. Fahmy, "Distributed Clustering in Ad-hoc Sensor Networks: A Hybrid, Energy-Efficient Approach," *Proceedings of IEEE INFOCOM*, Hong Kong, Vol. 1, 2004, pp. 629-640.

[15] J. Y. Cheng, S. J. Ruan, R. G. Cheng and T. T. Hsu, "PADCP: Poweraware Dynamic Clustering Protocol for Wireless Sensor Network," *IFIP International Conference on Wireless and Optical Communications Networks*, Bangalore, 13 April 2006, pp. 1-6.

[16] G. Smaragdakis, I. Matta and A. Bestavros, "SEP: A Stable Election Protocol for Clustered Heterogeneous Wireless Sensor Networks," *Proceedings of Second International Workshop on Sensor and Actor Network Protocols and Applications* (*SANPA*), Boston, 2004, pp. 251-261.

[17] M. Haase and D. Timmermann, "Low Energy Adaptive Clustering Hierarchy with Deterministic Cluster-Head Selection," 4*th International Workshop on Mobile and Wireless Communications Network*, San Jose, 2002, pp. 368-372.

[18] W. B. Heinzelman, A. P. Chandrakasan and H. Balakrishnan, "An Application-Specific Protocol Architecture for Wireless Microsensor Networks," *IEEE Transactions on Wireless Communications*, Vol. 14, No. 4, October 2002, pp. 660-670.

[19] C. Nam, H. Jeong and D. Shin, "The Adaptive Cluster Head Selection in Wireless Sensor Networks," *Proceedings of the IEEE International Workshop on Semantic Computing and Applications*, Incheon, 2008, pp. 147-149.

[20] A. Giridhar and P. R. Kumar, "Maximizing the Functional Lifetime of Sensor Networks," *Fourth International Symposium on Information Processing in Sensor Networks*, Los Angeles, 15 April 2005, pp. 5-12.

[21] X. Wang, W. Gu, S. Chellappan, K. Schosek and D. Xuan, "Lifetime Optimization of Sensor Networks under Physical Attacks," *Proceedings of IEEE International Conference on Communications* (*ICC*), Seoul, Vol. 5, May 2005, pp. 3295-3301.

[22] H. Zhang and J. C. Hou, "Maximizing α-Lifetime for Wireless Sensor Networks," 3*rd International Workshop on Measurement, Modeling, and Performance Analysis of Wireless Sensor Networks*, San Diego, 21 July 2005, pp. 70-77.

[23] V. Rai and R. N. Mahapatra, "Lifetime Modeling of a Sensor Network," *Proceedings of Design, Automation and Test in Europe*, Munich, Vol. 1, 7-11 March 2005, pp. 202-203.

[24] M. U. Ilyas and H. Radha, "Increasing Network Lifetime of an IEEE 802.15.4 Wireless Sensor Network by Energy Efficient Routing," *IEEE International Conference on Communications*, Istanbul, Vol. 9, June 2006, pp. 3978-3983.

[25] L. Shi, A. Capponi, K. H. Johansson and R. M. Murray, "Sensor Network Lifetime Maximization Via Sensor Trees Construction and Scheduling," *Third International Workshop on Feedback Control Implementation and Design in Computing Systems and Networks*, Annapolis, June 2008.

[26] P. Berman, G. Calinescu, C. Shah and A. Zelikovsky, "Power Efficient Monitoring Management in Sensor Networks," *IEEE Wireless Communication and Networking Conference*, Atlanta, March 2004, pp. 2329-2334.

[27] D. Brinza and A. Zelikovsky, "DEEPS: Deterministic Energy-Efficient Protocol for Sensor Networks," 2*nd ACIS International Workshop on Self-Assembling Wireless Networks*, Las Vegas, 2006, pp. 261-266.

[28] A. Aung, "Distributed Algorithms for Improving Wireless Sensor Network Lifetime with Adjustable Sensing Range," M.S. Thesis, Georgia State University, 2007.

[29] S. G. Akojwar and R. M. Patrikar, "Improving Life Time of Wireless Sensor Networks Using Neural Network Based Classification Techniques with Cooperative Routing," *International Journal of Communications*, Vol. 1, No. 2, 2008, pp. 75-86.

[30] Y. Chen, Q. Zhao, V. Krishnamurthy and D. Djonin, "Transmission Scheduling For Sensor Network Lifetime Maximization: A Shortest Path Bandit Formulation," *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, Toulouse, May 2006, pp. 145-148.

[31] K. Dasgupta, K. Kalpakis and P. Namjoshi, "Improving the Lifetime of Sensor Networks via Intelligent Selection of Data Aggregation Trees," *Proceedings of the Communication Networks and Distributed Systems Modeling and Simulation Conference*, 2003, pp. 19-23.

[32] H. Kang and X. Li, "Power-Aware Sensor Selection in Wireless Sensor Networks," *Proceedings of the* 5*th International Conference on Information Processing in Sensor Networks* (*IPSN*), 2006.

[33] M. Cardei and D. Du, "Summary on Improving Wireless Sensor Network Lifetime through Power Aware Organization," Seminar on Theoretical Computer Science, *Wireless Networks*, Vol. 11, No. 3, 2005, pp. 333-340.

[34] J. Park and S. Sahni, "An Online Heuristic for Maximum Lifetime Routing in Wireless Sensor Networks," *IEEE Transactions on Computers*, Vol. 55, No. 8, August 2006, pp. 1048-1056.

[35] J. Chang and L. Tassiulas, "Maximum Lifetime Routing

in Wireless Sensor Networks," *IEEE/ACM Transactions on Networking*, Vol. 12, No. 4, 2004, pp. 609-619.

[36] P. Djukic and S. Valaee, "Maximum Network Lifetime in Fault Tolerant Sensor Networks," *IEEE Global Telecommunications Conference, GLOBECOM*' 05, St. Louis, Vol. 5, December 2005, pp. 3106-3011.

[37] Y. T. Hou, Y. Shi, H. D. Sherali and S. F. Midkiff, "Prolonging Sensor Network Lifetime with Energy Provisioning and Relay Node Placement," 2*nd Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks*, Santa Clara, September 2005, pp. 295-304.

[38] M. Cardei, J. Wu, M. Lu and M. O. Pervaiz, "Maximum Network Lifetime in Wireless Sensor Networks with Adjustable Sensing Ranges," *IEEE International Conference on Wireless and Mobile Computing, Networking and Communications*, Montreal, Vol. 3, August 2005, pp. 438-445.

[39] K. Kalpakis, K. Dasgupta and P. Namjoshi, "Efficient Algorithms for Maximum Lifetime Data Gathering and Aggregation in Wireless Sensor Networks," *Computer Networks*, Vol. 42, No. 6, 2003, pp. 697-716.

[40] R. Madan and S. Lall, "Distributed algorithms for maximum lifetime routing in wireless sensor networks," *IEEE Transactions on Wireless Communications*, Vol. 5, No. 8, August 2006, pp. 2185-2193.

[41] J. C. Choi and C. W. Lee, "Energy Modeling for the Cluster-based Sensor Networks," *Proceedings of the* 6*th IEEE International Conference on Computer and Information Technology*, Seoul, September 2006, p. 218.

[42] S. Selvakennedy and S. Sinnappan, "An Energy-Efficient Clustering Algorithm for Multihop Data Gathering in Wireless Sensor Networks," *Journal of Computers*, Vol. 1, No. 1, April 2006, pp. 40-47.

[43] A. B. M. A. A. Islam, "A Novel Approach To Cluster Heterogeneous Sensor Network (CHSN)," *MSc Engineering Thesis*, Vol. 4, May 2009, pp. 31-41.

[44] A. B. M. A. A. Islam, C. S. Hyder, M. H. Kabir and M. Naznin, "Finding the Optimal Percentage of Cluster Heads from a New and Complete Mathematical Model on LEACH," *Wireless Sensor Network*, Vol. 2, No. 2, February 2010.pp. 129-140.

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

[46] D. Song, "Probabilistic Modeling of Leach Protocol and Computing Sensor Energy Consumption Rate in Sensor Networks," Technical Report, Texas A & M University, February 2005.

[47] T. Murata and H. Ishibuchi, "Performance Evaluation of Genetic Algorithms for Flowshop Scheduling Problems," *Proceedings of* 1*st IEEE Conference Evolutionary Computation*, Orlando, Vol. 2, June 1994, pp. 812-817.

[48] G. E. P. Box and M. E. Muller, "A Note on the Generation of Random Normal Deviates," *Annals of Mathematical Statistics*, Vol. 29, No. 2, 1958, pp. 610-611.

Wireless Sensor Network, **2010, 2, 555-561**
doi:10.4236/wsn.2010.27067 Published Online July 2010 (http://www.SciRP.org/journal/wsn)

Performance Analysis of Downlink MIMO WCDMA Systems Using Antenna Selection in Transmitter and MRC Plus LDD in Receiver over Correlated Nakagami-Fading Channels

Siavash Ghavami[1], Mahmood Mohassel Feghhi[2], Bahman Abolhassani[2]
[1]*School of Electrical & Computer Engineering, University of Tehran, Tehran, Iran*
[2]*School of Electrical Engineering, Iran University of Science and technology, Tehran, Iran*
E-mail: *s.ghavami@ece.ut.ac.ir, mmohaselfeghhi@ee.iust.ac.ir, abolhassani@iust.ac.ir*
Received April 9, 2010; *revised May* 4, 2010; *accepted* May 12, 2010

Abstract

In this paper bit error rate (BER) performance is analyzed for multiple input-multiple output (MIMO) communications systems using antenna selection in the transmitter, maximal ratio combining (MRC) and linear de-correlating detector (LDD) in the receiver in wide band code division multiple access (WCDMA) downlink channels with correlated Nakagami fading. The MRC maximizes signal to noise ratio of the received signal, then the LDD cancels out multiple access interference (MAI). Theoretical results are validated using computer simulations. Moreover, a pilot based estimation method is proposed to jointly estimate the channel gains and the rows of the LDD operator. Simulation results show that using this proposed method, diversity order is maintained in the receiver. Furthermore, our analysis shows the spectral efficiency degradation due to the pilot based strategy is negligible.

Keywords: MIMO, Transmitter Antenna Selection, WCDMA, Maximum Ratio Combining

1. Introduction

Wide band code division multiple accesses (WCDMA) has been proposed to satisfy ever-increasing demands for higher data rates, as well as to allow more users to simultaneously access the network [1]. So, it is employed in the third generation mobile networks to provide multimedia services with required qualities. Multiuser detectors (MUDs) are used to detect the desired signal and to simultaneously cancel out interferences coming from co-users in WCDMA systems [2]. In downlink scenario, blind multiuser detectors are proposed for multiple access interference (MAI) cancellation [3], but use of these detectors increases computational complexity of mobile stations (MSs). Another approach for MAI cancellation in downlink multiuser scenario is the precoding method at the base station (BS) [4], but it requires error free links between each MS and the BS, which is not the case in practical scenarios.

Multiple input-multiple-output (MIMO) systems significantly increase system capacity and improve performance [5,6] at the cost of increasing hardware complexity by increasing the number of transmitting and receiving antennas. Transmitter antenna selection (TAS) can reduce the cost of multiple antennas and at the same time can retain many advantages of MIMO systems. A combined transmitter antenna selection and maximal ratio combining (MRC) has been proposed in [7]. This method selects the transmitter antenna that maximizes the total received signal power at the receiver. Inactivating other transmitter antennas reduces the hardware complexity; furthermore, this method reduces the number of radio channels used in a MIMO system. Bit error rate (BER) performance of this method has been analyzed in independent and correlated Rayleigh fading channels, respectively in [8] and [9]. As well, recently a BER performance analysis of TAS/MRC has been studied in correlated Nakagami fading channels [10], however it is for a single user scenario in a non CDMA system.

In this paper, the downlink scenario of MIMO WCDMA systems using TAS/MRC plus LDD has been studied. MIMO is a strong tool for capacity increasing and performance improvement. But the limitation in the size of a MS necessitates employing receiver antennas with a

small distance among them; therefore the correlation among channel gains should be considered even in rich scattering environments. For hardware complexity reduction and keeping diversity order in the base station, TAS in the transmitter is a good candidate, which is considered in this paper. Furthermore for reducing effect of MAI, a linear de-correlating detector (LDD), which is a sub optimum multi user detector, is used in the receiver. The LDD has simple structure with good performance for MAI mitigation [2]. In this paper, the BER performance of a downlink WCDMA system in the correlated Nakagami fading channels is analyzed, and theoretical results are validated using computer simulations. MAI cancellation using MUDs needs to know the user's and co-users' spreading sequences, which increases the complexities of MSs. In this paper, we consider the case in which the links between each MS and the BS are error prone. So, precoding techniques can not be applied. Furthermore, we give up the blind MUDs for their high computational complexities [11]. In this paper, we propose a low complexity pilot based channel estimation method for the joint estimation of the channel gains and the rows of the LDD operator in order to cancel out the MAI. The proposed method does not require the spreading sequences of all active users (which are not available in the MS), as well as the calculation of the inverse cross correlation matrix. So, using this estimation method, the MSs can cancel out the MAI with-out prior knowledge about spreading sequences of co-users by decorrelating users' signals.

The remaining of this paper is organized as follows. In Section 2, system models are considered. BER performance analysis has been presented in Section 3. Joint estimation of channel and the LDD operator is proposed in Section 4. Simulation results are presented in Section 5 in order to validate our performance analysis and evaluate the performance of our proposed joint estimation method. Finally conclusions are presented in Section 6.

2. System Model

We consider the downlink scenario of a WCDMA system. The *j*th antenna of the MS receives signals of *K* users which have been sent by the *i*th transmitter antenna in the base station. It is given by

$$r_{i,j}(t)=\sum_{k=1}^{K}\sum_{l=-\infty}^{\infty}A_k d_k[l]h_{i,j}(t-lT_s-\tau_{i,j})+n(t), \qquad (1)$$

where A_k, $d_k[l]$, T_s and $\tau_{i,j}$ are, respectively, the received amplitude, *l*th data symbol of *k*th user, symbol period and path delay between *i*th transmitter antenna and *j*th receiver antenna and $n(t)$ is additive white Gaussian noise. The $h_{i,j}$ is expressed by:

$$h_{i,j}(t)=\frac{1}{\sqrt{N}}\sum_{m=0}^{N-1}c_k[m]p_{i,j}(t-mT_c), \qquad (2)$$

where N is the processing gain, $T_c=T/N$ is the chip time, $p_{i,j}(t)$ is the convolution of three components: the chip pulse shaping waveform, the channel filter between the *i*th transmitter antenna and the *j*th receiver antenna (which represents the channel echoes) and the receiver filter, with unit energy. $c_k[m]$ is the value of the *m*th chip of *k*th user's spreading sequence with $|c_k[m]|=1$. Data symbols $\{d_k[l]\}$ of different users are independent with identical distributions (i.i.d). The channel between the *i*th transmitter and the *j*th receiver antenna denotes as $\beta_{i,j}(t)=\alpha_{i,j}(t)e^{j\varphi_{i,j}(t)}$, is assumed to be a quasi-static fading and its envelope $\alpha_{i,j}(t)$ follows Nakagami-m distribution

$$p_{\alpha_{i,j}}(x)=\frac{2}{\Gamma(m)}\left(\frac{m}{\bar{\delta}}\right)^m x^{2m-1}\exp\left(\frac{-mx^2}{\bar{\delta}}\right) \qquad (3)$$

where $m=\bar{\delta}^2\Big/E\left[\left(\alpha_{i,j}^2-\bar{\delta}\right)^2\right]$, $\bar{\delta}=E\left(\alpha_{i,j}^2\right)$, $\Gamma(n)=\int_0^\infty u^{n-1}e^{-u}du$. Therefore $\delta_{i,j}=\alpha_{i,j}^2$ has gamma distribution,

$$p_{\delta_{i,j}}(x)=\left(\frac{m}{\bar{\delta}}\right)^m\frac{x^{m-1}}{\Gamma(m)}\exp\left(\frac{-mx}{\bar{\delta}}\right). \qquad (4)$$

2.1. Transmitter Antenna Selection

The TAS is performed so that the total received signal power is maximized. Mathematically, it is equivalent to selection antenna such that:

$$i_s=\arg\max_{1\le i\le L_t}\left(\sum_{j=1}^{L_r}\left|\beta_{i,j}\right|^2\right). \qquad (5)$$

The MS sends i_s using only $\lceil\log_2(L_t)\rceil$ bits, where $\lceil\cdot\rceil$ denotes ceiling operation. In performance analysis, channel estimation is assumed to be performed perfectly at the receiver, also the feedback link between the receiver and the transmitter is considered to be perfect and without delay (using forward error correction (FEC) it is possible to send a few flag bits, *i.e.*, $\lceil\log_2(L_t)\rceil$, without error even in error prone link). The size of a Ms limits the use of antenna diversity and this makes the channels correlated. The covariance matrix among channel power gains, e.g., $\boldsymbol{\delta}_{i_s}=\left[\delta_{i_s,1}\ \ \ldots\ \ \delta_{i_s,L_r}\right]^T$ in the receiver is

given by

$$\mathbf{\Omega} = E\left\{\boldsymbol{\delta}_{i_s}\boldsymbol{\delta}_{i_s}{}^{H}\right\}, \quad (6)$$

where, $E\{\cdot\}$ and ^{H}, respectively denote expectation and Hermitian operations.

2.2. BER Performance Analysis

In this section, BER performance of TAS/MRC plus LDD in the downlink scenario of WCDMA systems is analyzed. In the following, a bold capital letter denotes a matrix (**A**), a bold small letter denotes a vector (**a**), and an unbolded letter denotes a scalar (a or A). Samples of the channel are considered to be constant for each symbol; such that for the *l*th symbol defined as $\beta_{i,j}[l] = \beta_{i,j}(lT)$. Defining $\mathbf{A} = \mathrm{diag}(A_1,...,A_k)$, $\mathbf{C} = [\mathbf{c}_1^T, \ ... \ , \mathbf{c}_K^T]$, $\mathbf{c}_k = N^{-0.5}\left[c_k[0],...,c_k[N-1]\right]$, $\mathbf{d}[l] = \left[d_1[l],...,d_K[l]\right]^T$ and $\mathbf{n}[l] = \left[n_j^0[l],...,n_j^{N-1}[l]\right]^T$, which contains the noise samples in the *j*th receiver antenna, the received signal form the *i*th transmitter antenna to the *j*th receiver antenna is given by

$$\mathbf{r}_{i,j}[l] \ = \beta_{i,j}[l]\mathbf{CAd}[l] + \mathbf{n}[l]. \quad (7)$$

For any linear detector combined with MRC, the decision variable, $\hat{\mathbf{d}}[l]$ is obtained by a linear combination of $\mathbf{r}_{i,j}[l]$, *i.e.*,

$$\hat{\mathbf{d}}[l] = \mathrm{sign}\{\mathrm{Re}(\mathbf{D}\sum_{j=1}^{L_r}\beta_{i,j}^{*}[l]\mathbf{r}_{i,j}[l]))\}, \quad (8)$$

where the matrix **D** represents the operation of the multiuser detector. For the LDD, **D** is the Moore-Penrose generalized inverse of the code matrix **C** [2], given by

$$\mathbf{D}_{\mathrm{LDD}} = (\mathbf{C}^{\mathrm{T}}\mathbf{C})^{-1}\mathbf{C}^{\mathrm{T}} = \mathbf{R}^{-1}\mathbf{C}^{\mathrm{T}}, \quad (9)$$

where $\mathbf{R} = \mathbf{C}^{\mathrm{T}}\mathbf{C}$ is a $K \times K$ matrix containing the autocorrelation coefficients of the users' spreading codes. In (9), it is assumed that *K* users' codes are linearly independent, to guarantee the existence of $\mathbf{R}^{-1}$. The receiver output is obtained as follows

$$\begin{aligned} \mathrm{v}[l] &= \sum_{j=1}^{L_r}\mathbf{R}^{-1}\mathrm{C}^{T}\beta_{i,j}^{*}[l]\mathbf{r}[l] \\ &= \sum_{j=1}^{L_r}\mathbf{R}^{-1}\left|\beta_{i,j}[l]\right|^2\mathrm{C}^{T}\mathrm{CA}\mathbf{d}[l] + \mathbf{R}^{-1}\mathrm{C}^{T}\mathbf{n}[l] \\ &= \sum_{j=1}^{L_r}\left|\beta_{i,j}[l]\right|^2\mathrm{A}\mathbf{d}[l] + \mathbf{R}^{-1}\mathrm{C}^{T}\mathbf{n}[l]\,. \end{aligned} \quad (10)$$

The $\varepsilon_k = (\mathbf{R}^{-1})_{k,k} \geq 1$ is the noise enhancement factor produced by the de-correlating operation, therefore noise variance in the receiver side is increased to $\sigma_{n,LDD}^2 = \varepsilon_k\sigma_n^2$, where σ_n^2 is variance of $\mathbf{n}[l]$, which is equal to $N_0/2$.

In the first step of the BER performance analysis, the probability density function (PDF) of maximum channel gains, which has been selected by criterion given by (5) must be obtained, $\delta_{\max} = \max_{1\leq i\leq L_t}\{\delta_i\}$, where $\delta_i = \sum_{j=1}^{L_r}\delta_{i,j}$. The PDF of $\delta_{\max}$ is obtained as [12]

$$p_{\delta_{\max}}(x) = L_t(F_{\delta_i}(x))^{L_t-1}f_{\delta_i}(x), \quad (11)$$

where $F_{\delta_i}(x)$ and $f_{\delta_i}(x)$ are respectively, the cumulative distribution function (CDF) and PDF of δ_i. $F_{\delta_i}(x)$ and $f_{\delta_i}(x)$ for multi user scenario over correlated Nakagami fading channels are extracted similar to those of a single user scenario over correlated Nakagami fading channels, which proposed in [10].

The TAS in the transmitter and MRC in the receiver increase the received SNR of *k*th user to

$$\xi_k = \eta_k\sum_{i=1}^{L_r}|\beta_{i_s j}|^2, \quad (12)$$

in which $\eta_k = E_b/\left(\varepsilon_k\sigma_n^2\right)$, where E_b is average energy per bit in the transmitter. For the BPSK modulation the instantaneous BER of the *k*th user is obtained as $Q\left(\sqrt{2\xi_k}\right)$, where $Q(\cdot)$ is the Q-function. The BER is obtained by integration of instantaneous BER from zero to infinity, as [13]

$$P_{e,k} = \int_0^{\infty}Q(\sqrt{2\xi_k})p_{\xi_k}(\xi_k)d\xi_k\,. \quad (13)$$

Since ξ_k in (12) is $\eta_k\delta_{max}$, the PDF of ξ_k is also obtained similar to (11), e.g., $p_{\xi_k}(\cdot) = p_{\delta_{\max}}(\cdot)$. Hence, calculation of integral in (13) can be performed using $p_{\delta_{\max}}(\cdot)$. Calculation of above integral, which is related to BER performance of multi-user scenario, is similar to the calculation of BER performance in a single user scenario, which has been performed in [10]. The final BER is obtained as (14), where, ! denotes the factorial operation and $\{\lambda_l, 1\leq l\leq L_r\}$ are the eigenvalues of the matrix $\sqrt{\mathbf{\Omega}}/m$. As well, ψ_{lr} is defined as (15), and the expressions of a_j, b_j, c_j can be obtained using the multinomial theorem [14].

3. Joint Estimation of Channel Gains and LDD Operator

In this section, effect of joint estimation of the channel gains and the rows of the LDD operator is investigated.

$$P_{e,k} = L_t \sum_{l=1}^{L_r} \sum_{r=1}^{m} \frac{\psi_{lr}}{(\lambda_l)^r \Gamma(r)} \sum_{j=0}^{(L_t-1)(L_r\frac{m(m+1)}{2}+1)} a_j \left(\frac{\lambda_l}{b_j+\lambda_l}\right)^{c_j+r} \times \left(c_j+r-1/2\right)! \times \left(1-\sqrt{\frac{E_b\lambda_l}{E_b\lambda+\left(b_j+\lambda_l\right)\varepsilon_k N_0}}\right)^{c_j+r}$$
$$\times \sum_{s=0}^{c_j+r-1} 2^{-s} \binom{c_j+r-1+s}{s} \left(1+\sqrt{\frac{E_b\lambda_l}{E_b\lambda+\left(b_j+\lambda_l\right)\varepsilon_k N_0}}\right)^{s} \tag{14}$$

$$\psi_{lr} = \begin{cases} \left(\prod_{k=1,k\neq l}^{L_r}\left(1-\frac{\lambda_k}{\lambda_i}\right)^{-m}\right), & r<m \\ \frac{1}{(m-r)!(-\lambda_l)^{m-r}} \frac{d^{m-r}}{ds^{m-r}}\left(\prod_{k\neq l}(1-s\lambda_k)^{-m}\right)\Big|_{s=1/\lambda_l}, & r=m \\ 0, & r>m \end{cases} \tag{15}$$

In the downlink scenario of a WCDMA system, each MS knows only its own spreading sequence and doesn't have any knowledge about spreading sequence of any other user, however using the LDD in the receiver needs the knowledge of spreading sequences of all other active users. Moreover, even if a mobile user knows the code matrix of all users (e.g., $\mathbf{C}$) it needs to calculate the inverse cross correlation matrix among spreading sequences of all active users (e.g., $\mathbf{R}^{-1}$), which has high computational complexity. Furthermore, the mobile user is allowed to detect only its own data. On the other hand, the total $\mathbf{D}_{\text{LDD}}$ is not required in the mobile. And the *k*th user requires only the *k*th row of the $\mathbf{D}_{\text{LDD}}$ denoted by $\mathbf{D}^k_{LDD}$ in the *l*th detected data as can be seen in the following equation

$$\mathrm{v}_k[l] = \sum_{j=1}^{L_r} \mathbf{D}^k_{LDD} \left|\boldsymbol{\beta}_{i,j}[l]\right|^2 \mathbf{CAd}[l] + \mathbf{n}'[l], \tag{16}$$

where $\mathbf{n}'[l] = \mathbf{D}^k_{LDD}\mathbf{n}[l]$. To obtain $\mathbf{D}^k_{LDD}$ in the receiver, in this paper, a new method is proposed, in which the base station sends $\mathbf{D}^k_{LDD}$ through a pilot signal. In the proposed method, the base station transmits the $\mathbf{D}^k_{LDD}$ periodically, whose period is much larger than the symbol time since the number of mobile users changes much slower than the symbol time. Moreover, the base station spreads the pilot signal for transmitting $\mathbf{D}^k_{LDD}$ using the *k*th spreading sequence. This strategy avoids generating interference and maintains the system security since only the *k*th mobile user can obtain the $\mathbf{D}^k_{LDD}$ for detecting its data. The received pilot symbol is as follows:

$$\mathbf{r}_j^{pilot} = A_k \beta_{i_s,j}[l]\mathbf{c}_k \otimes \mathbf{F}_\mathbf{k} + \mathbf{n}'', \tag{17}$$

where $\mathbf{F}_\mathrm{k} = \mathbf{c}_k \otimes \mathbf{D}^k_{LDD}$, $\mathbf{n}'' = \mathbf{c}_\mathbf{k} \otimes \mathbf{n}$ and $\otimes$ denotes element by element multiplications of two vectors. If $A_k = 1$ then $\mathbf{r}_j^{pilot}$ will be equal to $\beta_{i_s,j}[l]\mathbf{F}_\mathbf{k} + \mathbf{n}''$. **Figure 1** shows block diagram of the proposed strategy for joint channel and LDD operator estimation, in which $\mathrm{TS}_k[l]$ denotes *l*th symbol of the training sequence of *k*th user. Training sequences for LDD operator estimation have been sent in non-overlapping time slots, hence MAI is not produced due to the transmission of pilot signals.

If the pilot symbols length of each user increases, the effect of $\mathbf{n}''$ in (17) reduces and a better estimation of $\mathbf{D}^k_{LDD}$ can be exploited. Error reduction of $\mathbf{D}^k_{LDD}$ estimation is obtained at the expense of increasing the training duration from NT_s to qNT_s, where q is the number of pilot symbols repetition for the desired user.

In the feedback link, $\beta_{i_s,j}[l]\mathbf{F}_k$ is sent from the mobile user to the base-station to be used for the TAS. Since, $\mathbf{F}_k$ is known at the base-station, $\beta_{i_s,j}[l]$ can be easily calculated at the base-station. Exactly like the previous section, feedback link is assumed to be error free and without any delay.

Using pilot based strategy for joint estimation of channel gains and LDD operator reduce the spectral efficiency. Hence, effect of using pilot based strategy on the spectral efficiency is analyzed in this section. The spectral efficiency of joint estimation of channel and LDD operator is defined as,

$$\eta = \frac{TIB}{TTB} = 1 - \frac{qK\lambda}{R_b}, \tag{18}$$

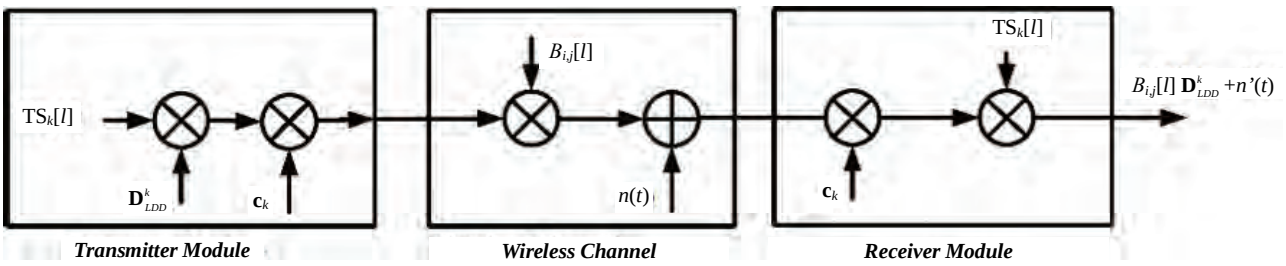


Figure 1. Block diagram of proposed joint channel estimation and LDD operator method.

where, *TIB* and *TTB* are *transmitted information bits* and *total transmitted bits*, respectively; also, λ and R_b are variation rate of number of arriving user to the cell and bit rate, respectively. The spectral efficiency of the proposed method is analyzed for practical values of q, K and λ in the next Section.

In the next section, the currently explained algorithm is used for the joint estimation of channel and the LDD operator. Also, the bit error probability performance of this method is evaluated and compared with the bit error probability of the same system with the perfect estimation of channel and the LDD operator.

4. Simulation Results

In this section, we validate our analysis, equation (14), using computer simulations. We simulate the TAS/MRC plus LDD scheme in flat Nakagami correlated fading channels. Furthermore, the effect of the proposed method for joint estimation of channel and the LDD operator is studied on the BER performance of TAS/MRC plus LDD system.

The base station transmits data with equal amplitudes for each of K = 25 users. Random sequence with length of 63 has been used for short spreading codes. A rectangular pulse shaping waveform $p(t)$ is used for shaping filter in the transmitter side. The type of modulation is BPSK. Simulations are performed for a MIMO system with 2 transmitter antennas and 3 receiver antennas over correlated Nakagami flat fading channels. The normalized correlation antenna matrix is the same as the one given in [9], it is given by a 3 × 3 matrix denoted by $\mathbf{\Omega} = \left[\Omega_1, \Omega_2, \Omega_3\right]$, where $\Omega_1 = \left[1 \quad 0.727 \quad 0.913\right]^T$, $\Omega_2 = \left[0.727 \quad 1 \quad 0.913\right]^T$ and $\Omega_3 = \left[0.913 \quad 0.913 \quad 1\right]^T$.

Figure 2 shows the BER of the TAS/MRC plus LDD in the aforementioned Nakagami correlated fading channels for m = 1, 2 *and* 3. It is obvious that our analytical result given by (14) is validated by computer simulations. Moreover, the system with TAS/MRC plus LDD has a better performance so that it has approximately 3.5 dB SNR gain relative to that of the MRC plus LDD without the TAS at $P_{e,k} = 10^{-4}$ with the diversity order m = 1.

Figure 3 shows the effect of proposed joint estimation of channel gains and the LDD operator on the performance of the TAS/MRC plus LDD with a training sequence having length of either 25 or 50, which needs once (q = 1) or twice (q = 2) repetition of $\mathbf{D}_{\text{LDD}}$ elements, respectively. From **Figure 3**, it is obvious that the diversity order remains almost constant since the slope of the three

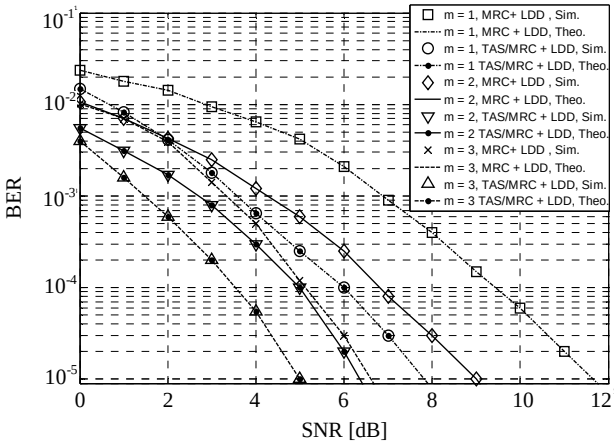


Figure 2. BER performance of MRC plus LDD and TAS/MRC plus LDD.

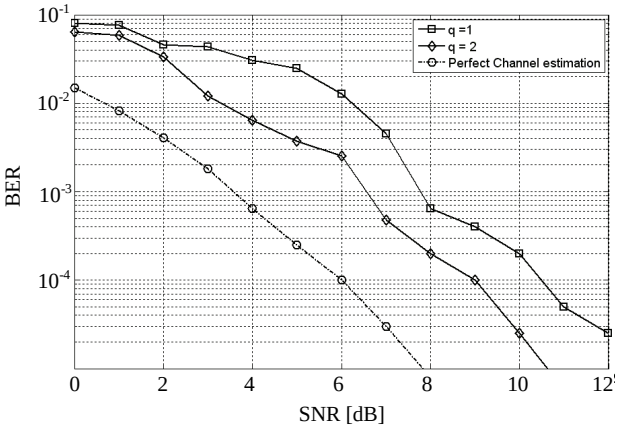


Figure 3. BER performance of TAS/MRC plus LDD for m = 1 with imperfect channel estimation.

curves are approximately the same, and there is only 4.5 dB or 3 dB loss in the SNR at $P_{e,k}=10^{-4}$ due to using the proposed joint estimation of channel and the LDD operator with training sequence whose length is 25 or 50, respectively. It is notable that by increasing q the SNR loss reduces.

The mean square error (MSE) of the proposed joint estimation of channel and the LDD operator is defined as,

$$MSE = E\left[\left\|\left(\mathbf{D}_{LDD}\right)_{k,:} - \left(\hat{\mathbf{D}}_{LDD}\right)_{k,:}\right\|^2\right] \qquad (19)$$

where $\left(\hat{\mathbf{D}}_{LDD}\right)_{k,:}$ is kth row of the estimated LDD operator, $E\{\cdot\}$ denotes expectation operator and $\left\|\left(\mathbf{D}_{LDD}\right)_{k,:} - \left(\hat{\mathbf{D}}_{LDD}\right)_{k,:}\right\|^2$ denotes square norm of matrix. **Figure 4** shows *MSE* of the proposed joint estimation of channel and the LDD operator over AWGN, in terms of number of pilot repetition q, for different values of SNR. It is obvious that by increasing q, *MSE* of $\left(\mathbf{D}_{LDD}\right)_{k,:}$ estimation is reduced.

Figure 5(a) shows the η in terms of the q for different values of active users in the cell with traffic variation of $\lambda = 10$ user/sec and $R_b = 5$ Mbps. It is obvious even with q = 50 bits and k = 90 users, the efficiency is greater than 0.99. **Figure 5(b)** is similar to **Figure 5(a)** with $\lambda = 20$ user/sec. It can be seen with q = 50 bits and k = 90 users, the efficiency is greater than 0.98. **Figure 5(c)** and **5(d)** are similar to **Figure 5(a)** with $\lambda = 50$ user/sec and $\lambda = 100$ user/sec, respectively.

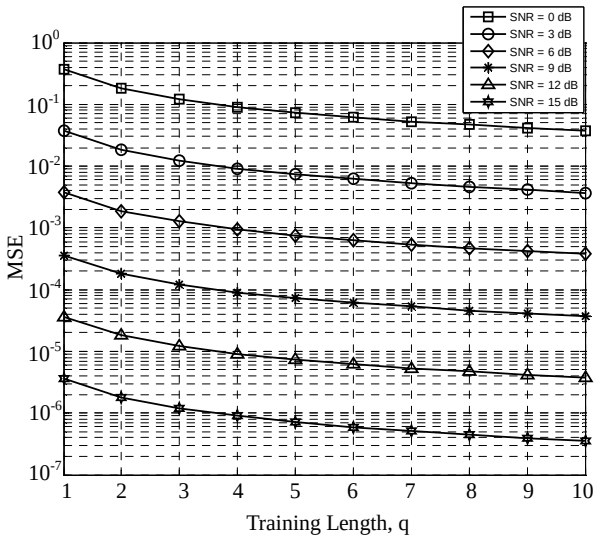
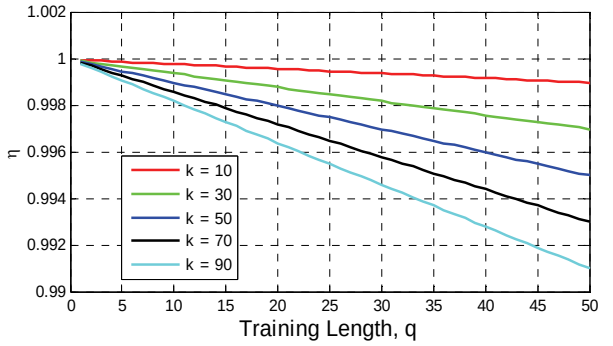
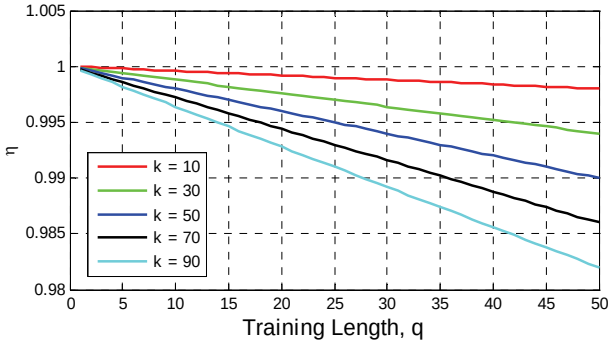


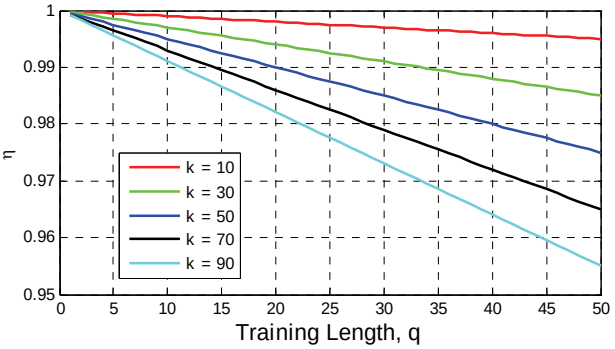
Figure 4. Mean Square error of joint estimation of channel and the LDD operator in terms of training length.



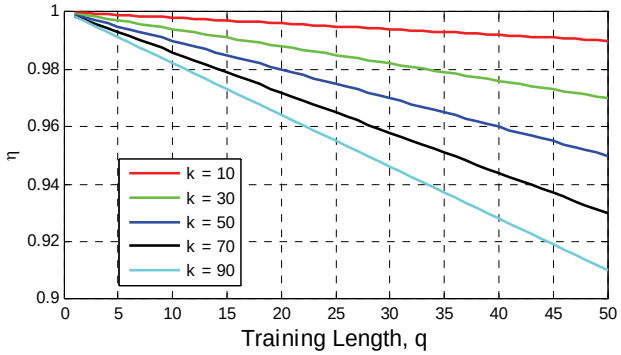
(a) λ_1 = 10 user/sec



(b) λ_2 = 20 user/sec



(c) λ_3 = 50 user/sec



(d) λ_4 = 100 user/sec

Figure 5. Efficiency in terms of number of pilot symbols repetition for different number of active users in cell with (a) variation of input traffic of λ_1 = 10 user/sec, (b) λ_2 = 20 user/sec, (c) λ_3 = 50 user/sec and (d) λ_4 = 100 user/sec.

4. Conclusions

In this paper, the BER performance analysis of the TAS/MRC plus LDD is performed for the downlink WCDMA network in correlated Nakagami fading channels. Equation (14), derived in the analysis, was validated using computer simulations. Also a pilot based channel estimation method is proposed for joint estimating of the channel gains and the LDD operator. Simulation results show that with joint estimation of channel gains and the LDD operator, diversity order is kept in the receiver. Moreover, our analysis shows that the spectral efficiency degradation due to using the pilot based strategy for joint estimation of channel gains and LDD operator is negligible.

5. References

[1] L. B. Milstein, "Wideband Code Division Multiple Access," *IEEE Journal on Selected Areas in Communications*, Vol. 18, No. 8, August 2000, pp. 1344-1354.

[2] S. Verdu, "Multiuser Detection," Cambridge University Press, Cambirdge, 1998.

[3] X. Wang and H. V. Poor, "Blind Multiuser Detection: A Subspace Approach," *IEEE Transactions on Information Theory*, Vol. 44, No. 2, March 1998, pp. 677-690.

[4] B. R. Vojcic and W. M. Jang, "Transmitter Precoding in Synchronous Multiuser Communications," *IEEE Transactions on Communications*, Vol. 46, No. 10, October 1998, pp. 1346-1355.

[5] E. Telatar, "Capacity of Multi-Antenna Gaussian Channels," *European Transactions on Telecommunications*, Vol. 10, No. 6, 1999, pp. 585-595.

[6] G. J. Foschini and M. J. Gans, "On Limits of Wireless Communications in a Fading Environment When Using Multiple Antennas," *Wireless Personal Communications*, Vol. 6, No. 3, March 1998, pp. 311-335.

[7] S. Thoen, L. Van der Perre, B. Gyselinckx and M. Engels, "Performance Analysis of Combined Transmit-SC/Receiver-MRC," *IEEE Transactions on Communications*, Vol. 49, No. 1, 2001, pp. 5-8.

[8] Z. Chen, J. Yuan and B. Vucetic, "Analysis of Transmit antenna Selection/Maximal-ratio Combining in Rayleigh Fading Channels," *IEEE Transactions on Vehicular Technology*, Vol. 54, No. 4, 2005, pp. 1312-1321.

[9] L. Yang, D. Tang and J. Qin, "Performance of Spatially Correlated MIMO Channel with Antenna Selection," *IEE Ectronics Letters*, Vol. 40, No. 20, 2004, pp. 1281-1282.

[10] B. Y. Wang and W. X. Zheng, "Exact BER of Transmitter Antenna Selection/Receiver MRC over Spatially Correlated Nakagami-Fading Channels," *IEEE Conference on Circuits and Systems*, Kos, May 2006, pp. 1370-1373.

[11] Y. Yao and H. V. Poor, "Blind Detection of Synchronous CDMA in Non-Gaussian Channels," *IEEE Transactions on Signal Processing*, Vol. 52, No. 1, January 2004, pp. 271-279.

[12] A. Papoulis, "Probability, Random Variables and Stochastic Processes," 4th Edition, McGraw-Hill, New York, 2001.

[13] J. G. Proakis, "Digital Communications," 4th Edition, McGraw-Hill, New York, 2001.

[14] R. L. Graham, D. B. Knuth and O. Patashnik, "Concrete Mathematics," Addison-Wesley, New York, 1989.

Wireless Sensor Network, **2010, 2, 562-570**
doi:10.4236/wsn.2010.27068 Published Online July 2010 (http://www.SciRP.org/journal/wsn)

Information-Driven Collaborative Processing for Diffusive Source Estimation in Wireless Sensor Networks

Hossein Khonsari, Mohammad Hossein Kahaei
Electrical Engineering Department, Iran University of Science and Technology, Tehran, Iran
E-mail: *hossein.khonsari@gmail.com, kahaei@iust.ac.ir*
Received April 6, 2010; *revised April* 21, 2010; *accepted May* 2, 2010

Abstract

This paper discusses an accurate distributed algorithm for diffusive source localization while maintaining the low energy consumption of sensor nodes in wireless sensor networks. In this algorithm, the sensor selection scheme based on the information utility measure is used. To update the estimation in each selected node, a neighborhood radius equal to the communication range of the sensor nodes is defined and all sensors located in the neighborhood circle, whose radius is equal to the neighborhood radius and the selected node is its centre, collaborate their information. To decrease the energy consumption, the neighborhood radius is reduced gradually based on the error covariance value of the estimation. In addition, this paper includes a new method for the initial point calculation which is important in the recursive methods used for distributed algorithms in wireless sensor networks. Numerical examples are used to study the performance of the algorithms. Simulation results show the accuracy of the new algorithm becomes better while its energy consumption is low enough.

Keywords: Information-Driven Collaborative Processing, Wireless Sensor Network, Diffusive Source Localization

1. Introduction

Densely scattered low-cost sensor nodes provide a rich and complex information source about the sensed world. A major application of wireless sensor networks is to perform monitoring tasks like localization and tracking of one or more targets in their coverage field. In this paper, we focus on the localization of a diffusive source. The problem under consideration has application in the fields of security, environmental and industrial monitoring and pollution control [1]. Since in a typical wireless sensor network each sensor node has limited battery power, efficient collaborative signal processing algorithms that consume less energy for computation and less bandwidth for communication is needed [2]. In centralized estimation methods, sensor nodes transmit their row observations to a fusion centre for processing [3-5]. However, some of their innate properties such as high-energy consumption limit their use in a wireless sensor network [6]. Therefore, recent research has concentrated on developing distributed processing.

Most of the current distributed estimation methods are categorized in to two groups. In one group, the distributed estimation is developed using the sequential Bayesian method [7-9]. In this method, the state belief (posterior density function) is updated in the selected nodes and when the quality of estimation is good enough (base on the predefined threshold), the sensor selection is stopped and the location is estimated. The most important disadvantage of this type of method is that the convergence of the estimation is not easy to be proven [6]. In the other group, the distributed estimation is yielded by implementing the common centralized estimation methods like Maximum Likelihood (ML) estimation in a distributed manner [10]. As in this type the communication burden is high, the most important challenge is to develop the accuracy of algorithms while considering the total energy consumption. In this paper, we propose a new method to improve the accuracy of diffusive source estimation while maintaining the energy consumption in a reasonable level.

The recent relevant work is presented in [6] where a distributed Information-Driven Collaborative Processing (IDCP) is derived. The essential point in this algorithm is that by applying information utility measure, a sequence of sensor nodes is chosen to reduce the required data communications under the framework of ML estimation;

however, for estimation update, only selected node observations are used. In this paper, we propose the modification of the above method to improve the estimation performance. In our algorithm called Modified IDCP (MIDCP), each node calculates the new estimation of the target location by averaging over the information of its own observation and neighbors. However, the energy consumption increases because of large amount of transmission. To mitigate this energy consumption we propose Energy Efficient MIDCP (EFMIDCP) in which the neighborhood radius is decreased gradually based on the estimation covariance value. In general, Iterative methods used in distributed algorithms may not yield the ML estimate if the initial value is not selected carefully. Therefore, we propose a new method to calculate initial value for the iteration algorithms in our scenario.

This paper is organized as follows. In Section 2, we present the physical and statistical models of the released substance distribution. The IDCP algorithm is investigated in Section 3. In Section 4, we propose the MIDCP and EFMIDCP algorithms. Section 5 includes a new method to calculate initial estimation in all of these iterative algorithms. In Section 6 the numerical examples are used to demonstrate the performance of the proposed methods. Conclusions are presented in Section 7.

2. Problem Formulation

2.1. Physical Model of a Difusive Source

Let $c(\mathbf{r},t)$ denote the substance concentration diffuseion at a position $\mathbf{r}=(x,y)$ and time *t*. For a source-free volume and space-invariant diffusivity, without biomedical reaction during the transport of biochemical agents, the concentration of a diffusing substance follows the well-known diffusion equation [5]:

$$\frac{\partial c}{\partial t}=k\nabla^2 c=k\left(\frac{\partial^2 c}{\partial x^2}+\frac{\partial^2 c}{\partial y^2}\right) \tag{1}$$

Where *k* is called the diffusivity in units of kg/m^{2}s and dependence on *r* and *t* is omitted to simplify the presentation. It has been proved that in a rectangular parallelepiped space and for certain types of initial and boundary conditions, the solution of (1) is the product of the solutions of the two single spatial-variable problems [5]. That is,

$$c(\mathbf{r},t)=c_x(\mathbf{r},t)c_y(\mathbf{r},t) \tag{2}$$

where $c_x(\mathbf{r},t)$ is the solution of the diffusion equation with single spatial-variable *x:*

$$\frac{\partial c_x(x,t)}{\partial t}=k\frac{\partial^2 c(x,t)}{\partial x^2} \tag{3}$$

with the similar definition for $c_y(y,t)$. We use the distribution equation which is derived in [5] for a continuous point diffusive source (our desired source). The concentration of our source in a semi-infinite medium is obtained using the following integral:

$$c(\mathbf{r},t)=\mu\int_{t_I}^{t}\frac{2}{8\left[\pi k(t-\tau)\right]^{\frac{3}{2}}}\times\left\{\exp\left[-\frac{|\mathbf{r}-\mathbf{r}_0|}{4k(t-\tau)}\right]\right\}d\tau \tag{4}$$

where $\mathbf{r_0}=(x_0,y_0)$ represents the exact source location, μ is the substance realizing rate and t_I is the initial time of diffusion.

2.2. Statistical Measurement Model

We assume that a continuous point diffusive source is at position $\mathbf{r_0}=(x_0,y_0)$, from which the diffusion substance is liberated continuously at a certain rate μ and at time t_I. The sensor nodes of a WSN are deployed in which each sensor node is located at a known position and can measure the substance concentration from the diffusive sources. The measurement taking by sensor node *i* at time t_j can be written as:

$$\begin{aligned}&y(\mathbf{r}_i,t_j)=c(\mathbf{r}_i,t_j)+b+e(\mathbf{r}_i,t_j)\\&e(r_i,t_j)\sim N(0,\sigma^2),\end{aligned} \tag{5}$$

where $\mathbf{r}_i$ is the known location of sensor node *I*, the term $c(\mathbf{r}_i,t_j)$ is the concentration distribution shown in Equation (4), *b* is a constant bias term representing the sensor's response to foreign substances assumed to be a known and $e(\mathbf{r}_i,t_j)$ is the sensor measurement's Gaussian noise independent in space and correlated in time, *i.e.,*

$$\mathbf{E}\left[e(\mathbf{r}_{i_1},t_{j_1})e(\mathbf{r}_{i_2},t_{j_2})\right]=\begin{cases}\sigma_{j_1j_2} & i_1=i_2\\ 0 & i_1\neq i_2\end{cases}. \tag{6}$$

Therefore, for each sensor node, we have $\mathrm{e}_i=[e_{i,1},...,e_{i,N}]^T$, an N-dimension Gaussian distributed random vector with the mean zero and covariance matrix Σ_i. By denoting $y_{ij}=y(\mathbf{r}_i,t_j)$, $e_{ij}=e(\mathbf{r}_i,t_j)$ and $\mu\alpha_{ij}(\boldsymbol{\theta})=c(\mathbf{r}_i,t_j)$, we can rewrite (1) as:

$$y_{ij}=\mathbf{a}_{ij}^T(\boldsymbol{\theta})\mathbf{x}+e_{ij}. \tag{7}$$

Where $\mathbf{a}_{ij}(\boldsymbol{\theta})=\left[\alpha_{ij}(\boldsymbol{\theta}),1\right]^T$ and $c(r_i,t_j)=\mu\alpha_{ij}(\boldsymbol{\theta})$,

$\mathbf{x}=[\mu,b]^T$ is called the linear parameter vector and $\boldsymbol{\theta}=[x_0,y_0]^T$ represents the source parameters.

3. Information-Driven Collaborative Processing

M sensor nodes are activated to estimate the location of the diffusive source located in the network's range. The covariance matrix, Σ_i, is known for each node because this matrix can be estimated during the calibration step in practical applications. The measurements $\{y_1,...,y_M\}$ of all sensor nodes can be written as a vector form [6]:

$$\mathbf{y} = \mathbf{A}(\boldsymbol{\theta})\mathbf{x} + \mathbf{e} \tag{8}$$

where $y=\left[y_1^T,...,y_M^T\right]^T$ is a $MN\times 1$ vector, $A(\theta)=\left[A_1^T(\theta),\right.$ $\left....,A_M^T(\theta)\right]^T$ is a $MN\times 2$ matrix and $e=\left[e_1^T,...,e_M^T\right]^T$ represents the additive Gaussian noise. Regarding to the assumptions, it follows a *MN*-dimension multiple normal distribution with mean zero and covariance matrix $\boldsymbol{\Sigma}$ with $\boldsymbol{\Sigma}=\mathrm{diag}\{\boldsymbol{\Sigma}_1,...,\boldsymbol{\Sigma}_M\}$, $\boldsymbol{\Sigma}$ is diagonal because the measurement noise of each two sensors is independent. We assume each sensor-node takes N samples.

According to the above measurement model, the log-likelihood function of the measurements vector **y** is [6]:

$$\begin{aligned} L(\mathbf{y};\boldsymbol{\theta}) &= -\frac{M}{N}\log 2\pi - \frac{1}{2}\sum_{i=1}^{M}\log|\Sigma_i| \\ &\quad -\frac{1}{2}\sum_{i=1}^{M}\left[\mathbf{y}_i - A_i(\boldsymbol{\theta})\mathbf{x}\right]^T\Sigma_i^{-1}\left[\mathbf{y}_i - A_i(\boldsymbol{\theta})\mathbf{x}\right]. \end{aligned} \tag{9}$$

By maximizing this log-likelihood function, the ML estimation of $\boldsymbol{\theta}$ is obtained:

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \left\{\sum_{i=1}^{M}\left[\mathbf{y}_i - \mathbf{A}_i(\boldsymbol{\theta})\mathbf{x}\right]^T \boldsymbol{\Sigma}_i^{-1}\left[\mathbf{y}_i - \mathbf{A}_i(\boldsymbol{\theta})\mathbf{x}\right]\right\}. \tag{10}$$

To derive a distributed estimation, it is necessary to use an incremental optimization method. Therefore, the Gauss- Newton method is used to implement a distributed estimation. When employing this method to solve the non-linear weighted least-square problem in (10), an iterative process is obtained to update the estimate of $\boldsymbol{\theta}$ [6]:

$$\begin{aligned} \boldsymbol{\theta}^{(k+1)} &= \boldsymbol{\theta}^{(k)} \\ &\quad +\left[H^T(\boldsymbol{\theta}^{(k)})\Sigma^{-1}H(\boldsymbol{\theta}^{(k)})\right]^{-1} \\ &\quad \times H(\boldsymbol{\theta}^{(k)})\Sigma^{-1}(\mathbf{y} - A(\boldsymbol{\theta}^{(k)})\mathbf{x}), \end{aligned} \tag{11}$$

where $H(\boldsymbol{\theta})=\dfrac{\partial[A(\boldsymbol{\theta})\mathbf{x}]}{\partial\boldsymbol{\theta}}$, is an $MN\times P$ Jacobian matrix, when P parameters of desired source are unknown. As it is necessary to collect all sensor node observations to update the estimation, it is not a fully distributed algorithm. So, the fully distributed information-driven collaborative processing (IDCP) investigated in the next section.

3.1. Distributed Information_Driven Maximum Likelihood Estimation

It is proven that the update stage of the Gauss-Newton method can be divided in to a sequence of updates at each sensor node. By applying this idea, instead of transmitting the messages through all the sensor nodes, at each sensor node some information is used to update the estimation and determine which sensor in the neighbor of the current node is the best one regarding information amount (more details are in [6]).

The algorithm is initialized by sensor node i=1. It is assumed that an initial value of $\boldsymbol{\theta}$ is available at this sensor node, denoted by $\hat{\boldsymbol{\theta}}^{(0)}$.

At sensor node *i*, the following steps are performed:

- *Data receiving*: the sensor node *i* has been activated by the previous node and has received the transmitted required data.
- *New variable calculation*: Calculate the new matrix $\mathbf{H}_i(\hat{\boldsymbol{\theta}}^{(i-1)})$ as

$$H_i(\hat{\boldsymbol{\theta}}^{(i-1)}) = \left.\frac{\partial\left[A_i(\boldsymbol{\theta})\mathbf{x}\right]}{\partial\boldsymbol{\theta}^T}\right|_{\boldsymbol{\theta}=\hat{\boldsymbol{\theta}}^{(i-1)}}; \tag{12}$$

then, update the information utility matrix, Γ_i, as

$$\begin{aligned} \Gamma_i &= \Gamma_{i-1} \\ &\quad + H_i^T(\hat{\boldsymbol{\theta}}^{(i-1)})\Sigma_i^{-1}H_i(\hat{\boldsymbol{\theta}}^{(i-1)}). \end{aligned} \tag{13}$$

- *Estimation update*: Update and obtain the current estimate $\hat{\boldsymbol{\theta}}^{(i)}$ as

$$\begin{aligned} \hat{\boldsymbol{\theta}}^{(i)} &= \boldsymbol{\theta}^{(i-1)} \\ &\quad +\Gamma_i^{-1}H_i^T\left(\hat{\boldsymbol{\theta}}^{(i-1)}\right)\times\Sigma_i^{-1}\left[\mathbf{y}_i - A_i(\boldsymbol{\theta}^{(i-1)})\mathbf{x}\right]. \end{aligned} \tag{14}$$

- *Estimation quality test*: Test the quality of the updated parameter, $\hat{\boldsymbol{\theta}}^{(i)}$, according to some performance measures, e.g., the trace or determinant of the covariance matrix in each sensor-node. If the estimation of parameter is "good enough," the estimation process is terminated; otherwise, the algorithm continues with the following steps.
- *Sensor node selection*: Select a sensor node from its neighbor according to the certain information

utility criteria. The selected node should provide the information that has the most potential to decrease the estimation uncertainty and increase the performance.

In [6], the Cramer-Rao bound is chose as information utility criteria and the estimation performance for two reasons: 1)The CRB is the lower bound on the variance of any unbiased estimators; it equals the inverse of the Fisher information matrix (FIM). Hence, the measures are calculated directly from the FIM. 2) The calculation of the FIM as a performance measure is an intrinsic part of the algorithm, *i.e.*, it can be obtained without any extra computation and data transmission. In [6] the recursive equation for FIM is obtained. By denoting the matrix $F_i(\boldsymbol{\theta})$ in the sequence $\{F_i(\boldsymbol{\theta}), i=1,2,...\}$ as the FIM of the parameter $\boldsymbol{\theta}$ when collect the measurements from the first *i* sensor nodes, *i.e.*, $\mathbf{y} = \{\mathbf{y_1}, ..., \mathbf{y_i}\}$, and according to the definition of FIM [11,12],

$$\begin{aligned} F_i(\boldsymbol{\theta}) &= -E\left[\frac{\partial^2 \ln p(\mathrm{y};\boldsymbol{\theta})}{\partial\boldsymbol{\theta}\partial\boldsymbol{\theta}^T}\right] \\ &= F_{i-1}(\boldsymbol{\theta}) - E\left[\frac{\partial^2 \ln p(\mathbf{y}_i;\boldsymbol{\theta})}{\partial\boldsymbol{\theta}\partial\boldsymbol{\theta}^T}\right] \end{aligned} \tag{15}$$

The second term on the Right Hand Side (RHS) of (16) can be calculated as [6]:

$$\begin{aligned} &-E\left[\frac{\partial^2 \ln p(\mathbf{y}_i;\boldsymbol{\theta})}{\partial\boldsymbol{\theta}\partial\boldsymbol{\theta}^T}\right] \\ &= E\left[\frac{\partial \ln p(\mathrm{y}_i;\boldsymbol{\theta})}{\partial\boldsymbol{\theta}}\frac{\partial \ln p(\mathbf{y}_i;\boldsymbol{\theta})}{\partial\boldsymbol{\theta}^T}\right] \\ &= H_i^T(\boldsymbol{\theta})\Sigma_i^{-1}H_i(\boldsymbol{\theta}) \end{aligned} \tag{16}$$

Where $H_i(\boldsymbol{\theta}) = \left[\frac{\partial[A_i(\boldsymbol{\theta})\mathrm{x}]}{\partial\theta_1} \cdots \frac{\partial[A_i(\boldsymbol{\theta})\mathrm{x}]}{\partial\theta_P}\right]$.

Hence, the recursive equation for the FIM sequence is obtained as:

$$F_i(\boldsymbol{\theta}) = F_{i-1}(\boldsymbol{\theta}) + H_i^T(\boldsymbol{\theta})\Sigma_i^{-1}H_i(\boldsymbol{\theta}) \tag{17}$$

By comparing (17) and (13) it can be easily observed that the updating formulas for the matrix $F_i(\boldsymbol{\theta})$ and $\boldsymbol{\Gamma}_i$ are the same. This equivalence represents that using FIM as the performance measure will not increase the computation complexity and the required transmission.

Consequently, if the trace of the FIM matrix is used as the information measure, the information utility function for node selecting is:

$$I(l;\boldsymbol{\theta}) = Tr\left\{H_l^T(\boldsymbol{\theta})\Sigma_l^{-1}H_l(\boldsymbol{\theta})\right\}. \tag{18}$$

Then, the sensor node l that maximizes this information utility function is selected as the next sensor node *i*+1:

$$i+1 = \arg\max_{l\in S} I(l;\boldsymbol{\theta}) \tag{19}$$

- *Data transmission*: The current sensor node transmits the current estimate $\hat{\boldsymbol{\theta}}^{(i)}$ and $\boldsymbol{\Gamma}_i$ to the selected node and then the current node returns to sleeping status.

4. Modified Information-driven Collaborative Processing and Energy Efficient Algorithms

4.1. Modified Information-Driven Collaborative Processing

In wireless sensor networks, good collaboration among distributed sensor nodes can be very useful to improve estimation accuracy and keep the energy consumption in a reasonable level. In the following, we propose a new approach for estimation update motivated by the above distributed estimation, *i.e.*, IDCP. Here, we assume that *S* is the collection of the sensor nodes located in the neighbor area of the current sensor node i and $l \in S$;

$$S = \left\{l : \|\mathbf{r}_l - \mathbf{r}_i\| \le r_0\right\}, \tag{20}$$

Where $\mathbf{r}_i$, r_0 represent the location of the *i*th sensor node and the neighborhood radius (here sensing range) of the sensor nodes, respectively.

Through this approach, the sensor node *i* sends the update information, $\hat{\boldsymbol{\theta}}^{(i-1)}$, to all of its neighbors, then all these neighbors calculate the new private Jacobian matrix, $\mathbf{H}_l(\hat{\boldsymbol{\theta}}^{(i-1)})$, and information utility matrix, $\boldsymbol{\Gamma}_l$. Based on these calculated matrixes, each node obtains the $\hat{\boldsymbol{\theta}}^{(l)}$ as:

$$\begin{aligned} \hat{\boldsymbol{\theta}}_l^{(i)} = \boldsymbol{\theta} &+ \Gamma_l^{-1}H_l^T\left(\hat{\boldsymbol{\theta}}^{(i-1)}\right) \\ &\times \boldsymbol{\Sigma}_l^{-1}\left[\mathbf{y}_l - A_l(\hat{\boldsymbol{\theta}}^{(i-1)})\mathbf{x}\right]. \end{aligned} \tag{21}$$

Each node transmits $\hat{\boldsymbol{\theta}}_i^{(i)}$ to the *i*th node when this node also performs the step 2 and 3 of IDCP algorithm to obtain $\hat{\boldsymbol{\theta}}_i^{(i)}$. Consequently, the estimation is updated as follow:

$$\hat{\boldsymbol{\theta}}(i) = \frac{1}{L}\sum_{l=1}^{L}\boldsymbol{\theta}_l^{(i)}. \tag{22}$$

Note that in (22) the estimated value of current node, $\hat{\boldsymbol{\theta}}_i^{(i)}$, is included because the current node is considered in its neighborhood set. In our new algorithm, Modified

IDCP (MIDCP), for estimation update, the information utility of all neighbor sensor nodes are used instead of one node and it can be reasonable to expect that the accuracy of the algorithm increases. Generally, since every possible value of the information utility in a sensor node's neighborhood is used, the node can estimate the location of the source more accurately in comparison with IDCP. The sensor selection scheme is as same as one in IDCP. However, one of the most important challenges in designing new algorithms for wireless sensor networks is energy consumption for signal processing. Therefore, it is necessary to consider energy consumption for new algorithms.

4.2. Energy Efficient Algorithm

In this paper we consider Rough order of Magnitude (ROM) model [13]. Technically, the model provides the energy required for a node to reach another node in one hop. Given l bits of data, the model states that the energy to transmit the data to a distance of d meters is:

$$E_t = l\varepsilon_{elec} + l\varepsilon_{amp} d^4 \tag{23}$$

and the energy to receive the data is

$$E_r = l\varepsilon_{elec} \tag{24}$$

The energy per bit to run the electronics such as the filters is represented by ε_{elec}, and the energy to run the power amplifier is presented by ε_{amp}. The values for these parameters are ε_{elec} = 50 nJ/bit and ε_{amp} = 0:0013 pJ/bit/m4. The model assumes $1/d^4$ for propagation loss in the channel because of the complicated transmission medium. It is predicted that the node antenna will be low to ground, possibly hidden under the grass. Finally, we set the packet size to be l = 384 bits to encompass the information required to perform distributed localization along with the overhead for error correcting and security [13]. It is important to note that the energy consumption in sensor nodes for computational tasks is much less than energy consumption for receiving and transmitting. As a result, we do not consider the energy amount used for computational processes.

It can be seen from (22) that the energy consumption in each node is related to distance between destination node and receiver node with the power of four. Therefore, one of the most efficient ways to reduce the energy consumption can be decrease the distance between transmitter and receiver nodes. This fact motivates us to design a new algorithm in which the neighborhood radius of the nodes decreases gradually. This new algorithm is called Energy Efficient MIDCP (EFMIDCP). The flowchart of EFMIDCP is shown in **Figure 1**.

As **Figure 1** shows, at first, the selected node receives the required parameters, $\hat{\boldsymbol{\theta}}^{(i-1)}$ and $F_{i-1}\left(\hat{\boldsymbol{\theta}}^{(i-1)}\right)$ from the previous node which has already updated the estimation. In fact, the inversion matrix of FIM is Cramer-Rao Bound (CRB) and it is a lower bound for unbiased estimators like ML. As a result, by calculating determinant of CRB the covariance error of the ML estimation is obtained at each iteration.

According to **Figure 1,** the next step is to decide whether the neighborhood radius is required to decrease or not. The Information utility of nodes located near the desired target is very high; therefore, saturation state of information is more likely to occur, *i.e.*, change in amount of energy is trivial from one node to another. Consequently, it can be expected that the performance of estimation is not change dramatically by reducing the number of nodes in this area.

It can lead us to decreasing the neighborhood radius for update estimation to abate energy consumption. Based on the above definitions, we use predefined thresholds and their correspondence neighborhood radius to decrease this radius at each iteration. According to (22), by applying this algorithm, the energy consumption decreases due to elimination of transmitting the data to further distance.

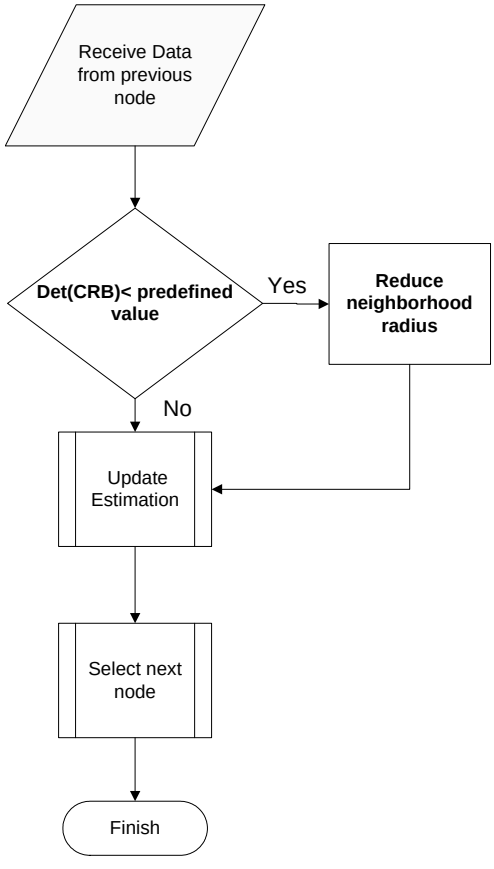


Figure 1. Flowchart of MIDCP algorithm.

5. New Method for Initial Target Location Estimation

All iterative algorithms, such as Gauss-Newton method, are sensitive to the initial point of estimation. For better conception, a sample function for the estimation error is shown in **Figure 2**. It can be seen that many sub-optimum points exist in error function. Thus, if the initial point is not estimated accurately, the algorithm requires much iteration to diverge; even in some cases the divergence is probable.

The method that we will propose in this section needs at least three sensor nodes. As we mentioned before, the distribution model for a diffusive source is:

$$c(\mathbf{r},t)=\frac{\mu}{4\pi k\left|\mathbf{r}-\mathbf{r}_0\right|}erf\left(\frac{\left|\mathbf{r}-\mathbf{r}_0\right|}{2\sqrt{k\left(t-t_0\right)}}\right) \tag{25}$$

Where $erf\left(x\right)=\frac{2}{\sqrt{\pi}}\int_0^x e^{-t^2}dt$ If the observation of one node can satisfy $t-t_0 \gg \left(\left|\mathbf{r}_i-\mathbf{r}_0\right|^2\right)/4k$, the *erf* function is approximately one and with renunciation of noise, the observation of this node is equal to:

$$y\left(\mathbf{r}_i,\infty\right)\approx\frac{\mu}{4k\pi\left|\mathbf{r}_i-\mathbf{r}_0\right|}+b \tag{26}$$

Therefore,

$$\begin{aligned}&\frac{\mathrm{y}\left(\mathbf{r}_i,\infty\right)-\mathrm{y}\left(\mathbf{r}_1,\infty\right)}{\mathrm{y}\left(\mathbf{r}_2,\infty\right)-\mathrm{y}\left(\mathbf{r}_1,\infty\right)}\\&\approx\frac{\left|\mathbf{r}_i-\mathbf{r}_0\right|^{-1}-\left|\mathbf{r}_1-\mathbf{r}_0\right|^{-1}}{\left|\mathbf{r}_2-\mathbf{r}_0\right|^{-1}-\left|\mathbf{r}_1-\mathbf{r}_0\right|^{-1}}\quad 3\le i\le n\end{aligned} \tag{27}$$

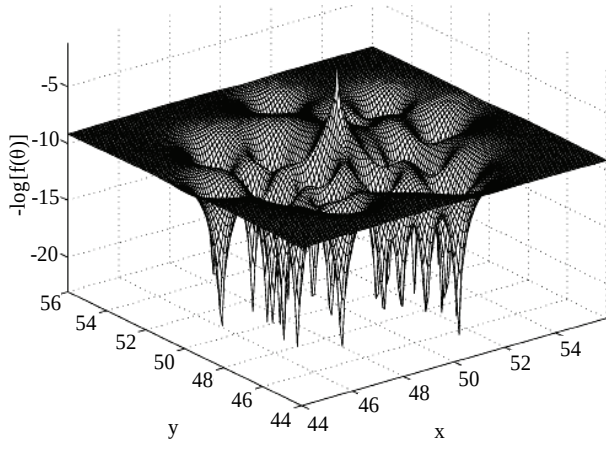


Figure 2. Logarithm of the error function of a distributed ML algorithm [14].

If n = 4, two unknown variable, $\mathbf{r}_0=\left(x_0,y_0\right)$, exist and they are obtained by using numerical methods to solve the set of equations and if n > 4 these are obtained by solving set of equations and then averaging over the results. In our proposed algorithm, we choose one node arbitrary and this node is considered as the central processor. It gathers all observation of its neighbor nodes and solves (27) to achieve the initial point for estimation. In addition, this node is the first node in our sequence of selected node.

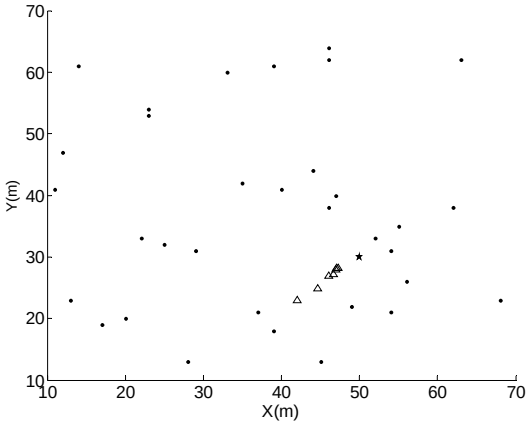
6. Numerical Examples

This section presents a simulation of a sensor network with M = 100 sensor nodes, distributed randomly in a 100? 00m field. At each sensor, the measurement of the diffusive source is generated according to (5). Here, source position $\mathbf{r}_0=(x_0,y_0)$ is the unknown parameter we need to estimate. We define a pair of neighbor sensor nodes whose distance is less than 20m, *i.e.*, the communication range (neighborhood radius) is 20 m. For the diffusion model, we consider the environment as homogenous semi-infinite with an impermeable boundary which can represent dispersion in air above the ground. We use a scenario of a stationary impulse source located at $\mathbf{r}_0=(50,30)$ meter. The bias term in (5) is $b=10^{-5}$ g/m^2, and the noise standard deviation is $\sigma=6\times10^{-6}$ g/m^2. We take 10 temporal samples at each sensor node with a sampling interval of 5 seconds. The other parameters μ, k, and t_I are taken to be 1 g/s, 20 m^2/s and 0 s, respectively**.**

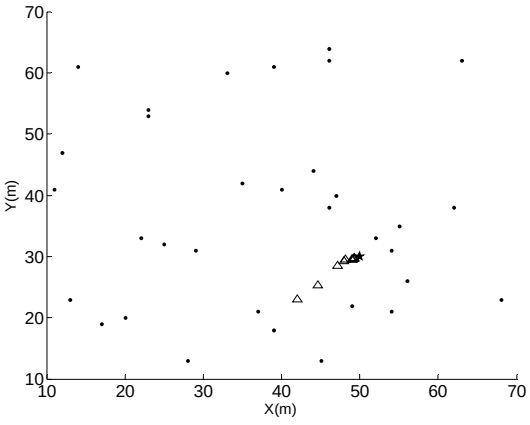
In **Figure 3,** the trajectory of target estimation for IDCP and MIDCP is shown. The trajectory is depicted with 10 iterations of both algorithms. This figure is the result of averaging over 200 independent experiments in which the location of sensor nodes and target are assumed constant but the observations and measurement noise of each sensor changes. In addition, the initial value for estimation is obtained based on our proposed algorithm in Section 5. In this figure, the "star" denotes the exact location of the desired target and "triangles" denote estimated location at each iteration. We observe that although the initial bias is relatively high, the trajectory nearly reach to the exact source location in MIDCP. This means the algorithm can converge accurately and nearly in comparison with IDCP.

For better conception, we compare bias estimation and covariance error of these two algorithms in the following. In **Figures 4 and 5** we compare the estimation bias $\sqrt{\left\|\theta-\hat{\theta}\right\|^2}$ of $\boldsymbol{\theta}$ and the logarithm of determinant of CRB for IDCP and MIDCP with respect to the number

of iterations. These figures are the result of averaging over 200 independent experiments in which the location of the sensor nodes and target are constant, but the observations and measurement noise of each sensor change.



(a)



(b)

Figure 3. Trajectory of target estimation in a square area with 100 sensor nodes randomly placed for a) IDCP and b) MIDCP.

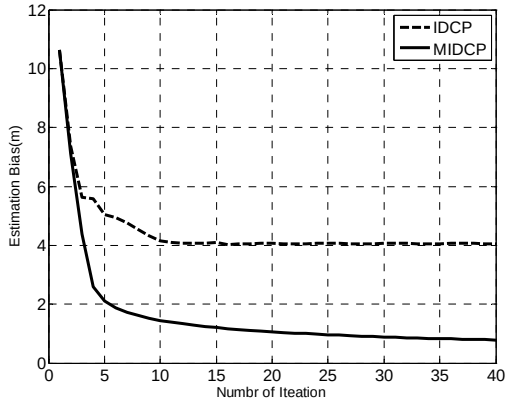


Figure 4. Estimation bias versus number of iterations for IDCP and MIDCP algorithms.

According to these two above figures, we find that the estimation performance of MIDCP is much better than IDCP and this means that using all information utilities of neighbor sensors can greatly improve the accuracy of the estimation.

Before we discuss the simulation results for EF-MIDCP, we provide the estimation bias and energy consumption of MIDCP with the different neighborhood radiuses. At first, we calculate the energy consumption for neighborhood radiuses equal to 5, 10, 15, 20 for MIDCP based on (23) and (24). The experiment conditions are similar to the previous one. The results are shown in **Table 1**.

As can be seen, by increasing the neighborhood radius (communication range) the energy consumption increase and the bias estimation decreases. Therefore, it is a tradeoff between the accuracy of the algorithm and the bias estimation. It motivates us to decrease the energy consumption without dramatic change in the accuracy of the algorithm; we decrease the neighborhood range with regard to the error covariance of MIDCP calculated at each iteration. (See Section 4.)

In the second example, we compare the performance of MIDCP with EFMIDCP. In **Figures 6 and 7** the estimation bias $\sqrt{\left\|\theta-\hat{\theta}\right\|^2}$ of $\boldsymbol{\theta}$ and logarithm of determinant of CRB with respect to the number of iterations are shown. These figures are the result of averaging over 200 independent experiments in which the location of sensor

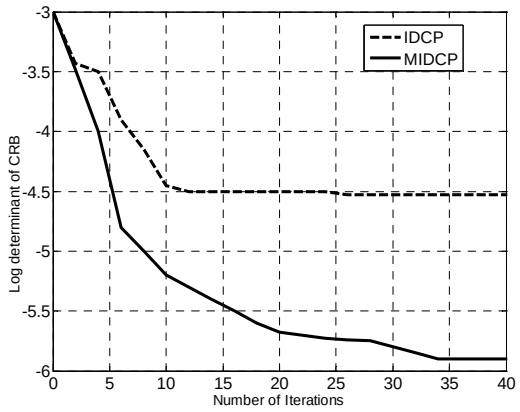


Figure 5. Log determinant of CRB versus number of iterations for IDCP and MIDCP algorithms.

Table 1. Energy consumption and estimation bias for MIDCP with different neighborhood radiuses.

Neighborhood radius	5	10	15	20
Bias Estimation(m)	5.25	0.85	0.88	0.67
Energy Used (μJ)	0.14	1.56	4.9	24.38

nodes and the target are assumed constant but the observations and measurement noise of each sensor change. Other conditions are as same as our first example. In this scenario, we change the neighborhood radius with the following pattern:

- If $\log(|CRB|) \geq -4.5$ then the neighborhood radius is set to 20.
- If $-5 \leq \log(|CRB|) \leq -4.5$ then the neighborhood radius is set to 15.
- If $\log(|CRB|) \leq -5$ then the neighborhood radius is set to 10.

According to these two figures, we find that the estimation performance of MIDCP is somewhat better than EFMIDCP. In addition, in this example we compare energy consumption of these tow algorithms and the result is shown in **Table 2**. As it shows, the energy consumption decreases dramatically and it can be observed that the EFMIDCP reduces the energy consumption by half while duplicates the accuracy of the algorithm.

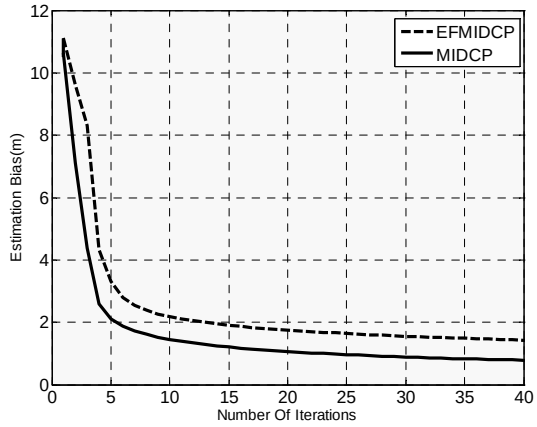


Figure 6. Estimation bias versus number of iterations for MIDCP and EFIDCP algorithms.

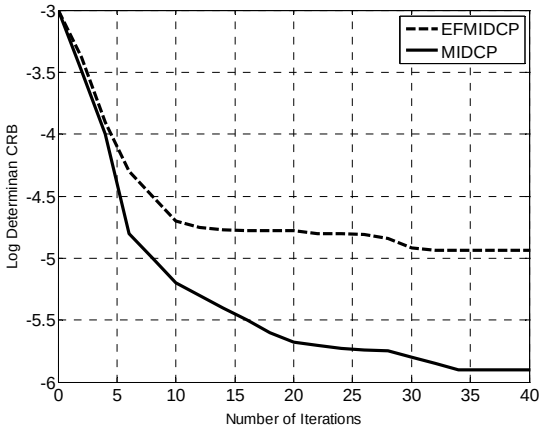


Figure 7. Log determinant of CRB versus number of iterations for MIDCP and EFMIDCP algorithms.

Table2. The energy and bias estimation for MIDCP and EFMIDCP.

	MIDCP	**EFMIDCP**
Bias Estimation (m)	0.67	1.45
Energy Used (μJ)	24.38	13.05

7. Conclusions

In this paper, we addressed the problem of developing an accurate and energy efficient distributed algorithm for a diffusive source location estimation in wireless sensor networks. At first, we defined the neighborhood radius that was equal to the communication range of the sensor nodes. Then, we used the information of every node located within the neighborhood circle. This method improved the accuracy of estimation. However, based on ROM model, we observed that the energy consumption in our new algorithm was somewhat high. As our primate goal was to reduce energy consumption and increase the accuracy, we proposed another algorithm in which the neighborhood range decreased gradually based on the covariance error of the estimation. In addition, we proposed a new algorithm for the initial point calculation which was necessary for iterative methods like Gauss-Newton method. Finally, we used numerical examples to compare the performance of our proposed methods and found that EFMIDCP was the most suitable algorithm to apply in wireless sensor networks when we considered both energy consumption and the accuracy of the estimation. In future work, a mathematical relationship between the neighborhood radius and the number of covariance error can be derived; therefore the neighborhood radius can be changed adaptively instead of changing it with regard to predefined value. In addition, the distributed implementation of other recursive methods such as Newton method can be derived.

8. Acknowledgements

This research has been supported by Iran Telecommunication Research Center, Tehran, Iran, which is appreciated.

5. References

[1] T. Zhao and A. Nehorai, "Detecting and Estimating Biochemical Dispersion of a Moving Source in a Semi-Infinite Medium," *IEEE Transactions on Signal Processing*, Vol. 54, No. 6, June 2006, pp. 2213-2225.

[2] X. Sheng and Y. H. Hu, "Maximum Likelihood Multiple-Source Localization Using Acoustic Energy Measurements with Wireless Sensor Networks," *IEEE Transac-*

tions on Signal Processing, Vol. 53, No. 1, January 2005, pp. 44-53.

[3] A. Pardo, S. Marco and J. Samitier, "Nonlinear Inverse Dynamic Models of Gas Sensing Systems Based on Chemical Sensor Arrays for Quantitative Measurements," *IEEE Transactions on Instrumentation and Measurement*, Vol. 47, No. 3, June 1998, pp. 644-651.

[4] H. Ishida, T. Nakamoto and T. Moriizumi, "Remote sensing and localization of gas/odor source and distribution using mobile sensing system," *Proceedings of the International Conference on Solid-State Sensors and Actuators*, Chicago, Vol. 1, June 1997, pp. 16-19.

[5] A. Nehorai, B. Porat and E. Paldi, "Detection and Localization of Vaporemitting Sources," *IEEE Transactions on Signal Processing*, Vol. 43, No. 1, January 1995, pp. 243-253.

[6] T. Zhao and A. Nehorai, "Information-Driven Distributed Maximum Likelihood Estimation Based on Gauss-Newton Method in Wireless Sensor Networks," *IEEE Transactions on Signal Processing*, Vol. 55, No. 9, September 2007, pp. 4669-4682.

[7] A. T. Ihler, J. W. Fisher, R. L. Moses and A. S. Willsky, "Nonparametric Belief Propagation for Self-Localization of Sensor Networks," *IEEE Journal on Selected Areas in Communications*, Vol. 23, No. 4, April 2005, pp. 809-819.

[8] J. L. Williams, J. W. Fisher and A. S. Willsky, "Optimization Approaches to Dynamic Routing of Measurements and Models in a Sensor Network Object Tracking Problem," *IEEE International Conference on Acoustics, Speech, and Signal Processing* (*ICASSP*), Philadelphia, Vol. 5, March 2005, pp. 1061-1064.

[9] J. Liu, J. Reich and F. Zhao, "Collaborative In-Network Processing for Target Tracking," *EURASIP Journal on Applied Signal Processing*, Vol. 2003, No. 4, 2003, pp. 378-391.

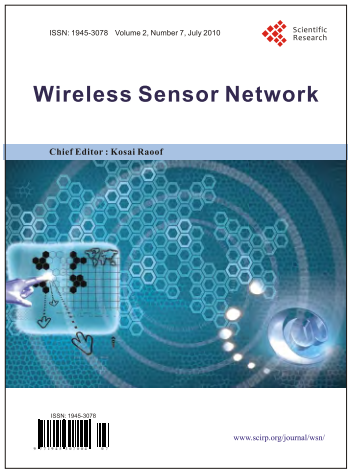
[10] M. G. Rabbat and R. D. Nowak, "Decentralized Source Localization and Tracking," *IEEE International Conference on Acoustics, Speech, and Signal Processing* (*ICASSP*), Montreal, May 2004, pp. 921-924.

[11] P. Stoica and A. Nehorai, "MUSIC, Maximum Likelihood, and Cramer-Rao Bound," *IEEE Transactions on Acoustics Speech and Signal Processing*, Vol. 37, No. 5, 1989, pp. 720-741.

[12] S. F. Yau and Y. Bresler, "A Compact Cramer-Rao Bound Expression for Parametric Estimation of Superimposed Signals," *IEEE Transactions on Signal Processing*, Vol. 40, No. 5, 1992, pp. 1226-1230.

[13] L. Kaplan, "Global Node Selection for Localization in a Distributed Sensor Network," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 42, No. 1, January 2006, pp. 113-135.

[14] D. Blatt and A. O. Hero, "Energy-Based Sensor Network Source Localization via Projection onto Convex Sets," *IEEE Transactions on Signal Processing*, Vol. 54, No. 9, September 2006, pp. 3614-3619.



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, 493-570
Published Online July 2010 in SciRes (http://www.SciRP.org/journal/wsn/).

TABLE OF CONTENTS

Volume 2 Number 7 **July 2010**

