International Journal on Advances in Networks and Services

IARIA

2009 vol. 2 nr. 4

The *International Journal on Advances in Networks and Services* is published by IARIA.
ISSN: 1942-2644
journals site: http://www.iariajournals.org
contact: petre@iaria.org

Sponsored by IARIA
www.iaria.org

International Journal on Advances in Networks and Services
Volume 2, Number 4, 2009

Internet and Web Services

- Thomas Michael Bohnert, SAP Research, Switzerland
- Serge Chaumette, LaBRI, University Bordeaux 1, France
- Dickson K.W. Chiu, Dickson Computer Systems, Hong Kong
- Matthias Ehmann, University of Bayreuth, Germany
- Christian Emig, University of Karlsruhe, Germany
- Geoffrey Fox, Indiana University, USA
- Mario Freire, University of Beira Interior, Portugal
- Thomas Y Kwok, IBM T.J. Watson Research Center, USA
- Zoubir Mammeri, IRIT – Toulouse, France
- Bertrand Mathieu, Orange-ftgroup, France
- Mihhail Matskin, NTNU, Norway
- Guadalupe Ortiz Bellot, University of Extremadura Spain
- Dumitru Roman, STI, Austria
- Monika Solanki, Imperial College London, UK
- Vladimir Stantchev, Berlin Institute of Technology, Germany
- Pierre F. Tiako, Langston University, USA
- Weiliang Zhao, Macquarie University, Australia

Wireless and Mobile Communications

- Habib M. Ammari, Hofstra University - Hempstead, USA
- Thomas Michael Bohnert, SAP Research, Switzerland
- David Boyle, University of Limerick, Ireland
- Xiang Gui, Massey University-Palmerston North, New Zealand
- Qilian Liang, University of Texas at Arlington, USA
- Yves Louet, SUPELEC, France
- David Lozano, Telefonica Investigacion y Desarrollo (R&D), Spain
- D. Manivannan (Mani), University of Kentucky - Lexington, USA
- Jyrki Penttinen, Nokia Siemens Networks - Madrid, Spain / Helsinki University of Technology, Finland
- Radu Stoleru, Texas A&M University, USA
- Jose Villalon, University of Castilla La Mancha, Spain
- Natalija Vlajic, York University, Canada
- Xinbing Wang, Shanghai Jiaotong University, China
- Qishi Wu, University of Memphis, USA
- Ossama Younis, Telcordia Technologies, USA

Sensors

- Saied Abedi, Fujitsu Laboratories of Europe LTD. (FLE)-Middlesex, UK
- Habib M. Ammari, Hofstra University, USA
- Steven Corroy, University of Aachen, Germany
- Zhen Liu, Nokia Research – Palo Alto, USA

- Winston KG Seah, Institute for Infocomm Research (Member of A*STAR), Singapore
- Peter Soreanu, Braude College of Engineering - Karmiel, Israel
- Masashi Sugano, Osaka Prefecture University, Japan
- Athanasios Vasilakos, University of Western Macedonia, Greece
- You-Chiun Wang, National Chiao-Tung University, Taiwan
- Hongyi Wu, University of Louisiana at Lafayette, USA
- Dongfang Yang, National Research Council Canada – London, Canada

Underwater Technologies

- Miguel Ardid Ramirez, Polytechnic University of Valencia, Spain
- Fernando Boronat, Integrated Management Coastal Research Institute, Spain
- Mari Carmen Domingo, Technical University of Catalonia - Barcelona, Spain
- Jens Martin Hovem, Norwegian University of Science and Technology, Norway

Energy Optimization

- Huei-Wen Ferng, National Taiwan University of Science and Technology - Taipei, Taiwan
- Qilian Liang, University of Texas at Arlington, USA
- Weifa Liang, Australian National University-Canberra, Australia
- Min Song, Old Dominion University, USA

Mesh Networks

- Habib M. Ammari, Hofstra University, USA
- Stefano Avallone, University of Napoli, Italy
- Mathilde Benveniste, Wireless Systems Research/En-aerion, USA
- Andreas J Kassler, Karlstad University, Sweden
- Ilker Korkmaz, Izmir University of Economics, Turkey

Centric Technologies

- Kong Cheng, Telcordia Research, USA
- Vitaly Klyuev, University of Aizu, Japan
- Arun Kumar, IBM, India
- Juong-Sik Lee, Nokia Research Center, USA
- Josef Noll, ConnectedLife@UNIK / UiO- Kjeller, Norway
- Willy Picard, The Poznan University of Economics, Poland
- Roman Y. Shtykh, Waseda University, Japan
- Weilian Su, Naval Postgraduate School - Monterey, USA

Multimedia

- Laszlo Boszormenyi, Klagenfurt University, Austria
- Dumitru Dan Burdescu, University of Craiova, Romania
- Noel Crespi, Institut TELECOM SudParis-Evry, France
- Mislav Grgic, University of Zagreb, Croatia

- Hermann Hellwagner, Klagenfurt University, Austria
- Polychronis Koutsakis, McMaster University, Canada
- Atsushi Koike, KDDI R&D Labs, Japan
- Chung-Sheng Li, IBM Thomas J. Watson Research Center, USA
- Parag S. Mogre, Technische Universitat Darmstadt, Germany
- Eric Pardede, La Trobe University, Australia
- Justin Zhan, Carnegie Mellon University, USA

Additional reviews by:

- Jorjeta Jetcheva, Carnegie Mellon, USA

International Journal on Advances in Networks and Services
Volume 2, Number 4, 2009

CONTENTS

MAC Protocols for Wireless Sensor Networks: Tackling the Problem of Unidirectional Links

Stephan Mank, Reinhardt Karnapke, Jörg Nolte
Distributed Systems/Operating Systems Group
Brandenburg University of Technology
Cottbus, Germany
{*smank, karnapke, jon*}*@informatik.tu-cottbus.de*

***Abstract*—Experiments have shown that unidirectional links are quite common in wireless sensor networks. Still, many MAC protocols ignore their existence, even though they have a tremendous impact on the performance of both TDMA- and contention based protocols. In contention based protocols the medium may be assumed free when it is indeed busy. In TDMA based protocols two neighboring nodes might get assigned the same slot even though there is an unidirectional link between them. In this paper we discuss the influence of unidirectional links on communication protocols in wireless sensor networks, focusing on MAC protocols. We also present two protocols that do not only eliminate the negative side effects of unidirectional links, but use them for message transmission as well.**

***Keywords*-Wireless Sensor Networks; MAC Protocols; Unidirectional Links**

I. Introduction

Wireless sensor networks are collections of small sensing and computation units that can cooperate with each other using over the air communication. Since these networks shall be deployed on a large scale (i.e. hundreds of nodes), the overall cost often dictates the usage of cheap radio transceivers. Many of these transceivers do not only lack hardware support for medium access control, their huge number also makes it near to impossible to calibrate all of them exactly the same, resulting in many differences in antennae characteristics. Due to these differences, which are also enhanced by the difference in orientation of the deployed nodes, a lot of unidirectional links (node A can send to node B but not vice versa) are introduced into the sensor network right from the beginning. Differences in height of position are also an influencing factor. After the sensor network is started, dynamic effects like atmospheric changes, animals walking by or people using electrical devices lead to an often changing radio neighborhood. These changes can be a complete breakage of links, or the transformation from a unidirectional to a bidirectional one and vice versa. Sometimes a unidirectional link changes its direction.

Most of todays sensor networks are meant to deliver the gathered data in one form or another to a sink for evaluation. But this requires multihop transmissions along changing routes. Finding a suitable route is the task of routing protocols, the MAC protocol only needs to supply one hop communication. Unidirectional links are a common phenomenon on both protocol layers - most routing protocols try to eliminate the negative effect they have on their routing choices, only some of them try to utilize them. MAC protocols face a harder problem, as the effect of a unidirectional link may not only be a wrong choice, but a lot of collisions leading to a bad channel utilization and packet loss.

In this paper we present the influences of unidirectional links on both protocol layers, and describe a way of increasing network connectivity, reliability and lifetime by using the unidirectional links in addition to the bidirectional ones in MLMAC-UL (TDMA-based) and ECTS-MAC (contention based), both presented first at Sensorcomm 2009 [1]. The effectiveness for both different approaches is shown in simulations as well as in experiments with TMote Sky sensor nodes.

This paper is structured as follows: Section II takes a closer look at unidirectional links, their occurrence in wireless sensor networks and their impact on routing and MAC protocols. Section III describes related work, while sections IV and V describe the protocols MLMAC-UL and ECTS-MAC respectively. In section VI our two protocols are evaluated, both with simulations and experiments on real sensor network hardware. We finish with conclusion and future work in section VII.

II. The Nature of Unidirectional Links

In theory a unidirectional link is defined quite simple. A link from node A to node B is unidirectional, if Node B can receive messages from A, but not vise versa. In practice, it is fairly hard to establish such criteria. It is not possible to monitor the status of all links globally. You can only measure the status of a link at a certain time. Moreover, only one direction of the link can be measured because transceivers can not transmit and receive at the same time. Worse still, links change over time. Due to e.g. atmospheric changes or someone walking into the area, a link that seems to be bidirectional at one moment can become unidirectional at any time.

The authors of [2] describe an experiment they conducted in the Lüneburger Heide. The original aim was to evaluate a routing protocol, which is not characterized further in the paper. Rather, the observations they made concerning the properties of the wireless medium are described, focusing on the frequency of changes and the poor stability of links. These experiments were conducted using 24 Scatterweb ESB [3] sensor nodes, which were affixed to trees, poles etc, and left alone for two weeks after program start. One of the duties of the network was the documentation of the logical topology (radio neighborhood of nodes), which was evaluated by building a new routing tree every hour, e.g. for use in a sense-and-send application. The neighborhood was evaluated using the Wireless Neighborhood Exploration protocol (WNX) [2], which can detect unidirectional and bidirectional links. Once this was done, all unidirectional links were discarded and only the bidirectional ones were used to build the routing tree. Figure 1a shows one complete

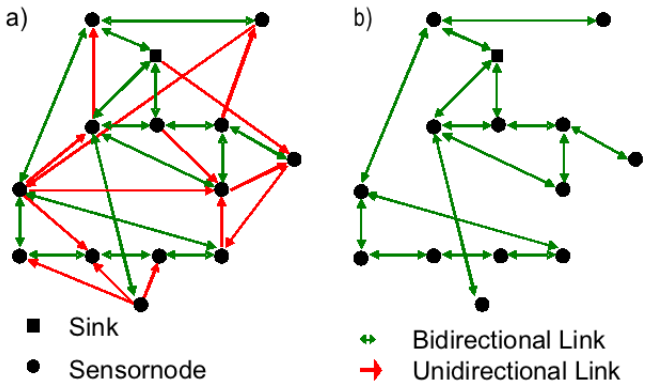


Figure 1. A Communication Graph from [2] (Presentation) [4]

communication graph obtained by WNX, while figure 1b shows the same graph without unidirectional links, where a lot of redundant paths have been lost by the elimination. In fact, one quarter of the nodes are only connected to the rest of the network by a single link when unidirectional links are removed. If this single link breaks, the nodes become separated, even though there are still routes available. Thus, the removal of unidirectional links increases the probability of network separation severely.

The authors of [5] hold a similar view. They evaluate the three kinds of links (asymmetric, unidirectional, bidirectional) using protocols like ETX (Expected Transmission Count) [6]. These protocols search for reliable links, but most focus on bidirectional ones. This leads to the fact that a link with a reliability of 50% in both directions is chosen above one with 100% from node A to node B and 0% from B to A. If data needs to be transmitted only from A to B without need for acknowledgment, this choice is obviously wrong. To prevent this wrong choice, the authors of [5] propose a protocol called ETF (Expected Number of Transmissions over Forward Links), which is able to use unidirectional links. They also show that the reach of reliable unidirectional links is greater that that of reliable bidirectional links. In experiments with XSM motes [5] 7 times 7 nodes were placed in a square, with a distance of about 1 meter between nodes. In four sets of experiments at different times of day each node sent 100 messages at three different power levels. Then the packet reception rate was recorded, which is defined for a node A as the number of packets A received from a node B divided by the number of messages sent (100). Then the packet reception rates of nodes A and B are compared. If the difference is less than 10%, the link is considered bidirectional. If it is more then 90% the link is considered unidirectional. The XSM nodes offer 9 different transmission strengths, of which three were evaluated: the lowest, the highest and the third in between. Table I shows the results of the experiments.

Table I
LINK QUALITY VERSUS TRANSMISSION STRENGTH

PRR	less than 10%	10-90%	more than 90%	links
power level 1	50%	43%	7%	500
power level 3	65%	22%	13%	1038
power level 9	88%	6%	6%	1135

The results show that even when using the maximum transmission strength 12% of the links would have been discarded by ETX (Expected Transmission Count) [6] and similar link quality evaluation protocols that focus only on bidirectional links. As the lifetime is one of the major optimization goals in a sensor network and receiving/transmitting consumes a lot of energy, it is rather uncommon to have all nodes constantly transmit using the highest transmission strength. In fact, current research projects like e.g. [7] try to minimize power consumption by adjusting the transmission strength depending on the required reach and reliability.

The observations of [5] are concluded in three points:

1) Wireless links are often asymmetric, especially if transmission power is low
2) Dense networks produce more asymmetric links then sparse ones
3) Symmetric links only bridge short distances, while asymmetric and especially unidirectional ones have a much longer reach. A conclusion drawn from this fact is that the usage of unidirectional links in a routing protocol can increase the efficiency of a routing protocol considering energy and/or latency.

A sensor network which monitors water pumps within wells is described in [8]. The sensors were used to monitor the water level, the amount of water taken and the saltiness of the water in a number of wells which were widely

distributed. The necessity for this sensor network arose because the pumps were close to shore and a rise in saltiness was endangering the quality of the water. The average distance between wells was 850 meters and the range of transmission was about 1500 meters. Communication was realized using 802.11 WLAN hardware both for the nodes as well as for the gateway. For data transmission between nodes Surge_Reliable [9] was used, which makes routing decisions based on the link quality between nodes.

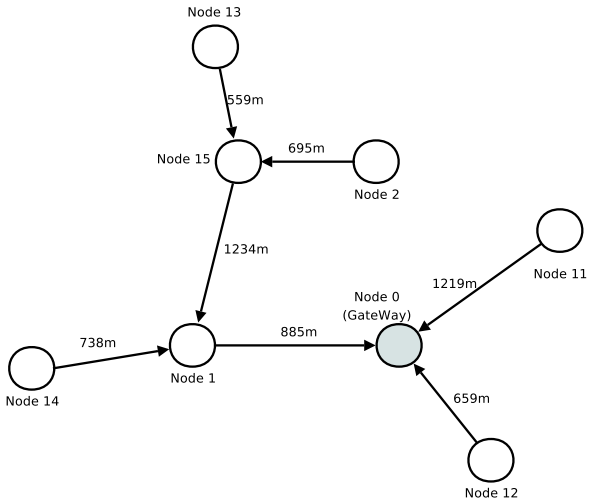


Figure 2. A Communication Graph from [8] that Follows the Theory

During the experiments the authors observed, that the (logical) topology of the network changed dynamically, even though all nodes were stationary. The authors claim that these changes were probably due to antenna size and changes in temperature and air moisture. In this context it is important to remember that the distance of nodes was far below the range of the transmitters (about 50%). While about 70% of the routing trees observed followed the theory (figure 2), there were a lot of strange exceptions. In one case the average distance between connected nodes even rose to 1135 meters, as nodes that should have been able to communicate directly with the gateway were connected to nodes on the far side instead. In one of these routing trees (figure 3), a single node had to take care of all communication with the gateway, even nodes that were on the other side were using it as next hop. The reason for this is that Surge_Reliable chooses the nodes with the best link quality, but only considers bidirectional links. If unidirectional links could have been used, the results could have been quite different.

VigilNet, a military sensor network for terrain surveillance is described in [10]. This project aims at the detection of moving vehicles using magnetic sensors attached to Mica2 sensor nodes. The transport of messages from the nodes to the sink was realized using a diffusion based algorithm, similar to Directed Diffusion [11], which produced a routing

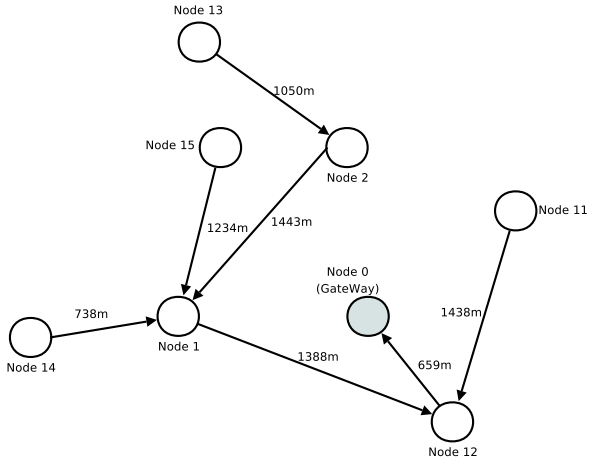


Figure 3. A Communication Graph from [8] with Long and Strange Links

tree with root at the sink. To eliminate unidirectional links, a protocol called Link Symmetry Detection was developed. Each node periodically transmitted the list of its neighbors. A node that received such a neighbor list checked the list to determine if it was mentioned. If it was not, the link was an incoming unidirectional one. When building the routing tree after deployment, the transmission power of all nodes was halved. Now all nodes determined their parent node from the neighbor lists received with this half strength. At the end of this setup phase, all nodes switched to full transmission power. The intention behind this scheme was to ensure that the connection to the father node would not break. During the experiments, the authors noted that asymmetric links were far more common than expected. They put this fact down to differences in hardware, as the transceivers were not calibrated before the experiment. Another interesting effect seen in these experiments is that only about 2/3 of all nodes were able to communicate directly with the sink, because only bidirectional links were used.

III. Related Work

The problem of unidirectional links has been recognized before, and protocols have been developed which can use them.

The Multicast MAC protocol (MMP) [12] does not directly address the problem of unidirectional links, but it offers an easy way to realize a multicast communication, which can easily be increased to broadcast. BMMM [13] and Maclayer Multicast [14] follow a similar approach. MMP is an extension of the IEEE 802.11 MAC in DCF mode. The Request To Send (RTS) message of MMP contains the addresses of all nodes that should receive the multicast message. When a node receives this RTS, it waits a certain time, correlating to its position in the RTS, and sends a CTS. When the slots for all CTS messages have passed and the sender of the RTS has received at least one CTS, it begins

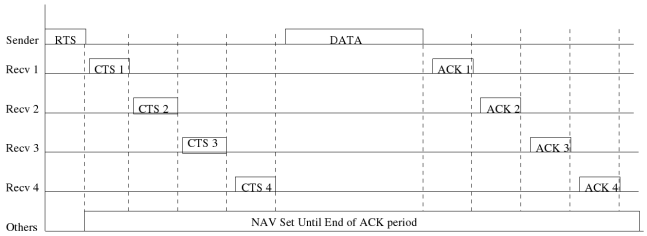


Figure 4. Message Propagation in MMP [12]

transmission of the data packet. After the transmissions, the acknowledgment messages are send by all of the receivers in the same order as the CTS messages (figure 4). While MMP needs to wait a time corresponding to the number of nodes addressed in the RTS message before sending data packets, the proposed ECTS-MAC waits only the time needed for a single ECTS message, thus providing much better scalability. Also, the size of the RTS is reduced drastically in ECTS-MAC, because the list of receivers is omitted (see section V).

AMAC [15] is built on top of the Sub Routing Layer (SRL) project [16], which is used to detect unidirectional links. When SRL is used with a routing protocol, it provides the abstraction of a network with only bidirectional links. To do this, it must identify unidirectional links, and find a suitable reverse route leading through multiple nodes. This is done using a reverse distributed Bellman-Ford algorithm. SRL also monitors the network for link changes. AMAC uses the information from SRL to make unidirectional links usable on the MAC layer. Four new types of messages are introduced to make communication over unidirectional links possible by forwarding protocol messages through neighboring nodes. AMAC uses a complex formula to identify the right nodes to forward all four types of messages, while the transmission of ECTS-messages in ECTS-MAC is done probabilistic. It defines 4 new messages: XRTS (Extended RTS), XCTS (Extended CTS), TCTS (Tunneled CTS) and TACK (Tunneled ACK). XRTS and XCTS are used to inform nodes about the communication that could normally not receive RTS and CTS, but which may still disturb the transmission because of their long communication range. The TCTS is sent by the destination of an RTS message if it was received over an unidirectional link. In this case direct sending of a CTS is not possible, therefore the TCTS must be forwarded by a neighboring node that can communicate with both participants of the communication (tunneled). Once the communication is complete, the destination sends a TACK message which is again tunneled for the same reason.

Another extension to IEEE 802.11 is BW_RES [17]. It is based on the principle of forwarding CTS packets to all nodes that may disturb the planned communication. To determine how far a BW_RES message must be forwarded, the transmission strengths of all nodes must be known. The lowest one equals one unit, the highest one N units. The authors show that a CTS message needs to be retransmitted 2N-1 times to ensure that it is heard at least N units distant. A node that receives a CTS message waits between 0 and 6 SIFS before transmitting the BW_RES packet to prevent collisions (figure 5). While this approach ensures that data communication in the presence of unidirectional links is possible, it delays the transmission and increases the network load proportional to the maximum difference in transmission strengths of nodes. In comparison, the network load produced by ECTS-MAC is rather low, depending on the chosen probability.

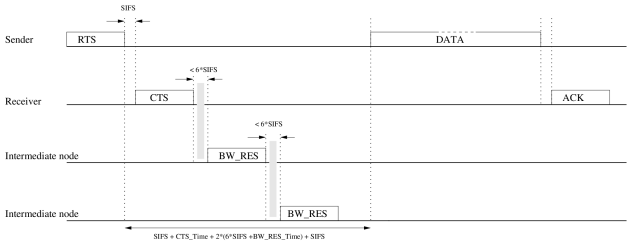


Figure 5. Message Propagation in BW_RES [17]

PANAMA (Pair wise Link Activation and Node Activation Multiple Access) [18] consists of two different algorithms. PAMA-UN (Pair wise link Activation Multiple Access Unidirectional Networks) is intended for unicast communication, while NAMA-UN (Node Activation Multiple Access for Unidirectional Networks) supplies broadcast communication. PANAMA is based on CDMA (Code Division Multiple Access) and uses DSSS (Direct Sequence Spread Spectrum). Also, Time is divided into slots. In each slot, nodes with orthogonal spread codes can transmit simultaneously. Codes are reassigned every slot, nodes compete for the codes by comparing their priority. The node with the highest priority has won the medium and all its neighbors configure their radio modules to use its spread code. The link characteristic (bidirectional or unidirectional) is a part of the bandwidth value which is featured in the computation of the priority. The main difference between NAMA-UN and PAMA-UN is the way priorities are computed. In NAMA-UN, the priority depends on the sending node, whereas in PAMA-UN it is calculated using all incoming links of both nodes participating in the communication. The most complex part of PANAMA is the calculation of priorities. Each node needs to know the exact priorities of all its neighbors at any time. It is 0 If the bandwidth from the sender to the receiver is 0 (unidirectional link from this node to its neighbor). A node wins the contention if its priority is higher than that of all its neighbors and there is no upstream-only-neighbor (neighbor with a unidirectional link to this node) that uses the same spread code. The priority of all neighbors

k in slot t is calculated as follows: $p_k^t = \sqrt[bw_k]{Rand(k+t)}$ where bw_k is the bandwidth of node k. $Rand$ is a random function which delivers a number between 0 and 1. $p_k = 0$ if $bw_k = 0$ In PAMA-UN the computation of the priority depends on all incoming links of both participating nodes x,y,: $p_{(x,y)}^t = \sqrt[bw_{(x,y)}]{Rand(x+y+t)}$. Both protocols, PAMA-UN and NAMA-UN depend on knowledge about the 2-hop neighbors of a node. To determine this, a neighborhood protocol is used, which transmits updates about the neighborhood of a node regularly. Each node can compute its 2-hop neighborhood by combining these messages from all its 1-hop neighbors. The update messages can contain information about multiple links. This information contains the ID of the neighbors, the status of the link (bidirectional or unidirectional), the type of change (add or delete a link/neighbor) and the current bandwidth. Depending on the rate of mobility the interval at which these messages are sent can be adjusted.

IV. MLMAC-UL

In previous work we introduced MLMAC [19], [20], a TDMA based MAC protocol for mobile wireless sensor networks. MLMAC divides time into frames, which are in turn divided into slots. Each node may use its own slot to transmit data to its neighbors, a slot reappears each frame. Nodes which have a common neighbor must have different slots to prevent collisions. For static networks it is fairly easy to find a schedule for all nodes that fulfills this property, for mobile nodes it is much harder. MLMAC uses an adaptive approach to enable each node in the sensor network to allocate a slot. In this approach there is no predefined starter node as in LMAC [21], rather the synchronization of nodes is started by the node that wants to transmit something first.

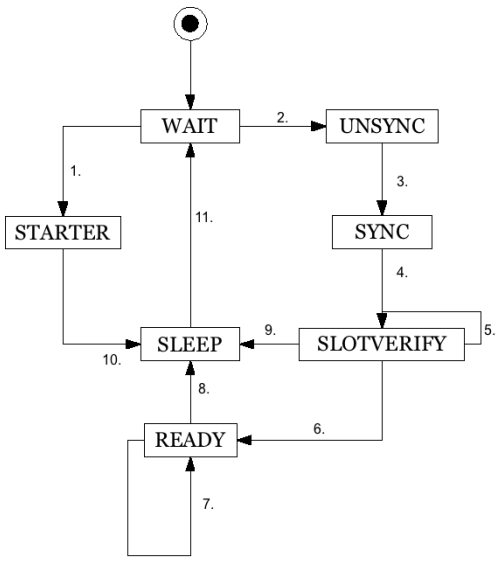


Figure 6. The Finite State Machine used in MLMAC [19]

In MLMAC a node may have one of 7 different states, and transitions from one to the other under certain conditions. The complete state-machine can be seen in figure 6. When nodes are first activated, MLMAC starts in the `WAIT`-state.

1) This node wants to transmit a message. It starts a global synchronization.
2) A control message from a node in `STARTER`- or `READY`-state was received. Synchronize local time.
3) After listening for one frame, choose a slot.
4) This nodes slot is active. Start transmitting a control message every frame in this slot
5) A collision seems to have occurred but the control message was received on a unidirectional link.
6) No collision occurred in the last frame.
7) A collision seems to have occurred but the control message was received on a unidirectional link.
8) A collision occurred and the link to the sender is bidirectional. Delete slot information.
9) A collision occurred and the link to the sender is bidirectional. Delete slot information.
10) A control message with a different, older synchronization was received. Remove all slot information.
11) After waiting for a certain time, return to the beginning and start again.

Most important for this work are the `ready`-state and the transition to the `sleep` state.

A node that has reached the `ready`-state is in a stable state, as long as no error occurs. If a collision occurs, the link from the sender is checked. If it is bidirectional, the node transitions into the `sleep`-state, because that is obviously wrong and a new slot has to be chosen. If the link is unidirectional, the node remains in the `ready`-state. The determination, whether a link is unidirectional or bidirectional is realized with a simple counter. Whenever transmissions are expected but not received, this counter is changed. After a predetermined number of missed messages, the link is considered unidirectional. This method has proven to be too ineffective for our purpose.

In this section we introduce the changes we made to MLMAC, to stop only detecting unidirectional links and ignore collisions that occurred because of them. Rather, MLMAC-UL uses a neighborhood discovery protocol to determine neighbors that can be used to inform the originator of a unidirectional link (the node that can be heard by the other one) about the link and make it usable to forward messages.

The first addition is an independent neighborhood discovery protocol, which is similar to the ones used in AMAC [15] and PANAMA [18] (see Section III). It transmits the neighborhood table of a node periodically but seldom. In the case of changes, only small update messages are sent. The periodic sending of tables is used to remove any errors resulting from loss of update packets.

Another change in MLMAC-UL is the fact that nodes can give up their slots. If a node has transmitted only status messages for a certain time (e.g., 6 frames) it will inform its neighbors that it is giving up the slot and that it may be used by another node. This is done by altering the status message a node transmits at the beginning of its slot. Moreover, a node may not only hold one slot in MLMAC-UL. Rather, each node can use as many slots as it needs by claiming any unused ones, when it has to transmit lots of data. Once the send queue is emptied, it can give the additional slots up one after the other. For this to be effective it is useful to define a larger frame size from the beginning, so that there are always enough free slots available (figure 7). This ability to hold more slots was introduced to reduce the delay and make MLMAC a better competitor against contention based protocols.

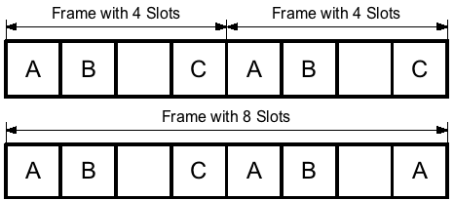


Figure 7. Using different Frame Sizes

Each node maintains a list of all its neighbors. Three entries define this list: The link quality, the unidirectionality status and the compressed neighborhood information from that neighbor. The link quality can be good (more than 90%reception rate), medium (between 30% and 90 % reception rate) or bad (less than 30% reception rate). The unidirectionality status can be either bidirectional, unidirectional_sender or unidirectional_receiver. The compressed neighborhood list is maintained by the neighborhood discovery protocol and used to identify the 2-hop-neighborhood of the current node.

The state machine of MLMAC-UL can be seen in figure 8. The arrows in the figure represent the transitions between states and are described in the following.

1) When a node needs to acquire its first slot it switches into the state `UNSYNC`.
2) The node was in state `UNSYNC` for one frame. It chooses a slot and transitions into the `SYNC`-state. If no slot was empty, the node stays in its current state for another frame.
3) When its chosen slot arrives, the node changes to state `SLOTVERIFY`.
4) The node sends in its slots. After one frame, it reaches the `READY`- state.
5) If a negative acknowledgment for the last slot was received, the slot is deleted and the node changes to state `SLEEP`.

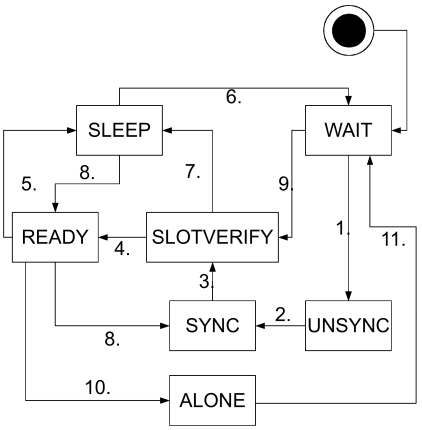


Figure 8. The State Machine of MLMAC-UL

6) The node returns to the `WAIT`-state after a random amount of time.
7) Same as 5.
8) There is data to be transmitted and no neighboring node is transmitting. The node chooses a slot and an identification for the synchronization. After waiting for a random time it transmits the data and switches to `READY`.
9) If this node did not communicate before or it had previously given up one slot a new slot is acquired and the node changes into the state `SLOTVERIFY`.
10) No messages from neighbors were received for 5 frames even though this node is transmitting. This means that this node is either completely isolated, or has only unidirectional links to others, but no incoming link from any of them. This node switches to the `ALONE`-state and does not try to transmit anymore, even when data is available.
11) A message from a neighbor was received, which means that this node is no longer alone or a certain number of frames (e.g., 200) have passed. The node switches to `WAIT` and starts again.

V. ECTS-MAC

ECTS-MAC (Extended Clear To Send MAC) is a contention based protocol for sparse networks with rare communication. It is similar to BW_RES [17] (see Section III), because it also tries to forward the CTS message to reduce the probability of collision. Unlike BW_RES, it does not calculate distances and power levels. Also, all ECTS messages are sent at the same time, whereas all BW_RES messages are sent one after another. This leads to more collisions of ECTS messages, but saves a lot of time. When a node receives a CTS message it forwards it with a certain probability (figure 9). Experiments have shown that 50% seems to be the optimal value for sparse networks. If the probability is less, the ECTS message is

not received by enough neighbors. If it is higher, the ECTS packets collide more often. These collisions are also the reason why the ECTS-MAC should only be used in sparse networks, as the ECTS packets would increase the network load in a dense network too much. To a certain extend, this effect is alleviated by reducing the probability of sending, but this also leads to more nodes that do not receive the ECTS message. The ECTS-MAC uses the neighborhood discovery protocol described in the previous section to detect unidirectional links. This is necessary to enable transmitting via a unidirectional link, because acknowledgments need to be forwarded to the sender using a second node.

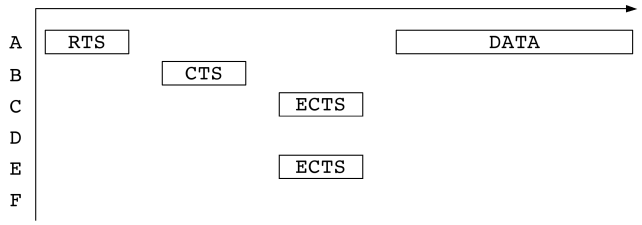


Figure 9. Propagation of ECTS Messages

VI. Evaluation

To measure the performance of MLMAC-UL and ECTS-MAC, we evaluated them against two other protocols: The original MLMAC and a modified version of MMP (Multicast Mac Protocol) [12] (see Section III). As its name suggests, MMP was designed for multicast, not for broadcast. We changed its behavior to enable broadcast transmissions, and to enable it to use unidirectional links. For this, we once again used the neighborhood discovery protocol described above. We call the resulting protocol NMAC (Neighborhood MAC). The functionality of NMAC is depicted in figure 10. Because of the neighborhood discovery protocol, the node knows how many neighbors it has and addresses them all in the RTS packet. When a node receives a RTS message it waits for a time corresponding to its position in the RTS before transmitting a CTS. If it has received at least one CTS, the sender of the RTS transmits the data package after the time for all CTS messages has passed.

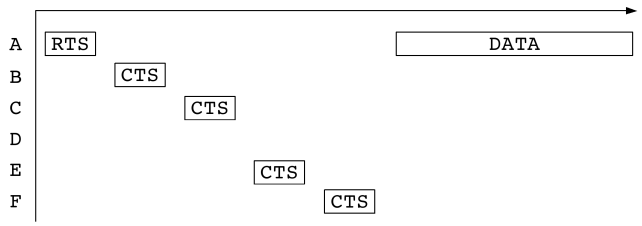


Figure 10. Message Propagation in NMAC

For our evaluations we used the discrete event simulator OMNeT++ [22] as well as real sensornet hardware. In all simulations the nodes transmitted with 19.2 KBit per second and a transmission strength of 10 milliwatt. For our real experiments we used TMote Sky sensor nodes from MoteIV corporation. These feature a MSP430 microcontroller with a frequency of 8MHz, 10 kB of Ram and 48 kB of flash. The radio module is IEEE 802.15.4 compatible and transmits 250kB/s, we configured the transmission strength to -25dBm to enable a multi hop scenario.

A. Single Hop Scenario Simulation

In this scenario the application behavior for a direct one-hop-neighborhood was simulated. The application tried to send as fast as possible. It generated a packet with 110 bytes data every 20 milliseconds, up to a total of 500 packets. In this simulation, all 4 protocols achieved a packet reception rate of nearly 100%. Figure 11 shows the amount of application data transmitted by each protocol. The figure shows that for 2-4 nodes the contention based protocols are able to transmit more data than the TDMA protocols. When more nodes are used, MLMAC-UL can gain an advantage because of the usage of multiple slots per node. As the number of slots was not changed for the MLMAC, it always delivers the same amount of data.

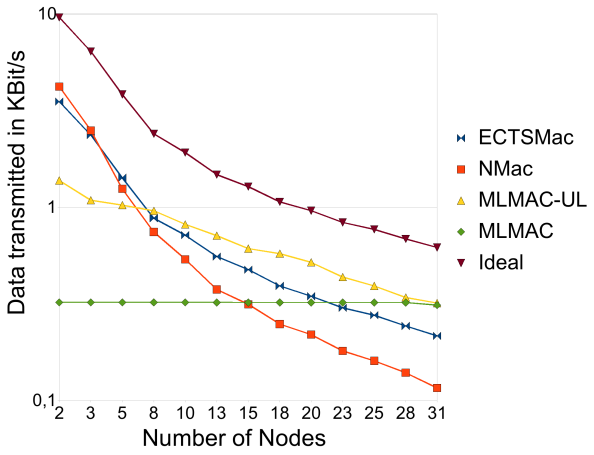


Figure 11. Data Transmitted 1-Hop Scenario

B. High Load Scenario Simulation

In this scenario a rectangle of 6 time 8 nodes was simulated. The application tried to send as fast as possible. It generated a packet with 110 bytes data every 20 milliseconds, up to a total of 500 packets. The node in the upper left corner started transmitting, each other node began transmitting its 500 packets after it had received the first packet. From a certain time on, all nodes want to transmit at nearly the same time, thus leading to a high

network load. We evaluated the number of packets that were transmitted flawlessly against the number of nodes in the 2-hop-neighborhood.

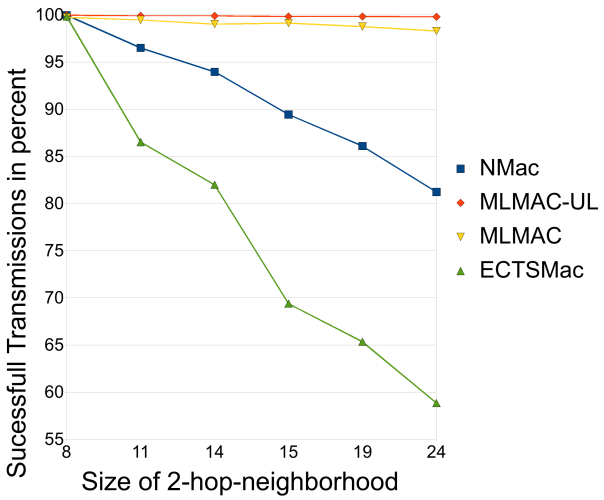


Figure 12. Delivery Ratio Rectangle Scenario

Figure 12 shows the percentage of successfully delivered packets for all 4 evaluated MAC protocols. As expected, the two TDMA based protocols were much better suited for this scenario then the contention based ones, which produced too many collisions.

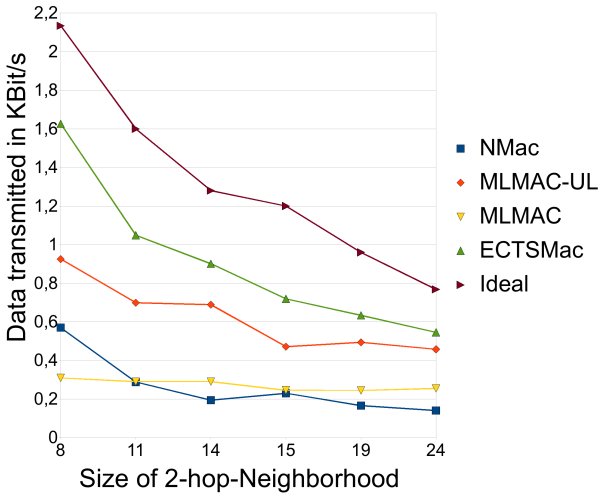


Figure 13. Data Transmission

Figure 13 shows the average amount of data each node was able to transmit for all 4 MAC protocols and the theoretical maximum. As can be seen, the ECTS-MAC is able to transmit most application data, followed by MLMAC-UL, MLMAC and NMAC. It is important to keep in mind here that this in the amount of application data transmitted, not received. If you correlate the bytes transmitted to the delivery ratio of the protocols, the performance of the ECTS-MAC drops considerably. The original MLMAC suffers from the fact that nodes may transmit only each frame, whereas MLMAC-UL allows each node to use multiple slots.

Another evaluation using the 6 times 8 nodes rectangle was used to determine the protocols' ability to deal with unidirectional links. To do this, we varied the rate of these from 0 to 70% in steps of 10. Figure 14 shows that MLMAC and MLMAC-UL can cope with the unidirectional links much better than ECTS-MAC and NMAC. Please note that these results were achieved using the neighborhood discovery protocol described above. If it is disabled, the performance of MLMAC-UL drops considerable, because it can no longer detect the unidirectional links and slots are given up too often. For the other protocols the impact is neglectable.

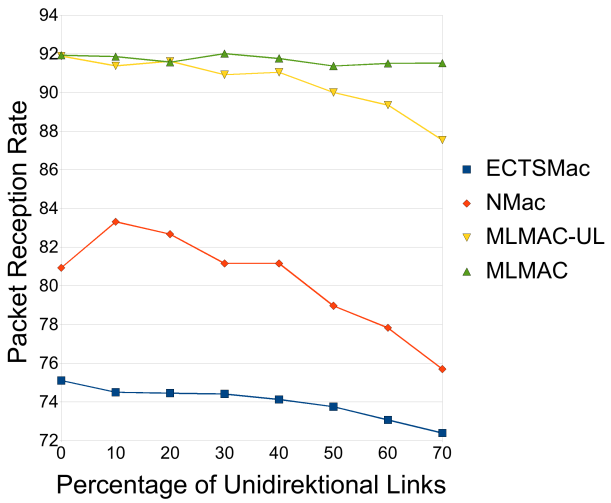


Figure 14. Variation of Unidirectional Links

C. Mobility Simulation

In this scenario the 4 protocols were evaluated using mobility with different speeds, random starting points and random destinations. The application sent packets of 110 Byte every second. This leads once again to a high network load. Figure 15 shows the number of received packets for

each protocol for the different speeds. Once again, MLMAC and MLMAC-UL provide the best results, with ECTS-MAC performing only a little worse. The strong problems of NMAC are the result of a high rate of collisions. This is due to the fact that nodes which are leaving each others vicinity and thus produce a high number of transmission errors are seen as unidirectional links by both nodes and thus not addressed in the RTS message. They don't forward the CTS, which leads to another rise in collisions. The problem gets worse when nodes re-enter each others vicinity shortly after leaving it, because their links remain marked as unidirectional too long.

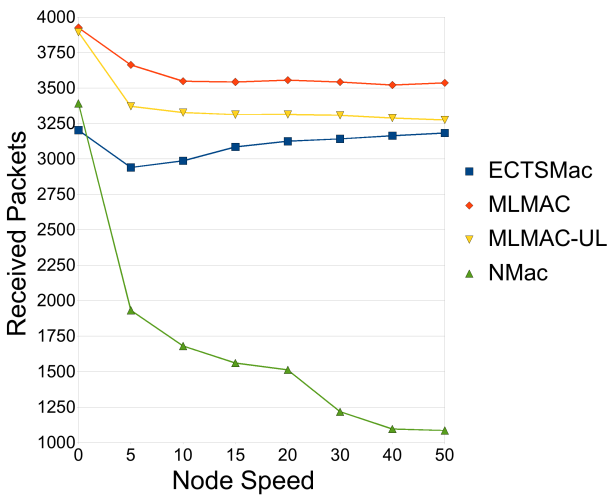


Figure 15. Received Packets

D. Simulated Flooding over 50 hops

In this set of simulations, the performance under low network load is evaluated. We simulated a line of 6 to 51 nodes, where each node was only able to communicate with its direct neighbors. Table II shows the time needed by each protocol to deliver a message over 50 hops. The times for the TDMA protocols are divided once using 5 slots and once using 31. Even though there were enough unused slots, the MLMAC-UL did not acquire new ones, because there was not much data to be sent and the send queue only ever held one packet. This leads to nearly the same time (one frame) needed as when using the original MLMAC, as the time for one hop only depended on the frame length. For all protocols, the time needed to reach the last node increased linearly with the number of nodes in use.

Table II
TIME NEEDED FOR 50 HOPS (MS)

NMAC	3,77
MLMAC-UL 31 slots	70,29
MLMAC 31 slots	69,25
ECTS-Mac	3,63
MLMAC-UL 5 slots	12,32
MLMAC 5 slots	10,79

E. Packet Overhead

This last evaluation in the simulator was based on the same topology as the high load scenario, but the size of the data generated by the application was varied between 20 and 110 Byte. It transmitted at random intervals between 0 and 5000 milliseconds. Figure 16 shows the relative overhead each protocol produced (protocol bytes/total bytes) for an increasing size of the 2-hop-neighborhood. The calculation includes the periodic messages from the TDMA based protocols and the RTS, CTS and ECTS messages from the contention based protocols. It can be seen that NMAC produces by far the highest overhead, followed by the ECTS-MAC. Thus, contrary to common belief, sending periodic status messages does not produce a high overhead.

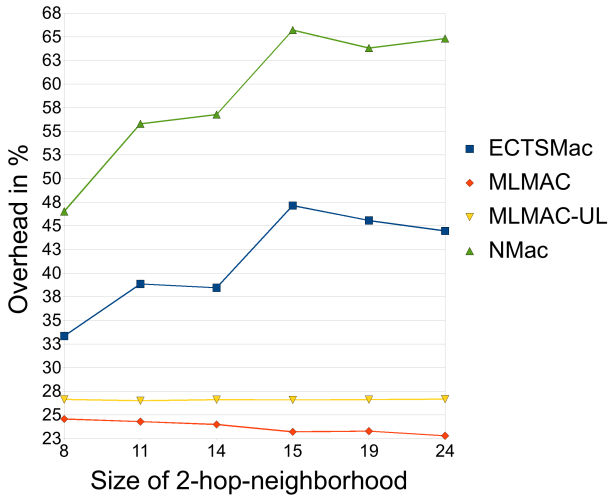


Figure 16. Protocol Overhead

F. Direct Neighborhood Experiments

In these experiments the application sent 500 packets of size 110 Byte every 10 Milliseconds. They were performed using 3, 7, 11 and 16 nodes. Figure 17 shows that for a small number of nodes all protocols perform relatively well. With an increasing number of nodes the performance of first NMAC and then MLMAC drop considerably. This is due to

the increased number of CTS and ECTS messages, which lead to a high network load, a lot of collisions and thus a low throughput.

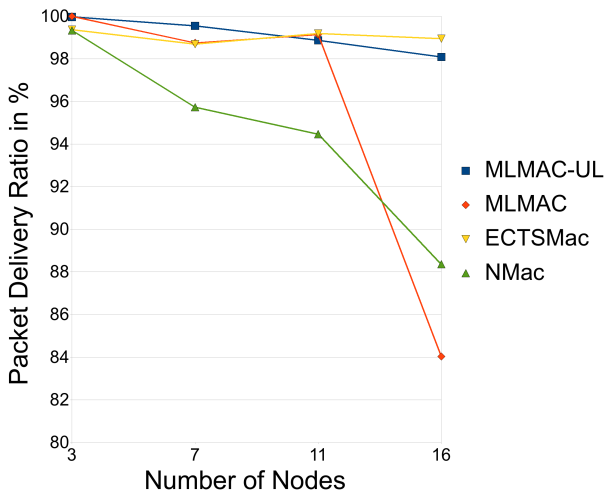


Figure 17. Packet Delivery Ratio Single Hop Experiments

G. High Load Scenario Experiments

For these experiments we placed 14 TMote Sky sensor nodes on the floor in a building. As there are no ways to define link quality in a real experiment we could only measure it. Figure 18 shows the resulting communication graph. It can be seen that the radio neighborhood of the nodes and the link quality differ a lot.

The application was once again the one producing the high network load. The left side of Figure 19 shows the average time needed to transmit one packet. MLMAC-UL and ECTS-MAC were the fastest ones, with MLMAC following and NMAC bringing up the rear. On the right side of the figure you can see the total number of received packets for each protocol. All protocols received nearly the same amount of messages, with only NMAC being considerably better

But this fact has to be put in perspective: all 4 protocols were evaluated one after another, using the same nodes and, most important, the same batteries. NMAC was the first protocol to be evaluated, which means that is had the advantage of fresh batteries which have been shown to have a positive effect on the range of the transceivers and thus link quality.

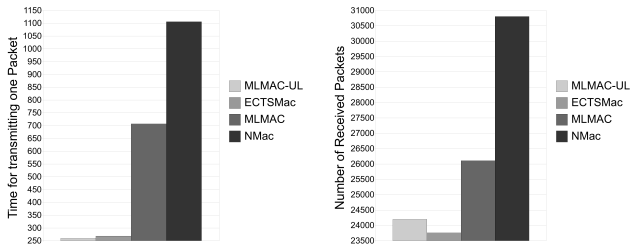


Figure 19. Time to Send(l) and Number of received Packets(r)

H. Memory Consumption

On Figure 20 the memory consumption of the protocols is shown, with 3 slots for the TDMA protocols on the left and 16 slots on the right. It is also differentiated whether the neighborhood discovery protocol was used or not, only the original MLMAC is shown only once, because it never uses that protocol. Please note that the numbers shown are for RAM consumption, the usage of flash memory follows the same distribution. On the figure it can be seen that MLMAC-UL needs most memory and ECTS has the lowest memory consumption. Combining this fact with the other results leads to the observation that for networks with low memory allowance and few nodes the ECTS-MAC should be chosen while the MLMAC-UL is best suited for denser networks with high load.

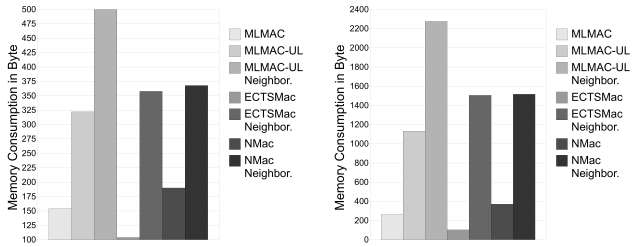


Figure 20. Memory Consumption for 3(l) and 16 Slots(r)

VII. Conclusion and Future Work

In this paper we have discussed the influence of unidirectional links on communication protocols. We presented two MAC protocols that can utilize unidirectional links to increase network connectivity, reliability and lifetime. MLMAC-UL is an enhancement to MLMAC, a TDMA based protocols for wireless sensor networks. ECTS-MAC is a contention based protocol that informs nodes that are connected to a sending node through unidirectional links about the impending communication by forwarding the CTS messages through multiple hops. Both protocols were evaluated by comparison with other protocols in simulations

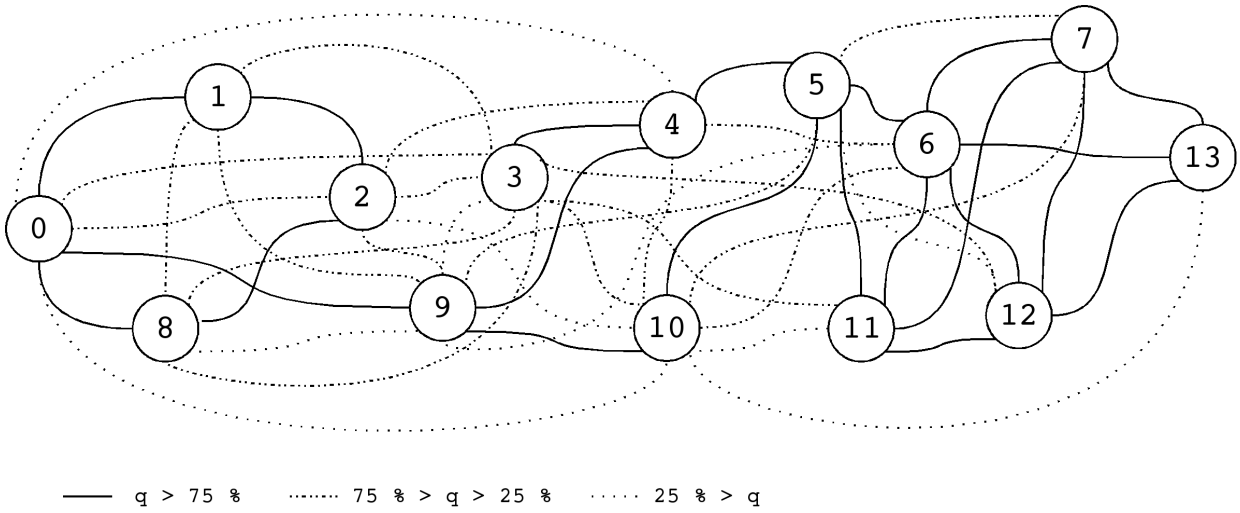


Figure 18. Link Graph where q denotes the Link Quality

and experiments with TMote Sky sensor node hardware. Both protocols show good results for different scenarios.

The choice of protocol depends strongly on the intended scenario and thus the application. If the network load is expected to be fairly low, ECTS-MAC is a good candidate. For high load scenarios however, MLMAC-UL performs far better due to the high number of messages generated by ECTS-MAC and the resulting collisions. Of course, node density and memory size are also important factors as The memory footprint of MLMAC-UL is considerably larger that that of ECTS-MAC. If the application needs lots of memory on a typical sensor node, MLMAC-UL simply might not fit in. In high density networks TDMA protocols normally suffer from a large frame size. If only a few nodes need to transmit data, they still have to wait for the slots of all other nodes to pass before transmitting again. In this case MLMAC-UL would prevail over ECTS-MAC only because of its adaptive nature, as nodes that need to transmit more data can acquire additional slots and release them once they are not needed anymore.

In the future we will continue our research on routing protocols that can make use of unidirectional links, building on the MAC protocols presented here. The possibilities of sharing information across layers, e.g. the data gathered by the neighborhood discovery protocol of MLMAC-UL, will also be explored.

REFERENCES

[1] S. Mank, R. Karnapke, and J. Nolte, "Mlmac-ul and ects-mac - two mac protocols for wireless sensor networks with unidirectional links," in *Third International Conference on Sensor Technologies and Applications*, (Athens, Greece), 2009.

[2] Turau, Renner, and Venzke, "The heathland experiment: Results and experiences," in *Proceedings of the REALWSN'05 Workshop on Real-World Wireless Sensor Networks.*, Jun 2005.

[3] J. Schiller, A. Liers, H. Ritter, R. Winter, and T. Voigt, "Scatterweb - low power sensor nodes and energy aware routing," in *Proceedings of the 38th Hawaii International Conference on System Sciences*, 2005.

[4] V. Turau, "The heathland experiment: Results and experiences, presentation. www.tm.uka.de/forschung/spp1140/events/kolloquium/2005-11/turau.pdf."

[5] L. Sang, A. Arora, and H. Zhang, "On exploiting asymmetric wireless links via one-way estimation," in *MobiHoc '07: Proceedings of the 8th ACM international symposium on Mobile ad hoc networking and computing*, (New York, NY, USA), pp. 11–21, ACM Press, 2007.

[6] D. S. J. D. Couto, D. Aguayo, J. Bicket, and R. Morris, "A high-throughput path metric for multi-hop wireless routing," in *MobiCom '03: Proceedings of the 9th annual international conference on Mobile computing and networking*, (New York, NY, USA), pp. 134–146, ACM, 2003.

[7] R. Min and A. Chandrakasan, "A framework for energy-scalable communication in high-density wireless networks," in *ISLPED '02: Proceedings of the 2002 international symposium on Low power electronics and design*, (New York, NY, USA), pp. 36–41, ACM, 2002.

[8] T. L. Dinh, W. Hu, P. Sikka, P. Corke, L. Overs, and S. Brosnan, "Design and deployment of a remote robust sensor network: Experiences from an outdoor water quality monitoring network," in *LCN '07: Proceedings of the 32nd IEEE Conference on Local Computer Networks*, (Washington, DC, USA), pp. 799–806, IEEE Computer Society, 2007.

[9] A. Woo, T. Tong, and D. Culler, "Taming the underlying challenges of reliable multihop routing in sensor networks," in *SenSys '03: Proceedings of the 1st international conference on Embedded networked sensor systems*, (New York, NY, USA), pp. 14–27, ACM Press, 2003.

[10] T. He, S. Krishnamurthy, L. Luo, T. Yan, L. Gu, R. Stoleru, G. Zhou, Q. Cao, P. Vicaire, J. A. Stankovic, T. F. Abdelzaher, J. Hui, and B. Krogh, "Vigilnet: An integrated sensor network system for energy-efficient surveillance," *ACM Trans. Sen. Netw.*, vol. 2, no. 1, pp. 1–38, 2006.

[11] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva, "Directed diffusion for wireless sensor networking," *IEEE/ACM Trans. Netw.*, vol. 11, no. 1, pp. 2–16, 2003.

[12] H. Gossain, N. Nandiraju, K. Anand, and D. P. Agrawal, "Supporting mac layer multicast in ieee 802.11 based manets: Issues and solutions," in *LCN '04: Proceedings of the 29th Annual IEEE International Conference on Local Computer Networks*, (Washington, DC, USA), pp. 172–179, IEEE Computer Society, 2004.

[13] R. M. Yadumurthy, A. C. H., M. Sadashivaiah, and R. Makanaboyina, "Reliable mac broadcast protocol in directional and omni-directional transmissions for vehicular ad hoc networks," in *VANET '05: Proceedings of the 2nd ACM international workshop on Vehicular ad hoc networks*, (New York, NY, USA), pp. 10–19, ACM, 2005.

[14] S. Jain and S. R. Das, "Mac layer multicast in wireless multihop networks.," in *COMSWARE '06: Proceedings of the 1st International Conference on Communication System Software and Middleware*, 2006.

[15] G. Wang, D. Turgut, L. Bölöni, Y. Ji, and D. C. Marinescu, "A simulation study of a mac layer protocol for wireless networks with asymmetric links," in *IWCMC '06: Proceedings of the 2006 international conference on Wireless communications and mobile computing*, (New York, NY, USA), pp. 929–936, ACM, 2006.

[16] V. Ramasubramanian, R. Chandra, and D. Mosse, "Providing a bidirectional abstraction for unidirectional ad hoc networks," 2002.

[17] N. Poojary, S. V. Krishnamurthy, and S. Dao, "Medium access control in a network of ad hoc mobile nodes with heterogeneous power capabilities," in *in: IEEE International Conference on Communications (ICC), 2001*, pp. 872–877, 2001.

[18] L. Bao and J. J. Garcia-Luna-Aceves, "Channel access scheduling in ad hoc networks with unidirectional links," in *DIALM '01: Proceedings of the 5th international workshop on Discrete algorithms and methods for mobile computing and communications*, (New York, NY, USA), pp. 9–18, ACM, 2001.

[19] S. Mank, R. Karnapke, and J. Nolte, "An adaptive tdma based mac protocol for mobile wireless sensor networks, best paper award," in *International Conference on Sensor Technologies and Applications*, 2007.

[20] S. Mank, R. Karnapke, and J. Nolte, "Mlmac - an adaptive tdma mac protocol for mobile wireless sensor networks," in *Ad-Hoc & Sensor Wireless Networks: An International Journal, Vol.8 Nr.1-2*, 2009.

[21] L. van Hoesel and P. Havinga, "A lightweight medium access protocol (lmac) for wireless sensor networks: Reducing preamble transmissions and transceiver state switches," in *INSS, Japan*, Jun 2004.

[22] A. Varga, "The omnet++ discrete event simulation system," in *Proceedings of the European Simulation Multiconference (ESM'2001)*, (Prague, Czech Republic), June 2001.

A Novel Fault Diagnosis Technique in Wireless Sensor Networks

Anas Abu Taleb, J. Mathew and D.K. Pradhan
Department of Computer Science
University of Bristol
Bristol, UK
{abutaleb, jimson, pardhan}@cs.bris.ac.uk

Taskin Kocak
Department of Computer Engineering
Bahcesehir University
Istanbul, Turkey
taskin.kocak@bahcesehir.edu.tr

***Abstract*—In sensor networks, performance and reliability depend on the fault tolerance scheme used in the system. With increased network size traditional fault tolerant techniques have proven inadequate. Further, identifying and isolating the fault is one of the key steps towards reliable network design. Towards this, we propose two new algorithms to detect and substitute faulty nodes at different levels in the network. In the proposed approach, the network is divided into zones which are having a master for each zone. Moreover, the masters of the zones are connected in a De Bruijn graph based network. When a fault occurs, the masters are checked, tested. After that, the sensor nodes in the suspected zone are tested. Our fault model assumes communication, processing and sensing faults caused by hardware failures in a node. We analyzed the performance of the first algorithm according to the number of messages it needs to diagnose faulty nodes. In addition, the performance of a 4-node De Bruijn graph was also studied by measuring the end-to-end delay. Finally, the performance of the second algorithm was studied by measuring the fault detection accuracy.**

Keywords- ***Wireless Sensor Networks, Fault Tolerance, Fault Diagnosis, De Bruijin Graph***

I. INTRODUCTION

The advances in wireless communication and electronics made it possible to develop low-cost sensor nodes, which can be deployed easily in specific areas in order to accomplish a specific mission by forming a wireless sensor network (WSN). It might be difficult or dangerous for humans to enter these areas because nodes in this type of networks are expected to operate in inhospitable environments [2]. Therefore, sensor nodes are expected to operate for periods ranging from days to years without any human intervention. There is a tremendous need for fault tolerant WSNs because, sensor nodes are subject to various types of failures and faults such as communication, processing and sensing faults.

A sensor network must be capable of identifying and replacing the faulty nodes in order to make sure that the network's quality-of-service (QoS) is maintained. Identifying faulty sensor nodes is not an easy task as it is difficult and time consuming for the base station to keep the information about all the sensor nodes in the network. When addressing fault tolerance in WSNs, three types of node failures must be taken into account. First, when the sensor node is faulty and not providing data. Second, when the node processes data erroneously. Third, occurs when we have an active node that is providing incorrect data.

In this paper, we propose a new technique consisting of two algorithms to identify faults occurring at different levels or places in the network, i.e. faults that occur at the zones masters and the sensor nodes associated with the zones masters. The proposed technique divides the network into disjoint zones while having a master for each zone. When a fault occurs, the first algorithm is triggered to test the masters. The technique will not trigger the second algorithm unless all the masters are diagnosed fault free by the first algorithm. Thus, when the second algorithm is triggered, the master of the suspected zone is responsible for identifying the suspected faulty nodes. As a result, the master will start searching for sleeping nodes to wake up and depending on the reading it gets from the suspected and awakened nodes, the master can decide whether the suspected nodes are faulty or not, moreover, it can decide on which node to switch off. A preliminary version of this paper is published in [1], in which a technique to detect faulty sensor nodes was presented.

The paper is organized as follows. In section II the related work is reviewed. Then the concept of De Bruijn graph is discussed and explained in section III. In section IV, the network architecture and the fault model are defined. In section V the proposed technique is described. Section VI describes the simulator used and illustrates the simulated scenarios. Also, we use an example of a potential chemical spill to describe various concepts. The simulation results are also reported in this section. Finally the paper is concluded in section VII.

II. RELATED WORK

Several works have addressed the problem of how to deal with faults occurring in wireless sensor networks in order to achieve fault tolerance [3][4][5]. These researches consider the faults that result from sensor nodes failures, which affect the network connectivity and coverage. The research proposed in [3], makes use of redundancy and uses a technique to decide on which nodes to keep active and on which to put in a sleep mode. The technique aims to provide the sensor field with the best possible coverage. In addition, it maintains network connectivity to route information. When an active node fails it is substituted by one of the sleeping nodes. However, other researchers have addressed

the problem of having active nodes that provide incorrect data which results in making inappropriate decisions. The research proposed in [4] focused on such issues and proposed a mechanism to detect and diagnose data in consistency failures in wireless sensor networks.

The mechanism proposed in [4] uses two disjoint paths to send the sensed data to a static sink. After the sink receives both copies, it will compare them to check if they match. If the two copies match, both the data and the paths are considered to be fault free otherwise, a third disjoint path will be established. Then, the sensor node will send three copies on the three disjoint paths to the sink. The sink will compare these copies and decides on the faulty path. Finally, a diagnosis routine will be executed to identify the faulty node within the faulty path.

Another research has taken fault tolerance in account, so that to achieve fault tolerance the sensor network is partitioned into distinct clusters and the node that has the highest energy level is selected to be the cluster head where only cluster heads are allowed to communicate with the base station [5]. Therefore, they introduced a two-phase fault tolerant approach which consists of detection and recovery where the status of the cluster heads is checked periodically. Sensors associated with a faulty cluster head are recovered by joining them to another cluster [5].

The research described in [6] proposed a scheme based on multi-path routing combined with channel coding to achieve fault tolerance. It uses a fuzzy logic based algorithm that is energy and mobility aware to select multiple paths. When selecting the paths, the algorithm takes the remaining energy, mobility and the distance to the destination into account. Another research has proposed a design for a system to diagnose the roots of faults occurring in wireless sensor networks. The authors have proposed an algorithm to diagnose the cause of faults in which the behavior of sensor nodes in monitored locally. The diagnosis procedure will be triggered when a node detects a strange behavior [7]. In [8], a general framework to achieve fault tolerance in wireless sensor networks was proposed. The framework is based on a learning and refinement module which provides adaptive and self-configurable solutions.

A localized algorithm for fault detection to identify faulty sensors that is based on having neighbor sensor nodes testing each other was proposed in [9]. In [10], an efficient algorithm to trace failed nodes in sensor networks was proposed. In addition, they demonstrate that if the network topology is conveyed efficiently to the base station, it allows tracing the failed entities quickly with moderate communication overhead.

In [11], the authors proposed fault tolerant algorithms to detect the region of an event in wireless sensor networks. Also, they assume that nodes report a binary decision to indicate the presence of an event or not and considered a byzantine behavior for the faulty nodes, which means that the faulty nodes will be providing arbitrary values. Hence, they proposed a randomized decision scheme and a threshold decision scheme which a sensor node can use to decide on which binary decision to send by comparing the decision it has with the decisions of its neighbors

In [12], a fault map was constructed using a fault estimation model. In order to build the fault map, sensor nodes are required to send additional information that can be used by the fault estimation model. Furthermore, a cluster based algorithm to estimate faults in wireless sensor networks was proposed. In [13], a target detection model for sensor networks was proposed. In addition, two algorithms to facilitate fault tolerant decision making were presented. The first algorithm is based on collecting the actual readings from the neighboring nodes. In the second algorithm, the sensor node obtains the decisions made by the other neighboring nodes to take a final decision.

A distributed cluster based fault tolerant algorithm was proposed in [14]. The cluster head sends a small packet to indicate that it is still alive. Hence, a sensor node in the same cluster listens to the transmissions of its neighbors and to that of the cluster head. When a sensor node does not receive the short packets sent by the cluster head, it will trigger fault detection. Depending on the number of nodes that have not heard from the cluster head, it can be decided whether the cluster head is faulty, as the faulty node can be a member of the cluster and not the cluster head itself. If the cluster head was faulty, the cluster members will select a new cluster head. The authors in [15] apply error correcting codes to achieve fault tolerance. As a result, a distributed fault tolerant classification approach was proposed. The approach proposed is base of fault tolerant fusion rules that are used to obtain local decision rules at every sensor. In addition, the authors proposed two algorithms that can be used to find good code matrices to be used by the classification approach.

The work proposed in this paper differs from that presented by other researches in two aspects. First, the mechanism according to which sleeping nodes are activated to test active node. Second, the reading of neighboring nodes i.e. nodes covering the same terrain, are needed and compared only when the network is suspected to contain faulty nodes.

Moreover, we compare the performance of our work to the performance of the work presented in [11] because both techniques make use of neighboring nodes to detect a fault. In addition, no restriction on the number of neighboring nodes is imposed. Also, both techniques make use of threshold in their operation.

III. DE BRUIJN GRAPH

Part of the work proposed in this paper is based on constructing a De Bruijn graph based network at the zones masters level. This graph has interesting properties that make it important to investigate its use in WSNs. The degree of this graph is bounded, which means the degree of the network remains fixed even when the network size increases. In addition, this graph has interesting properties such as small diameter, high connectivity and easy routing. Furthermore, De Bruijn graph contains some important networks such as ring. Regarding fault tolerance and extensibility, these graphs maintain a good level of fault tolerance and self-diagnosability. For instance, in the presence of a single fault in the network, it takes four additional hops to detour around the faulty node and the

control information needed to do so can be integrated locally between the faulty node's neighbors. Also, De Bruijn graph is extensible in two methods that are described in [18].

As a result, it will be interesting to investigate the used of De Bruijn graphs in sensor networks in order to increase the fault tolerance capabilities. In other words, if some nodes in the network were deployed according to De Bruijn graph, the network will have the ability to tolerate the presence of faulty nodes in the network and remain functional. In this work, the zone masters are assumed to be connected according a De Bruijn graph. The network assumed to be working if a zone master fails and the rest of the nodes in network will remain functional and the fault free zone master are capable of communicating with each other. Thus, accomplishing the network mission until the problem in the network is resolved.

The De Bruijn graph denoted as DB(r, k) has $N = r^k$ nodes with diameter k and degree 2r. This corresponds to the state graph of a shift register of length k using r-ary digits. A shift register changes a state by shifting in a digit in the state number in one side, and then shifting out one digit from the other side. If we represent a node by $I = (i_{K-1} i_{K-2}, ..., i_1 i_0)$ where $i \in 0,1,...,(r-1)$, $0 \le j \le (k-1)$, then its neighbors are represented by $i_{k-2} i_{k-3}, ..., i_0 p$ and $p i_{k-1} i_{k-2}, ..., i_1$, where $p = 0,1,...,(r-1)$. The DB(2, k), which is called binary De Bruijn graph, can be obtained as follows. If we represent a node I by a k-bit binary number, say, $I = i_{k-1} i_{k-2}, ..., i_1 i_0$, then its neighbors can be presented as $i_{k-2}, ..., i_1 i_0 0$, $i_{k-2}, ..., i_1 i_0 1$, $0 i_{k-1} i_{k-2}, ..., i_1$, and $1 i_{k-1} i_{k-2}, ..., i_0$.

IV. NETWORK ARCHITECTURE AND FAULT MODEL

We assume that the network is densely deployed and consists of heterogeneous nodes; which means that in addition to the ordinary sensor nodes, the network consists of some nodes that are more energy rich than others. The energy rich nodes are placed or deployed in a way that guarantees them to form a De Bruijn based network, while the rest of the nodes are deployed randomly. Also, the network has most of the nodes awake and a small number of nodes are in a sleep status. The nodes are fully static and the network is divided into four zones. In each zone the active node with the highest energy level will be chosen to be the zone master, for example in the shaded zone in Fig.1 the zone master is node 9, where the dark dots are the active nodes. After that, the master acts as a data sink and will be responsible for identifying faulty nodes in its zone while the remaining nodes in a zone can only send the sensed data to their master.

After being elected as zones maters for their zones, the zone masters communicate among themselves. In other words, each zone master knows the neighboring zone masters in the neighboring zones. Hence, a De Bruijn graph based network, consisting of the zone masters only, is constructed. This graph has interesting properties that assist in increasing fault tolerance capabilities of the network. Figure 2 shows the DB(2,2) De Bruijn Graph.

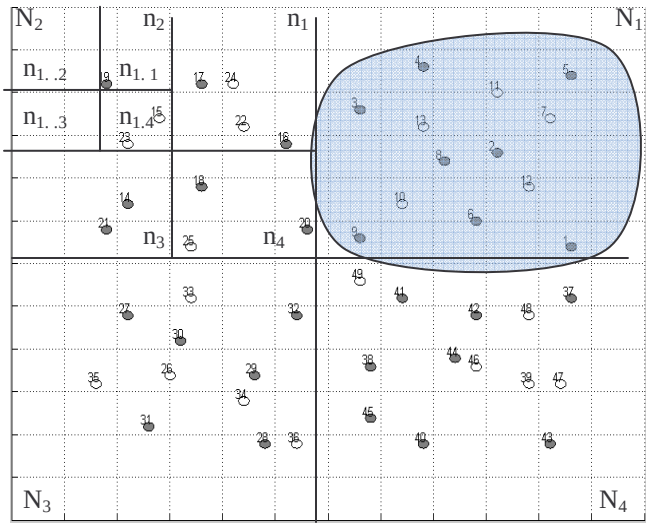


Figure 1. The Network Architecture.

After constructing the De Bruijn based network, when a zone is suspected to contain faulty nodes, the master of that zone will be tested and diagnosed using the distributed fault diagnosis algorithm described in section V. If the zone master is faulty, it will be substituted by one of its neighboring sleeping nodes, as a result, the De Bruijn based network will not need to be constructed again. On the other hand, if the master was fault free, the technique proceeds to test the nodes in that zone, as described before, until the faulty node is identified. When a sensor node is suspected to be faulty, the master activates some of the sleeping nodes to check the correctness of that node and to substitute it when the suspected node is identified as faulty. Figure 1 illustrates the network architecture where the main zones are the big squares denoted by N_1, N_2, N_3 and N_4. Also, the division process is illustrated in Fig.1 where N_2 is divided into four subzones denoted by n_1, n_2, n_3 and n_4. Furthermore, n_2 is divided into $n_{2.1}$, $n_{2.2}$, $n_{2.3}$ and $n_{2.4}$, thus node 19 is suspected to be faulty which is in $n_{1.1}$.

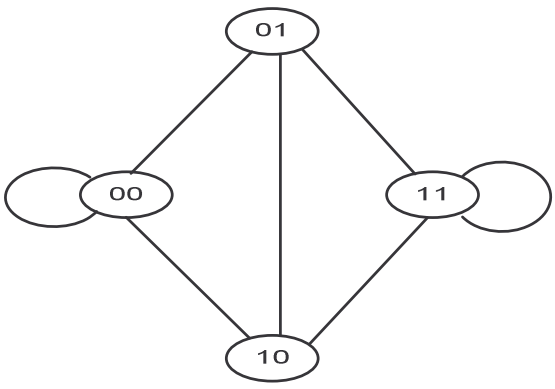


Figure 2. DB(2,2) Binary De Bruijn Graph.

A sensor node is considered to be faulty, if it reports a value that deviates from the expected one [16]. In this work, we restrict our attention to permanent faults, resulting from sensing and communication faults caused by hardware failures are considered at the sensor nodes level. At the zone master level, permanent faults caused by communication and processing fault are considered. Processing faults are taken into account at the zone master level because, the zone masters have to calculate their zones throughput periodically. As a result, if a master is suffering from processing failure the calculation acquired will be misleading.

For communication faults, we propose to measure the throughput (T) of a zone or a subzone and compare the calculated value of throughput to predefined thresholds of the tested zone (τ) or subzone (τ_{sub}). As a result, the presence of a communication fault is detected if $T < \tau_{sub}$. On the other hand, we detect the presence of a sensing fault by measuring the discrepancy between the readings reported by the sensor nodes involved in the test. Discrepancy between sensors readings is used because we consider that the sensor nodes report the actual values rather than binary decisions [17].

V. PROPOSED IDENTIFICATION TECHNIQUE

Because WSNs are deployed in inhospitable environments, sensor nodes are prone to faults such as communication, sensing and processing faults. As a result, the node that suffers from a communication fault will not be reporting data to its zone master in the same rate as a non-faulty one does. This will result in the throughput of that zone to decrease. On the other hand, nodes that suffer from a sensing fault will be providing data frequently to the zone master, but the data reported will be erroneous. Also, the nodes may suffer from processing faults at the zone master level. Hence, the aggregated or fused data at the faulty master will be erroneous and affects the decision made to detect the faulty nodes. Thus, there is a need to detect these faults and eliminate their effect.

Therefore, we consider dividing the network into zones, to allow faster identification and location of faults occurring in a zone since the master is responsible for a small number of nodes. In addition, the master can keep track of the data sent to it by the members of its zone more efficiently. In addition, the approach starts by testing the master nodes in the first stage to avoid testing individual nodes in the zones when the master is faulty.

A. Overview of The Proposed Approach

The technique is based on periodically calculating the throughput of the four zones. Each zone master will calculate the throughput of its zone and will compare it to a predefined threshold; if it is less than the threshold, the distributed diagnosis algorithm will be triggered to test the masters. If a zone master was diagnosed as faulty, it will be replaced and the technique will not proceed to test the sensor nodes in that zone. However, if all the masters were diagnosed as fault free, the technique proceeds to test the sensor nodes in the zone that has provided low throughput.

As a result, the zone master will start dividing its zone virtually into quadrants. After that, the zone master will calculate the throughput of each quadrant and will compare it to another threshold. If the throughput of one of the quadrants is less than a threshold, the zone master will divide that quadrant for another four quadrants. The zone master will keep dividing the zone virtually and calculating the throughput until it reaches a quadrant that contains only one node. As a result, it can identify that the node enclosed in that quadrant is the suspect that is causing the throughput to be low. After identifying the suspect node, the zone master will start searching for sleeping nodes that are near to the suspect to wake them up to test the suspect node. Note that a zone or a subzone is divided by calculating its center, after that, it will be divided into four subzones that have equal size.

In addition, when a sensor node reports data to the zone master, the data will be compared to the node status and the data ranges values stored in the master. If the data reported deviates from the stored values, the master will start dividing the zone virtually until a suspect is identified. After that, it will start searching for neighboring sleeping nodes to activate in order to test the suspect.

The proposed technique has the following distinctive feature; first the distributed fault diagnosis algorithm that is used to diagnose faults at the zones masters level does not use a central node to trigger and carry out the diagnosis process. The second feature is the way in which faulty nodes associated with the zone master are identified or pinpointed by dividing the zone into quadrants. The third feature is the mechanism used to make sure that the suspect node is faulty which is conceptually similar to our previous work on the multi-processor environment [19]. In the Roll-forward Checkpointing Scheme, two copies of the same task will be run on two different processing modules while having a pool of spare processing modules. At every checkpoint, the state of the two processing modules is compared, if they mismatch, the state of the last checkpoint on which the state of the two processing modules has matched will be loaded into a spare processing module, while the other two processing modules continue the execution of the task beyond the checkpoint where a mismatch occurred. At the next checkpoint, the state of the spare processing module will be compared to the stored state of the other two processing modules. As a result, the processing module whose state disagrees with that of the spare will be the faulty one. After identifying the faulty processing module, the state of the non faulty processing module is copied to the faulty one to restore its state [19]. A similar scheme was applied in this work by activating one of the sleeping neighbors of the suspected node. Both nodes will sense their region simultaneously. After receiving the data, the sink compares the data sent by both nodes. If they match or were similar, the suspect node will be considered fault free and the activated neighbor goes to sleeping mode again. Otherwise, another sleeping neighbor is activated, and after the three nodes sense their region and send data to the sink, the sink can identify the faulty node using the mechanism mentioned above. If the faulty node was the originally active one, it is deactivated and one of the activated neighbors is selected to substitute it. On the other hand, if the faulty node was the

first activated neighbor, it is flagged as faulty and is suspend from the network.

B. Design and Implementation

The technique is divided into the following phases:

1) Initialization Phase

The nodes are grouped into four zones depending on their positions and the active node with the highest energy level in each zone is chosen as the zone master. The zone master keeps track of the data sent to it from the other nodes in its zone. Also, it acts as a data sink for the nodes. This means that the ordinary nodes in the zone can only send data to the zone master which is responsible to forward it to the base station. After that, the zones masters will communicate with each other. Hence, each zone master knows its neighboring zone masters. As a result, a De Bruijn graph based network consisting of the masters only is constructed.

The zone master will be able to keep track of the data it received and of the nodes belonging to its zone by maintaining an information table and a registration table.

In the information table the zone master records the sender's ID of the received message, the packet length and the time stamp to indicate when the message was received. As a result, this table gives the master the ability to keep track of the data sent in its zone. An example is given in Table 1.

Table 1. Information table

NodeID	Message_length	Time_Stamp
8	4	0.501
1	4	0.504
6	4	1.002
2	4	1.003

The registration table is used by the master to keep track of the nodes inside its zone and their positions. In addition, it contains some entries that will give the master node the ability to divide the zone into quadrants or subzones when needed. In other words, XMax and YMax entries are used to know the coordinates of the zone. In addition, they are used by the master when there is a need to divide the zone into subzone. The Center attribute is calculated because it is used as a reference point when dividing a zone or a subzone. An example is given in Table 2.

In order to be able to detect the presence of a sensing fault, a third table, which is called *Grid table*, is maintained by each zone master. The zone master divides its zone virtually to a grid and stores the information in the grid table. Note that three binary values are used to indicate the status of a node because a node reports the value that corresponds to its original reading which depends on the node's proximity from an event. An example is given in Table 3.

Table 2. Registration Table

NodeID	XPosition	YPosition	XMax	YMax	Center
1	23	5	30	30	15
2	16	13	30	30	15
10	7	7	30	30	15
11	16	17	30	30	15

Table 3. Grid Table

Square-Number	Enclosed_nodes	Low	Medium	High
1	[9, 10]	0	0	1
4	3	0	1	0
7	[4, 13]	1	0	0
9	5	1	0	0

In the initialization phase, all the nodes will be providing low data values to indicate that no event was detected. After that, when an event occurs the value can be changed to medium or high based on the position of the node. A sensor node has three different values to choose from when it is about to send data to the master which are low, medium and high. For example, if the sensor node is in a place where there is a very high concentration of a chemical spill, it will send the value stored in the high field of the grid table to indicate that there is a high chemical spill in its region. As a result, the other nodes will choose to send low, medium or high data values depending on their positions and distance from that node.

In Fig.3, the pseudo code used in the initialization phase is illustrated. It can be observed that after the nodes are deployed, the network is divided into four zones and the nodes are allocated to the zones as mentioned before. In addition, each zone master will initialize its registration table, mentioned above, and will store the needed information about the nodes belonging to its zone. Finally, the grid table will be initialized, i.e. the zone will be virtually divided into a grid, and the node or nodes belonging to every square in the grid are identified.

```
For each zone
    Find the node with the highest energy level to be the
     zone_master
    Initialize Registration table for zone_master
   Set the ID of the new entry to the ID of the current node
   Get node position
```

```
    Calculate zone center
    Update zone registration table
    Initialize Grid table for zone_master
End Loop
For each zone Grid table
  Locate nodes to the grid squares according to their positions
End Loop
 For each zone_master
   Find the neighboring zone_masters
End Loop
```

Figure 3. Initialization Pseudo Code.

2) Failure Detection Phase

The zone master is responsible for checking the throughput of its zone periodically. This is done by calculating the throughput of the zone since the last time the throughput was checked until the current time. This process can be described as taking a snapshot of the information table depending on the specified period of time. Subsequently, the master compares the calculated value of the throughput to a threshold "τ". If the value of the calculated throughput is greater than τ, the master concludes that there is no communication fault in the zone. However, if the value of the calculated throughput is less than τ, the master assumes that there is a communication fault in the zone and initiates the failure detection phase.

The failure detection phase starts by testing the masters first because, the zone master might be suffering from a processing fault. The distributed De Bruijn based fault diagnosis algorithm is used to test the masters. The number of nodes in a De Bruijn based network is assumed to be equal to r^m, where r is a parameter that bounds the number of faults that can be diagnosed in each cluster and will be referred to as *base* parameter in this paper. The variable *m* is the radix-r representation of the node address e.g. $y_{m-1}y_{m-2}, ..., y_0$ is the radix-r representation of node y. Also, the number of faults that can be diagnosed is equal to r -1 [20]. In addition, we assume that nodes can test their neighbors only.

The algorithm is based on building directed tree structure for the De Bruijn based network. According to our previous work in [20], r different tree structures can be built where each one of them has a different root. In this paper, the base variable *r* is equal to 2 which mean we can diagnose only one fault and we can build two tree structures for the De Bruijn based network. Figures 4 and 5 illustrate two trees that can be built for a 4-node De Bruijn based network.

Consider Fig. 4, the following conditions are satisfied:

- The test tree must contain all the nodes in the cluster.
- The number of non leaf nodes is equal to r^{m-1}.
- The number of leaf nodes is $(r-1)r^{m-1}$.
- Any combination of r – 1 nodes must appear in at least one tree.

The algorithm is triggered at the zones masters level. As a result the first tree is built to test the zones masters. The test tree is traversed in an inorder fashion. According to Fig. 4 the root node, 0, initiates the process by sending a test packet to node 2. Then, node 2 checks if it is a leaf node. In this case, node 2 in a non leaf node, thus a test packet will be sent to its left child, node 1. This process continues until we reach a leaf node. When a leaf node, for example node 1, receives a test packet, it will execute the required computation for the test and send the result back to its parent, node 2. Node 2 compares the result received from node 1 with the expected or the predefined one. If a miss match occurs node 1 will be considered faulty and its status will be reported back to the root node that is responsible for sending it to the base station.

Note that, the algorithm will stop after finding the faulty node. Also, the faulty node can be detected only if it is a leaf node in the test tree shown in Fig. 4. However, if the faulty node is a non leaf node in the first tree, the algorithm cannot diagnose whether the non leaf node is faulty or there is a communication problem between that node and one of its children. As a result, when a non leaf node is suspected to be faulty, the algorithm will stop searching the tree shown in Fig. 4 and will construct the second test tree shown in Fig. 5. After constructing the second tree, the test packets will be passed in the same manner as mentioned before. The faulty node can be detected because; it is a leaf node in the second tree. After diagnosing the nodes at one level, the algorithm proceeds to test the nodes in the subsequent level.

The test packet sent to diagnose the nodes triggers the tested node to perform a specific computation whose result is known in advance. Therefore, if the tested node provides a value that deviates from the expected one it will be diagnosed as faulty.

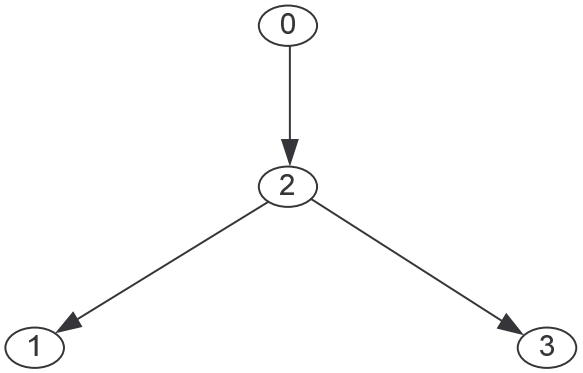


Figure 4.Diagnosis algorithm Tree A.

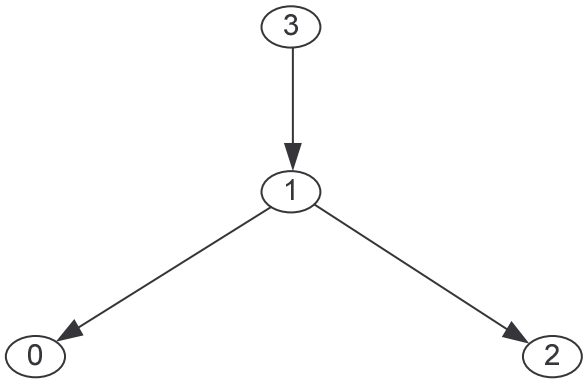


Figure 5.Diagnosis algorithm tree B.

If all master nodes were diagnosed fault free, our approach proceeds to test the sensor nodes associated with the zone master that has calculated low throughput.

Based on the information maintained in the registration table, the master, that has calculated a low value of throughput, starts dividing its zone into quadrants. In the first stage the zone master divides its zone into four quadrants or subzones. After that, it will calculate the throughput of each of the quadrants based on the same snapshot taken before. Furthermore, the throughput of each of the quadrants will be compared to another threshold "τ_{sub}"; if the calculated value of a quadrant's throughput is less than τ_{sub}, the quadrant will be further divided into another four quadrants because it is the most likely quadrant to contain the faulty node. The zone master will keep repeating the division process until it reaches to a quadrant that only contains a single node. Depending on the calculated throughput and the comparison with thresholds, the node causing the throughput to be lower than the threshold is identified and is considered as a suspect node that suffers from a communication fault.

In order to decide whether the suspect node is faulty, a technique that makes use of the redundancy in sensor networks is applied. In other words, based on the information stored in the registration table, the zone master will start searching for the nearest sleeping node to the suspect. After identifying such a node the master will wake it up so that it can start sensing. After a period of time the master will calculate the throughput of the suspect node and the node it woke up based on a new snapshot of the information table. If the difference between the two values of the throughput is larger than a threshold "Δ" and the throughput of the suspect is less than that of the awakened node, another sleeping node will be awaken in order to be able to decide whether the suspect is faulty or the awakened node i.e. having a third node sensing in the same area will help to solve the conflict.

After activating two nodes, which are the nearest to the suspect node, the suspect node and the other two nodes will start sensing. After a period of time the throughput of the three nodes will be calculated and compared to Δ, if the values of the throughput of the awakened nodes are similar and their differences with the value of the suspect node is large, the master can decide that the node that was suspected to be faulty is suffering form a communication fault and will be switched off and one of the awakened nodes that is nearer to the faulty node will be kept awake and the other one will go back to sleep.

The presence of a sensing fault is detected by comparing the data received from a node to its entry in the grid table. If the data reported is within the correct range and the node has a correct or a matching status, it will be considered fault free otherwise, the master will start dividing its zone virtually until it finds the suspected node, after that the same technique that was mentioned above to test the suspect node by waking sleeping nodes up is used.

```
For each zone
   Initialize Information table for zone_master
End Loop
For Each zone_master
   If message destination = zone_master
      Update Information table of the zone_master
   End If
End Loop
Set period to 2
Set time to the result of dividing current time by period
Set threshold to 100
 Set decrement to 10
 Set i to 1 // this variable is used to control the access of the
  subzone array and make the process recursive
  If time = 0
     Calculate throughput for each zone until current time
     Set subthreshold to 50
     Set new_threshold to subthreshold
      For each zone
       If zone throughput < threshold
        Divide zone
        For each subzone
          If zone throughput < threshold
            Divide zone
           For each subzone
            If the number of nodes in the subzone >1
            Calculate throughput of the subzone
             If subzone throughput < subthreshold
               Divide the subzone
               Get division_array
// array where the subzones arrays are stored
                  While i <= 4
                       get number of nodes in subzone(i)
// the first subzone array in the division_array
                     Calculate subthroughput of subzone(i)
                     If (number of nodes > 1 and
                      (subthroughput < new_threshold – decrement))
                           divide subzone
                           set divison_array to new_array
// replace the old division_array with a new division  array resulting
//from the new division
                           If new_threshold > 10
                            Set new_threshold to new_thresold-
                             decrement
```

```
                        End If
                      Else
                       If number of nodes in the subzone =1
                        Set suspect_node to node ID in the
                          subzone
                        Find a neighboring sleeping node to
                          wake up
                        Increment i by 1
                       End If
                     End If
                   End Loop
              End If
             Else
               If number of nodes in the subzone = 1
                  Set suspect_node to node ID in the subzone
                  Find a neighboring sleeping node to wake up
               End If
             End If
          End Loop
        End If
     End Loop
End If
Get current time
Set time to the result of dividing current time by period
If time = 0
 Calculate throughput of the three nodes until current time
 Compare the values and find the faulty node
End If
```

Figure 6. Communication Fault Identification Pseudo Code.

The pseudo code in Fig.6 illustrates how a faulty node is identified. Note that, the throughput is calculated according to equation (1).

$$T = (N * L) / P \quad (1)$$

where T is the throughput to be calculated, N is the number of messages received by the master, L is the message length and P is the time period on which throughput is calculated.

```
For each entry in the grid table
 Find node ID that is equal to the message source
 If (status = high and data = high) or (status = medium and data =
 medium) or (status = low and data = low)
    Set suspect to 0
 Else
     Divide zone
     While i <= 4
        get number of nodes in subzone(i)
        If (number of nodes > 1
          Divide subzone
          Set divison_array to new_array
        Else
           Set suspect to nodeID
           Find a neighboring active node
           Incremenr i by 1
        End If
```

```
      End Loop
  End If
End Loop
Set sensing_threshold to 5
Get current time
Set time to the result of dividing current time by period
If time = 0
   Get data provided by both nodes
   Compare the data of both nodes
   If difference in readings > sensing threshold
      Find a neighboring sleeping node and wake it up
   End If
End If
Get current time
Set time to the result of dividing current time by period
If time = 0
 Get data provided by the three nodes
 Compare the data of the three nodes and find the faulty node
End If
```

Figure 7. Sensing Fault Identification Pseudo Code.

In Fig.7, the variable *status* is used to check that the data reported by the node is correct according to its status, while the variable *data* is used to check if the data reported is correct and is actually within the expected range according to the nodes status and the ranges stored in the zone master.

```
Calculate zone_center
For each node in the zone_master registration table
  Compare node position to the center
  Allocate node to a subzone according to its position
End loop
```

Figure 8. Division Procedure.

In Fig 8, the code that is used to divide the zone into subzones is illustrated. A zone is divided into subzones by calculating the zone center as the x and y axis values of the main zone which are stored in the registration table and according to the node position and the value of the zone center. Therefore, the node will be allocated to an array that represents each subzone. Note that the ID of the zone master will be known to this procedure from the code in Fig.3.

VI. SIMULATION

A. *The Simulator*

The simulator used to conduct the experiments is TrueTime 1.5 which is MATLAB/Simulink based. Its main feature is that it gives its users the ability to co-simulate the interaction between the continuous dynamics of the real world and the architecture of the computer [21], [22].

B. *Simulation Scenarios*

1) *Faults Occurring At The Zone Master Level Only*

In this scenario, faults were injected at the zone master level, i.e. sensor nodes associated with a faulty zone master were fault free and the zone master was suffering from processing fault, as a result throughput was calculated erroneously. Note that, only one fault was injected at the masters level as the algorithm can detect one fault only. In this scenario, the faulty master was in different level in the diagnosis tree; in one case it was a leaf node in tree A. In another case, it was a non leaf node in tree A. Thus, tree B was built in order to detect the fault in such case.

This scenario is proposed to show the ability of the algorithm to detect faults at the zone masters level occurring at different levels in the diagnosis tree.

2) Faults Occurring In One Zone Only

In this scenario, all the active nodes in the network will be providing data to their masters. However, only nodes belonging to one zone will be suffering from faults as a result, the technique to identify and locate faulty nodes will be initiated in that zone only. In addition, the faulty nodes in that zone will be in different positions within the zone, which means that when the division process is started the faulty nodes will be in different quadrants or subzone and each quadrant may contain more than one faulty node. Also, a chemical spill will occur and affect nodes in this zone only.

This scenario was proposed to show the ability our technique to divide more than one quadrant into different levels until the faulty node is identified and replaced. Figure 9 illustrates the described scenario.

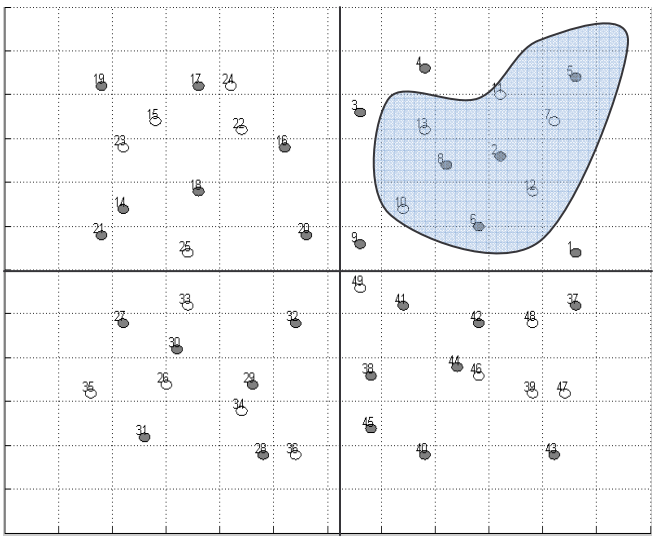


Figure 9. The Second Scenario.

Similar to the second scenario, all active nodes in the network will be providing data to their zone master but, the nodes that belong to two different zones will be faulty. The chemical spill will occur and will affect nodes in both zones.

This scenario is created to show the ability of our technique to locate and identify faulty nodes in different zones synchronously. The faulty nodes in this scenario might not be in the four quadrants of each zone when the division starts. In other words, after the zone is divided into quadrants, some quadrant may provide values of the throughput higher than the threshold mentioned in section V., while other will have throughput value lower than that threshold which indicates that there is a problem in that quadrant. Figure 10 illustrates the scenario.

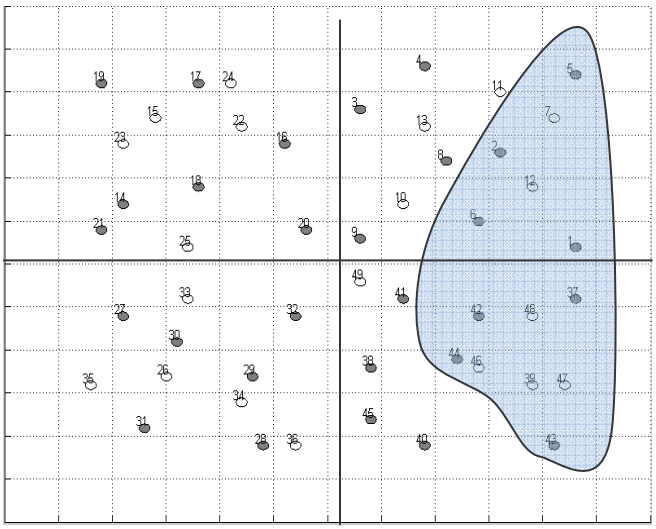


Figure 10. The Third Scenario.

C. Simulation Results

The scenarios studied were based on a network consisting of 50 nodes deployed randomly in 60x60 units region. The performance of the two fault diagnosis algorithms described in this paper was not compared because, they work at different levels in the network.

The distributed fault diagnosis algorithm used to detect fault zone masters is evaluated according to the number of messages required to detect the faulty master. Table 4 shows the number of messages required by the first algorithm depending on the level at which the faulty node occurs in the diagnosis tree.

Table 4. Number of Messages for the First Algorithm.

Case	Number of Messages
1	4
2	6
3	10
4	12

Cases 1 and 2 in table 4 represent the cases where the faulty nodes were the leaf nodes in the first test tree, while the remaining two cases are gained when we have to build the second test tree. It can be observed that the distributed

diagnosis algorithm requires a small number of messages to be exchanged between the masters; because these master nodes are more rich in energy they can afford to send a small number of messages to accomplish the diagnosis process.

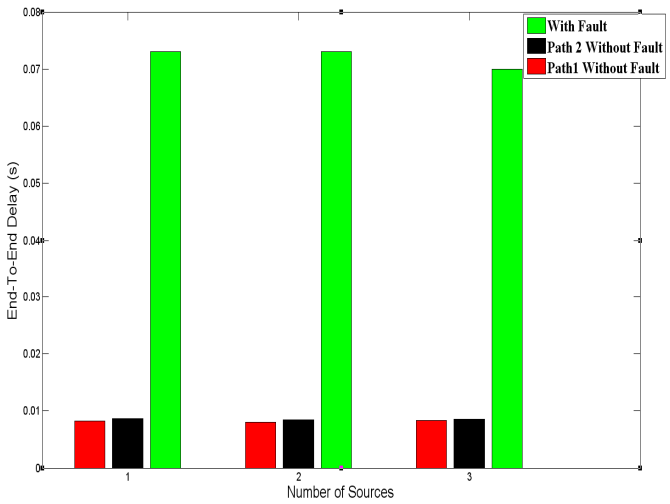


Figure 11.Comparison of the Performance of De Bruijn Network.

Figure 11 shows the performance, in terms of average end-to-end delay for 4-node De Bruijn based network. It can be observed that both paths have similar delay under fault free conditions. However, when a fault was injected, the nodes in the network had to switch between path 1 and path 2 to detour around the faulty node, which caused the average end-to-end delay to increase. Not that, these results were gained by selecting random source and random destinations and the result for each case was obtained by averaging the results of 10 runs.

In the simulation, sensor nodes were faulty nodes were randomly chosen and the technique was tested with the following number of faulty nodes 2, 4, 6 and 8. In addition, the performance of the technique presented in this paper was compared, in terms of detection accuracy, to that of the Randomized Decision Scheme (RDS) presented in [11], where the detection accuracy can be defined as the percentage of the number sensor nodes that are detected to be faulty by a technique to the total number of faulty nodes the WSN [9].

Figure 12 shows the detection accuracy with respect to the number of faulty nodes. From Fig 8, it can be observed that as the number of faulty nodes increases, the detection accuracy decreases. This can be regarded to the ratio of neighboring sleeping nodes to the suspected node because the studied technique depends on awaking two nodes for every suspect node. As a result it can be inferred that the higher the redundancy of the network, the better the performance of our technique. In some cases, when there are not enough sleeping nodes near the suspect, the technique will awake the first sleeping node that is the nearest to the suspect but, because of not having enough sleeping node, the awakened node could be a bit far from the suspect and may not be under the same conditions as a result, not providing similar readings.

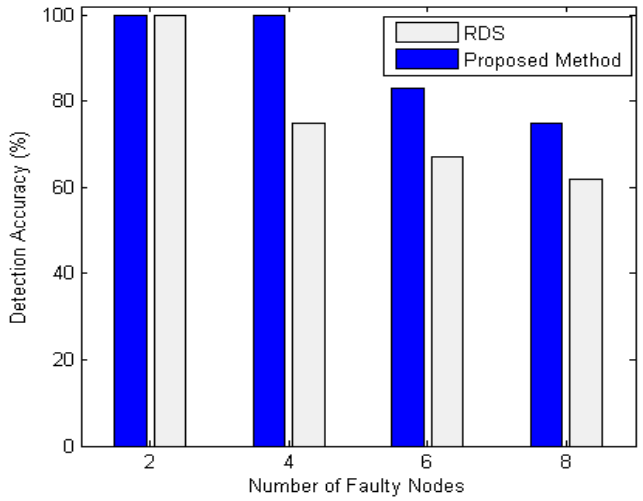


Figure 12. Comparison of Communication Faults Detection Accuracy.

Figure 13 shows the detection accuracy of our technique when having sensor nodes suffering from a sensing fault. It can be observed that when the number of faulty nodes was increased, the detection accuracy decreased because of not having enough fault free nodes near the faulty nodes.

Our technique has shown better performance than that of RDS, because in RDS the threshold value is selected randomly. As a result, in some cases the threshold value was suitable to help RDS detect fault node. However, in other cases this value was not suitable to be used in the detection which results in reducing the accuracy of fault detection in RDS.

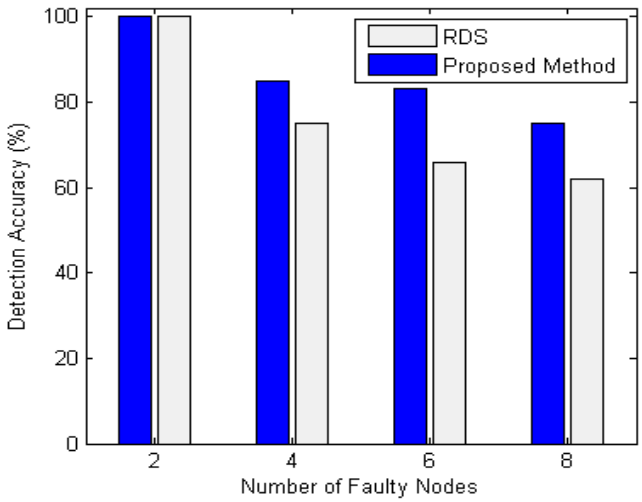


Figure 13. Comparison of Sensing Faults Detection Accuracy.

VII. CONCLUSION

In this paper, we proposed a new technique, consisting of two algorithms, to identify and substitute faulty nodes in wireless sensor networks. The proposed technique divides the network into four zones while having a master node for each zone. The first algorithm proposed in this paper is used

to diagnose faulty zone masters. On the other hand, the second algorithm is used to test and substitute faulty sensor nodes, i.e. non master nodes, in the network.

The simulation has shown that the proposed technique does not require a lot of messages to be exchanged in order to detect the fault master node. Also, it has shown that because the master nodes are connected in a De Bruijn graph, the end-to-end delay is low under faulty and fault free conditions. Furthermore, the simulation has shown the ability of the technique to identify several faulty nodes in the same zone. Also, it has illustrated that the technique is capable of identifying more than one faulty node in more than one zone at the same time. Finally the algorithm is tested by simulating two different scenarios. Our results show that the detection accuracy was very high when the number of faulty nodes was small compared to the number of sleeping node.

Future work for this work may include studying the effect of the second algorithm on the energy consumption and the life time of the network. In addition, the effect of having the sensor nodes in a zone connected in a De Bruijn graph will be studied.

REFERENCES

[1] A.A.Taleb, D.K. Pradhan and T. Kocak, "A Technique to Identify and Substitute Faulty Nodes in Wireless Sensor Networks," Third International Conference on Sensor Technologies and Applications, 2009. SENSORCOMM '09., pp.346-351, 18-23 June 2009.

[2] A. Mainwaring, Culler, D. Culler, J. Polastre, R. Szewczyk and J. Anderson , “Wireless Sensor Networks for Habitat Monitoring,” First ACM Workshop on Wireless Sensor Nerworks and Applications, Atlanta, Georgia, USA, Sept 2002.

[3] Y. Zou and K. Chakrabarty, “A Distributed Coverage and connectivity Centric Technique for Selecting Active Nodes in Wireless Sensor Networks,” on IEEE Trans. on Computers, vol. 54, no. 8, pp. 978-991, Aug 2005.

[4] K. Ssu, C. Chou, H. C. Jiau and W. Hu,“Detection and diagnosis of data inconsistency failures in wireless sensor networks,”in The Int. Journal of Computer and Telecommunications Networking, vol. 50,no. 9, pp. 1247-1260, June 2006.

[5] G. Gupta and M. Younis, "Fault-tolerant clustering of wireless sensor networks," *IEEE Conf.* on *Wireless Communications and Networking, (WCNC).* pp.1579-1584, March 2003

[6] Q. Liang, “Fault-tolerant and energy efficient wireless sensor networks: a cross-layer approach,” in Proceedings of IEEE Military Communications Conference (MILCOM ’05), vol. 3, pp. 1862–1868, Atlatnic City, NJ, USA, October 2005.

[7] A. Sheth, C. Hartung and R. Han (1999). A decentralized fault diagnosis system for wireless sensor networks. In: Proceedings of the 2nd IEEE International Conference on Mobile Ad-Hoc and Sensor Systems (MASS) 2005. pp. 192–194.

[8] I. Saleh, H. El-Sayed and M. Eltoweissy, "A Fault Tolerance Management Framework for Wireless Sensor Networks," Innovations in Information Technology, 2006 , vol., no., pp.1-5, Nov. 2006.

[9] J. Chen, S. Kher, and A. Somani “Distributed Fault Detection of Wireless Sensor Networks”. In Proc of the 2006 Workshop in Dependability issues in wireless ad hoc networks and sensor networks, L.A., California, USA, Sept. 2006.

[10] J. Staddon, D. Balfanz and G. Durfee, “Efficient tracing of failed nodes in sensor networks”. In Proc of the 1st ACM international workshop on Wireless sensor networks and applications, Atlanta, Georgia, USA, Sept 2002.

[11] B. Krishnamachari, S. Iyengar, "Distributed Bayesian algorithms for fault-tolerant event region detection in wireless sensor networks," Computers, IEEE Transactions on , vol.53, no.3, pp. 241-250, March 2004.

[12] Yue-Shan Chang; Tong-Ying Juang; Chih-Jen Lo; Ming-Tsung Hsu; Jiun-Hua Huang, "Fault Estimation and Fault Map Construction on Cluster-based Wireless Sensor Network," . *IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing, 2006*, vol.2, no., pp.14-19, 5-7 June 2006.

[13] T. Clouqueur, K.K. Saluja, P. Ramanathan, "Fault tolerance in collaborative sensor networks for target detection," *Computers, IEEE Transactions on* , vol.53, no.3, pp. 320-333, Mar 2004.

[14] Yongxuan Lai; Hong Chen, "Energy-Efficient Fault-Tolerant Mechanism for Clustered Wireless Sensor Networks," *Proceedings of 16th International Conference on Computer Communications and Networks, 2007. ICCCN 2007.*, vol., no., pp.272-277, 13-16 Aug. 2007.

[15] Tsang-Yi Wang; Han, Y.S.; P.K. Varshney, Po-Ning Chen, "Distributed fault-tolerant classification in wireless sensor networks," *IEEE Journal on Selected Areas in Communications*, vol.23, no.4, pp. 724-734, April 2005.

[16] S. Hwang and Y. Baek, ”Fault Tolerant Time Synchronization for Wireless Sensor Networks,” LNCS-3894, Springer, pp. 480–493, March 2006.

[17] F. Koushanfar; M, Potkonjak; A, Sangiovanni-Vincentell, "Fault tolerance techniques for wireless ad hoc sensor networks," Sensors, 2002. Proceedings of IEEE , vol.2, no., pp. 1491-1496 vol.2, 2002.

[18] M. R. Samatham and D. k. Pradhan, “THE De Bruijn Multiprocessor Networ: A versatile Parallel Processing and Sorting Network for VLSI,” IEEE Trans. on Computers, Vol 38, No. 4, April 1989.

[19] D. K. Pradhan and N. H. Vaidya, “Roll-Forward Checkpointing Scheme: A Novel Fault-Tolerant Architecture,” on IEEE Trans. On Computers, vol. 43, no. 10, pp. 1163-1174, Oct. 1994.

[20] D. K. Pradhan, and S. M Reddy, “A fault-tolerant communication architecture for distributed systems,” IEEE Trans. on computers., pp. 863-870, 1982.

[21] M. Andersson, D. Henriksson, A. Cervin and K. Årzén, “Simulation of wireless networked control systems”. In Proc of the 44th IEEE conference on Decision and control and European Control onference ECC, Seville, Spain 2005.

[22] TrueTime, Lund University. (2008, June 10) [Online]. Available: http:// www.control.lth.se/truetime/

Integrated System for Malicious Node Discovery and Self-destruction in Wireless Sensor Networks

Madalin Plastoi
Politehnica University of Timisoara
Timisoara, Romania
madalin.plastoi@aut.upt.ro

Ovidiu Banias
Politehnica University of Timisoara
Timisoara, Romania
ovidiu.banias@aut.upt.ro

Daniel-Ioan Curiac
Politehnica University of Timisoara
Timisoara, Romania
daniel.curiac@aut.upt.ro

Constantin Volosencu
Politehnica University of Timisoara
Timisoara, Romania
constantin.volosencu@aut.upt.ro

Roxana Tudoroiu
Politehnica University of Timisoara
Timisoara, Romania
tudoroiu.roxana@ac.upt.ro

Alexa Doboli
State University of New York
Stony Broke, New York, USA
adoboli@ece.sunysb.edu

***Abstract* — With the tremendous advances of the wireless devices technology, securing wireless sensor networks became more and more a vital but also a challenging task. In this paper we propose an integrated strategy that is meant to discover malicious nodes within a sensor network and to expel them from the network using a node self-destruction procedure. Basically, we will compare every sensor reading with its estimated values provided by two predictors: an autoregressive predictor [1] that uses past values provided by the sensor under investigation and a neural predictor that uses past values provided by adjacent nodes. In case the absolute difference between the measured and the estimated values are greater then a chosen threshold, the sensor node becomes suspicious and a decision block is activated. If this block decides that the node is malicious, a self-destruction procedure will be started.**

Keywords - wireless sensor networks, prediction, malicious node discovery, self-destruction

I. INTRODUCTION

With the continuous progress in micro-electro-mechanical systems (MEMS) and radio technologies, a new concept arose - wireless sensor networks (WSN). A wireless sensor network, being a collection of tiny sensor nodes with limited resources (limited coverage, low power, smaller memory size and low bandwidth), proves to be a viable solution to many challenging civil and military applications. Their deployment, sometimes in hostile environments, can be dangerously perturbed by any type of sensor failure or, more harmful, by malicious attacks from an opponent.

Sensor networks because of their specific limitations are susceptible to various kinds of attacks that cannot be prevented only by traditional methods (e.g. cryptography): eavesdropping, traffic analysis, selective forwarding, spoofing, wormhole attack, sinkhole attack, Sybil attack and Hello flood attack are the most significant [2]. But, almost certainly the most important danger, due to the inherent unattended characteristic of wireless sensor networks, is represented by node-capturing attack [3], where an enemy acquires full control over sensor nodes through direct physical contact. A node capturing attack is very feasible because of at least two reasons: a) practically, we cannot demand an efficient access control to thousands of nodes distributed in a large territory; and b) it is very difficult to assure tamper-resistance requirements because sensor nodes frequently need to be inexpensive to justify their use.

After an attacker gains the physical control over a sensor node he can extract secret information such as cryptographic keys to achieve unrestricted entrance to higher network levels, or by using reverse engineering techniques he can find security holes to compromise the entire sensor network.

Our proposed countermeasure is based on the fact that a corrupted node is better to be expelled from the network as soon as its malicious activity is started [4]. Even if more sensors are expelled, the WSN will function as designed because of one inherent feature: spatial redundancy [5].

In order to identify a corrupted sensor node, we presumed that even if it may still send authenticated messages (e.g., it can use the cryptographic keys already stored in its memory), it might not operate according to its original specifications sending incorrect readings to the base station. We will identify these sensors by using prediction techniques and will eliminate them by starting a self-destruction node procedure.

The rest of the paper is organized as follows. The second section presents a detailed description of our strategy. Section 3 presents the technique used for the self-destruction procedure applied to corrupted nodes. In the last two sections, experimental results and conclusions are offered.

II. PREDICTION BASED METHODOLOGY

In a large number of applications where wireless sensor networks interact with sensitive information or function in hostile unattended environments, it is crucial to develop security related mechanisms.

Due to their nature, these networks have to resist to a plethora of possible attacks. The attackers will try to obtain in-network information or to corrupt the network partially or totally. For making this possible, the attackers will try to gain control over one or several network nodes. Our proposed defending strategy is based on the detection of malicious sensor nodes using predictors and the elimination of their effects by expelling them from the network using a self-destruction node technique.

A. The Sensor Network Assumptions

In order to assure a high rank of efficiency for our malicious node detection and self-destruction strategy we chose a sensor network having the following features:

a) The sensor network is static, i.e., sensor nodes are not mobile; Moreover, each sensor node knows its own position in the field.

b) The base station, sometimes named access point, acting as a controller and as a key server, is supposed to be a laptop class device and supplied with long-term power. We also assume that the base station will not be compromised.

c) Between the three most common wireless topologies (star, mesh and cluster-tree) we chose a star topology (e.g. Cellular Wireless Network [6] and SENMA [7]) for our sensor network. Star topology is a point-to-point architecture where each sensor node communicates directly with the base station. The main characteristics of the star topology are: there are no node-to-node connections and no multi-hop data transmission; sensor synchronism is unnecessary; sensor do not listen, only transmit and only when polled for; complex protocols are avoided; dependability of individual sensors is much less significant. Because of these features, attacks on routing protocols (spoofing, selective forwarding, sinkhole attack, wormhole attack, Sybil attack and Hello flood attack) are almost impossible.

d) We rely on efficient secret-key cryptography with pre-distributed keys using Skipjack, RC5 or AES algorithms to encipher all data communications inside the sensor network; These three symmetric encryption algorithms have a common attribute that makes them an attractive alternative in case of sensor networks: they are able to encrypt short or medium size messages, like the ones send by sensors and received by base stations, in the case of limited power consumption. By using such appropriate cryptographic techniques the damaging potential of the passive attacks (eavesdropping and traffic analysis) can be ignored.

e) The measurements supplied by each sensor have an important deterministic character, rather than a strictly random (stochastic) one. In this case there exists a correlation between past values and the current one, giving us the power of prediction. As an example, the future values of temperature measurements in a location are strongly related with past and present values, so a prediction method can be applied.

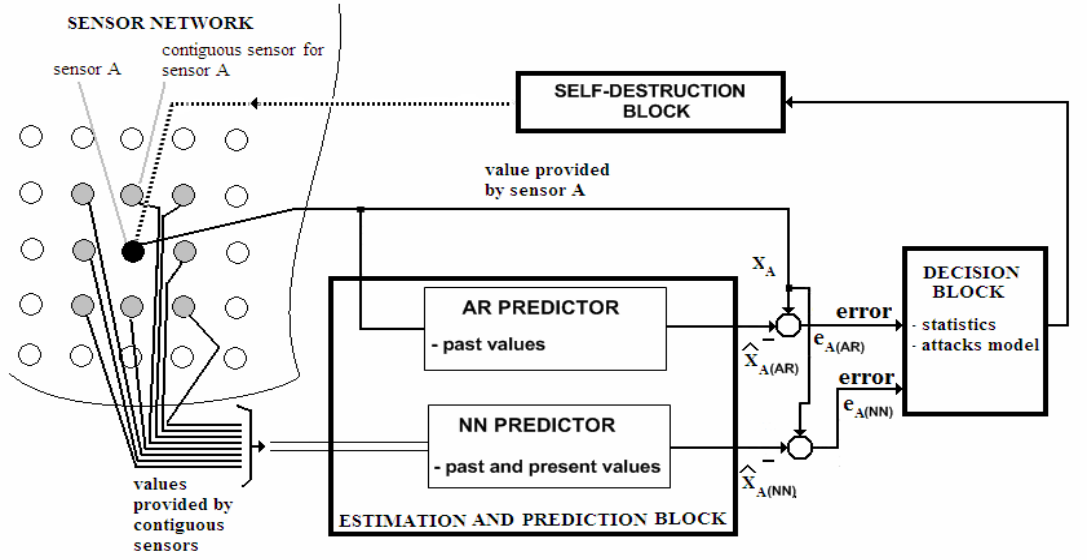


Figure 1. Malicious node discovery and self-destruction with mixed prediction

B. The Proposed Strategy

Our strategy exploits two types of observations over the current and past measurements of the wireless sensor network nodes. First, the temporal redundancy in the sense that previous provided measurements from each sensor will be used to decide if the given sensor operates as desired or not. And second, based on the past and present values of

neighboring nodes, the operation of the given sensor is classified as right or wrong.

We decided to use a mixed estimation and prediction system with two predictors – an autoregressive predictor (AR) and a neural network predictor (NN). The first predictor will describe the node evolution using its past values while the second one will describe node evolution using present and past values provided by neighboring nodes.

In Fig. 1 is presented the system architecture, composed of 4 major components: a) The Wireless Sensor Network under investigation; b) The Estimation and Prediction Block; c) The Decision Block; and d) The Self-destruction Procedure. The Estimation and Prediction Block includes two on-line predictors, one autoregressive predictor and one neural network predictor. The outputs of these two predictors represent the inputs for the Decision Block. At this level the system will provide automatic decisions upon engaging or not the self-destruction procedure for the given node.

In the following paragraphs we will present both predictors, including the mixed strategy.

Our stratagem to identify a corrupted sensor node is based on the fact that even if it may still send authenticated data it may not operate according to initial requirements, sending incorrect readings to the base station. These nodes will be identified in the moment they begin to broadcast erroneous information. For this purpose, we will present further two types of predictors, one autoregressive predictor, and one neural network predictor. Based on these predictors we will decide if a self-destruction procedure is needed.

In order to implement the first predictor, we considered that an autoregressive (AR) model efficiently approximates the evolution in time of the measurements provided by each sensor. An autoregressive or AR model describes the evolution of a variable only using its past values. This class of systems evolves due to its "memory", generating internal dynamics, and is defined as follows:

$$x(t) = a_1 \cdot x(t-1) + ... + a_n \cdot x(t-n) + \xi(t), \quad (1)$$

where $x(t)$ is the measurement series under investigation, a_i are the autoregression coefficients, n is the order of the autoregression and ξ is assumed to be the Gaussian white noise. By convention, the time series $x(t)$ is assumed to be zero mean. If not, another term a_0 is added in the right member of equation (4). Establishing the correct model of order n is not a simple task and is influenced by the type of data measurements and by computing limitations of the base station. Reasonable values of the order n are between 3 and 6.

If the a_i coefficients are time-varying, the equation (1) can be rewritten as:

$$x(t) = a_1(t) \cdot x(t-1) + ... + a_n(t) \cdot x(t-n) + \xi(t). \quad (2)$$

The model (2) can be used either to estimate the coefficients $a_i(t)$ in case the time series $x(t),...,x(t-n)$ are known (recursive parameter estimation), either to predict future value in case that $a_i(t)$ coefficients and past values $x(t-1),...,x(t-n)$ are known (AR prediction).

In parallel with the AR predictor, we use a three order feed forward neural network predictor with two hidden layers. The input layer consists of neurons which are associated with values provided by adjacent nodes at moments *t*, *t-1* and *t-2*. The output layer has only one neuron for the estimated value.

For network's node A, the estimated value by the neural predictor is:

$$\hat{x}_{A(NN)}(t) = f(X_{A,adj}(t-1),...,X_{A,adj}(t-n), B_{A(NN)}), \quad (3)$$

$$X_{A,adj}(t-i) = \left(x_{A,adj1}(t-i),...,x_{A,adjm}(t-i)\right)^T, \quad (4)$$

$$B_{A(NN)} = \left(b_{A,adj1},...,b_{A,adjm}\right)^T, \quad (5)$$

where: $x_{A,adjk}(t-i)$ is the value from the adjacent node k at $t-i$ moment, $B_{A(NN)}$ is a vector containing the trust factors of each of the *m* adjacent nodes of A, and *n* is the predictor order.

The trust factor $b_{A(NN)}$ has a linear dependence on the previous node trust factor and the error value. However, the computation of $b_{A(NN)}$ is made in the decisional block.

After the node's output of the neural prediction, an error value is obtained:

$$e_{A(NN)}(t) = \left| x_A(t) - \hat{x}_{A(NN)}(t) \right|. \quad (6)$$

C. The Autoregresive Prediction

Our strategy exploits the temporal redundancy in the sense that previous provided measurements from each sensor will be used to decide if the sensor operates as desired or not. The plan is the following: an attacked sensor node that will attempt to insert false information into the sensor network will be recognized by comparing its output value with the value predicted using past readings offered by that specific sensor (Fig. 2). In the case that any malicious activity is observed, the Decision Block is triggered to decide if the self-destruction procedure must be started for this specific node in order to prevent its further undesired activity.

The complete mechanism workflow runs while network is active. By considering a specific node symbolized by A, this process is done in the following steps:

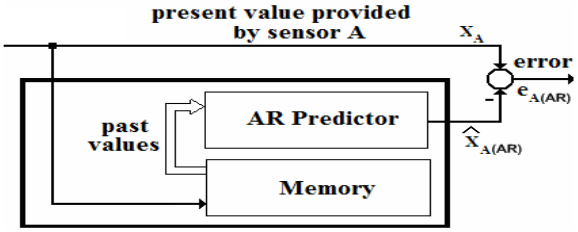


Figure 2. Autoregressive predictor

a) Associate a node trust indicator with every sensor node. The specified sensor node A will have a trust factor denoted by $b_{A(AR)}$. This integer value is initially set to zero ($b_{A(AR)} = 0$ for a fully reliable sensor node) and is incremented during our methodology every time a potential malicious activity is encountered. This trust factor must be reset to zero, if no potential malicious activity is encountered for a long period of time. The node trust indicator represents, in other words, the perception of confidence between the WSN and that specific sensor.

b) At every moment of time *t*, estimate the present value $\hat{x}_{A(AR)}(t)$ provided by sensor node A, using the past readings $x_{A(AR)}(t-i)$ provided by the same sensor A. For the sensor A, we can write:

$$\hat{x}_{A(AR)}(t) = f(x_{A(AR)}(t-1),...,x_{A(AR)}(t-n)) \quad (7)$$

where n is the estimator's order. In our approach, an on-line AR predictor performs this step.

c) Compare the present value $x_A(t)$ measured by the sensor node A with its estimated value $\hat{x}_{A(AR)}(t)$ by calculating the error:

$$e_{A(AR)}(t) = x_A(t) - \hat{x}_{A(AR)}(t) \quad (8)$$

d) Increment the node trust indicator b_A if the error $e_{A(AR)}(t)$ exceeds a given threshold ε_{AR} : $|e_A(AR)(t)| > \varepsilon_{AR}$ (this is done only one time inside a transitory time zone – due to the internal structure of recursive predictors, an error obtained at instant time *t*, is propagated/attenuated in the recursive predictor response for some instants, obtaining a transitory regime). If the node trust indicator is higher than a chosen value γ $(b_{A(AR)} > \gamma)$, the node could be declared as potentially malicious and depending on the Decision Block output, a self-destruction procedure could be started for the A node.

The associated pseudocode is presented in Fig. 3.

```
//this function is performed for each node in the network
int Autoregression(int nodeId)
{
SET b_A(AR) , ε_A(AR) ; //node trust indicator, threshold
WHILE (network is active)
{
…
x_A =READ sensor A; //get sensor actual value for node ID equal
                    //with nodeID
…
x̂_A(AR) = ARpredict(prior x_A values); //call AR prediction
e_A(AR) = x_A − x̂_A(AR) ; //calculate the error
IF (ABS ( e_A(AR) ) > ε_A(AR) )
{
  IF (AR predictor is not in transitory regime)
  { b_A(AR) = b_A(AR) + 1; //increment node trust indicator
    START thread TRANSITORY_REGIME;
          //a counter set on k //and will be decremented every
          //instant until it becomes zero
  }
  DECISION_BLOCK (node with node ID equal to nodeID);
          //call //decision method
}}
…
}
```
Figure 3. AR implementation pseudocode

First of all, we have to associate a threshold $\varepsilon_{AR} > 0$ for every sensor node. This threshold will be used to decide if a sensor operates normal or abnormal and its measured value depends on the type of the sensor and its specific and desired operation in real environment. For a specific sensor A, the threshold will be denoted by $\varepsilon_{A(AR)}$.

After this initialization, at every instant *t*, we will compute the estimated value $\hat{x}_{A(AR)}(t)$ relying only on past the values $x_A(t-1),...,x_A(0)$ and we will use both parameter estimation and prediction as presented in the following steps:

First step: we will estimate the parameters $a_i(t)$ using a recursive parameter estimation method. From a large number of methods for estimating AR coefficients we decided to use a numerically robust RLS (recursive least square) variant based on orthogonal triangularization, known in literature as QRD-RLS [8]. One of the reasons is that it can be implemented efficiently on the base stations level (laptop class device).

Second step: we will obtain the prediction value $\hat{x}_{A(AR)}(t)$ using the following equation:

$$\hat{x}_{A(AR)}(t) = a_1(t) \cdot x_A(t-1) + ... + a_n(t) \cdot x_A(t-n) + \xi(t) \quad (9)$$

The corresponding pseudocode for implementing the estimation procedure is presented in Fig. 4:

```
float ARpredict(prior x_A values)
{ CALCULATE autoregression coefficients a_i ;
   // an estimation using QRD-RLS method
  CALCULATE predicted value x̂_A(AR) ;
  //compute sensor predicted value as a result of (9)
  RETURN x̂_A(AR) ; }
```

Figure 4. Autoregressive prediction pseudocode

Following, we will compare the present value $x_A(t)$ measured by the sensor node with its estimated value $\hat{x}_{A(AR)}(t)$, and the error $e_{A(AR)}$ will be computed using equation (8).

D. The neural network prediction

In order to improve the efficiency and reliability of the Estimation and Prediction Block, we implemented a neural network predictor based on measurements provided by contiguous sensors [9]. This predictor will work in parallel with the autoregressive predictor to discover all malicious activity. In Fig. 5 the neural network predictor is presented:

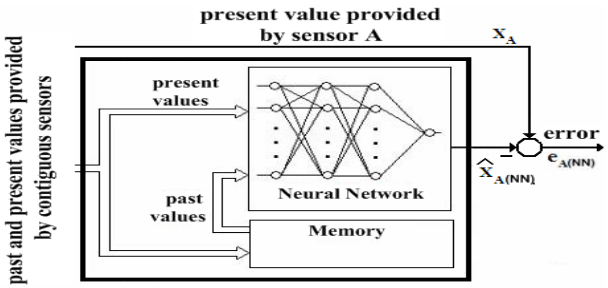


Figure 5. Neural Network predictor

A neural network predictor consists of three or more layers of artificial neurons (Fig. 6). Usually, neural networks have at least three layers, an input layer, an output layer and a hidden one. Our implemented neural network has four layers because we used two hidden layers. Neurons are linked one with each other through quantitative relations known as weights. Each layer is activated by an activation function. Most used activation functions are: direct identity (for input layers), sigmoid, hyperbolic tangent and linear functions. In fact this neural predictor model exposes a composition function which describes how the inputs are transformed into outputs.

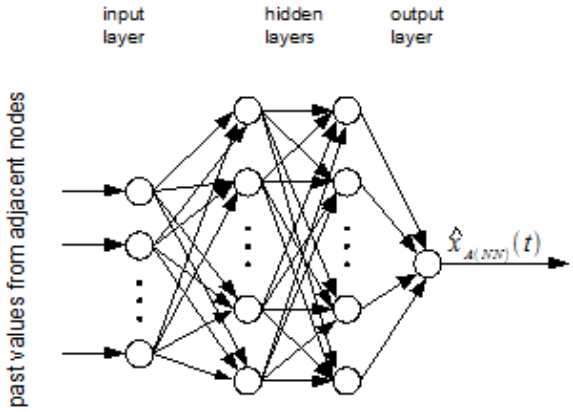


Figure 6. Neural network predictor

Each neuron implements the following function:

$$f(x) = K(\sum_i \omega_i g_i(x)), \quad (10)$$

where K is a composition function, ω_i are the weights, and g_i is a vector containing neuron's inputs $g = (g_1, g_2, ..., g_n)$ [10].

In order to use a neural network predictor, two steps must be performed:

Neural network training - will be established the number of hidden neurons and the neural network weights ω_i by performing successive training sessions using Levenberg-Marquardt method [11]. For the hidden layers we will use hyperbolic tangent activation function and for the output layer we will use a linear activation function. This training step is done off-line, prior to the neural network predictor implementation on the base station.

Neural network on-line prediction - will be obtained the prediction value $\hat{x}_{A(NN)}(t)$, computing for each neuron the equation (10). This procedure starts with the neurons from the input layer and ends with the neuron from the output layer and it's implemented as an on-line predictor on the base station. In the end, the error $e_{A(NN)}$ is computed using equation (6). The associated pseudocode for the neural network predictor is presented in Fig. 7:

```
//this function is performed for each node in the network
int NNPrediction(int nodeId)
{
SET b_A(NN), ε_A(NN) ; //node trust indicator, threshold
WHILE (network is active)
{
 …
x_A =READ sensor A; //get sensor actual value for node ID equal
//with nodeID
 …
```

```
x̂A(NN) = NNpredict (present and past adjacent nodes values);
    //call NN prediction
eA(NN) = xA − x̂A(NN) ; //calculate NN error
IF (ABS ( eA(NN) ) > εA(NN) )
  IF (NN predictor is not in transitory regime)
  { bA(NN) = bA(NN) + 1  //increment node trust indicator
    START thread TRANSITORY_REGIME;
          //a counter set on k //and will be decremented every
          //instant until it becomes zero
  }
 DECISION_BLOCK (node with node ID equal to nodeID);
    //call decision method; Fig. 8.
}
…
}
```

Figure 7. NN implementation pseudocode

The initialization steps are the same as for the AR prediction, a threshold $\varepsilon_{A(NN)} > 0$ being associated with sensor node A.

At every instant *t*, we will compute the estimated value $\hat{x}_{A(NN)}(t)$ relying on the present and past values of the adjacent neighbors of the node A.

E. The Decision Block

The Decision Block will provide the decision to start the self-destruction procedure of a malicious node, based on the pair of errors, ($e_{A(AR)}(t), e_{A(NN)}(t)$) that will be used as inputs in an expert system, by computing two trust indicators $b_{A(AR)}$ and $b_{A(NN)}$.

These trust indicators are initially set to zero $(b_{A(AR)} = 0, b_{A(NN)} = 0)$ signifying that in the beginning, the sensor is considered to be fully reliable. After that, when potential malicious activity is detected, the trust indicators are incremented. In order to filter possible sporadic malfunctions of the sensors, these counters must be reset to zero at specific intervals in time.

An efficient Decision Block can be implemented either as a fuzzy-based system, either as a rule-based system. In this paper we have chosen a rule-based approach. The Decision Block architecture is presented in the Fig. 8:

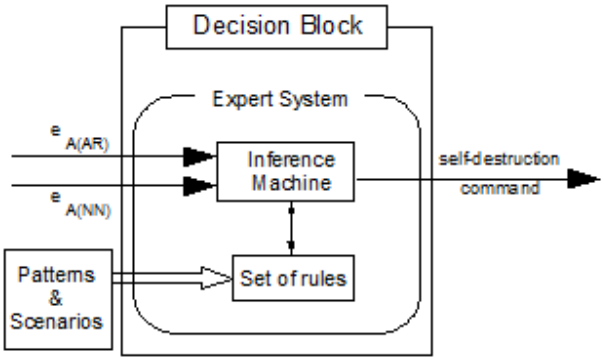


Figure 8. Decision Block

Based on the previously defined patterns & scenarios, the decision block was implemented as an expert system. We defined two parameters α, β , with $0 < \alpha < \beta$, for classifying each of the trust factors $b_{A(AR)}, b_{A(NN)} \in \mathrm{N}$, as follows:

a) 5 categories for $b_{A(AR)}$

$$b_{A(AR)} = 0; b_{A(AR)} \in (0, \alpha); b_{A(AR)} = \alpha; \quad b_{A(AR)} \in (\alpha, \beta); b_{A(AR)} = \beta \tag{11}$$

b) 5 categories for $b_{A(NN)}$

$$b_{A(NN)} = 0; b_{A(NN)} \in (0, \alpha); b_{A(NN)} = \alpha; \quad b_{A(NN)} \in (\alpha, \beta); b_{A(NN)} = \beta \tag{12}$$

Using this classification, we developed 25 types of rules for the pair $(b_{A(AR)}, b_{A(NN)})$ having the inputs $b_{A(AR)}$, $b_{A(NN)}$, α , β , and the outputs the activation or not of self-destruction procedure. The rules formalization is presented in Fig. 9:

```
RULE X:  // activation of self-destruction procedure

 if Evaluate( bA(AR) , bA(NN) , α , β ) then
     Self_Destruction( sensor A);

RULE Y:  // not activation of self-destruction procedure

 if Evaluate( bA(AR) , bA(NN) , α , β ) then
     DoNothing();
```

Figure 9. Rule definition

As presented in Fig. 9, the activation of certain rules lead to the self-destruction procedure of the given node (Self_Destruction procedure). This procedure is presented in the following paragraph.

III. MALICIOUS NODE SELF-DESTRUCTION

If a certain node has been tagged as malicious, the base station will initiate a self-destruction sequence for that specific node. The self-destruction routine is divided into several actions:

- Erase node RAM memory that contains susceptible network information, driven software and cryptographic keys and also other additional memories (e.g. flash memory, if present);
- Drain node battery in different ways like R/T radio flood or node logical unit infinite cycle [12];
- Destroy node radio device;
- Delete node unique identifier from the lists of each of the neighbor nodes, including base station; This way an already captured node won't gain authentication rights if an attacker tries to reintroduce it in the network (will disable auto-organization property);
- Mask node measurement nature by hiding the type of the sensor that has been used (each node has one or more sensors and knows in a logical way which of them is used for measurements).

The above actions have to be performed in order of their importance, although some kind of concurrency could be assured. For example: the initiation of self-destruction could start the procedure for draining node battery, but in the same time it could conduct erasing actions for memory and cryptographic keys.

Self-destruction should take into consideration all network characteristics from design to deployment including the topology. Also it strongly depends on the node hardware profile. For the proposed star network model, self-destruction will imply only the base station and the compromised node – as we stated earlier, each node communicates directly with the base station.

Basically, self-destructive sequence may be a software routine embedded into node's memory or sent bit by bit from the base station to the aimed node. Entire code has to be compatible with nodes and base station operating system. The pseudocode for node self-destruction is presented in Fig. 10:

```
void Self_Destruction (sensor A)
 WHILE (sensor A is in network)
 {
  START thread
   CONSUME battery energy //broadcast specific messages;
  START thread
  {ERASE node memory; //erase RAM and flash memory
    DISABLE auto-organization property;
              //delete node identifier from all neighbor's lists
    DESTROY node radio device;
    MASK node measurement nature;
              //for hiding the type of the sensor
 }}
```

Figure 10. Node self-destruction pseudocode

In the optimistic scenario, after self-destructive routine was initiated, the intended node is destroyed and obvious, undetectable in the network. All references to its identifier will disappear from all network devices. Cryptographic keys and stored information will be also deleted.

In the pessimistic scenario, the self-destructive response won't have any triggered action attached to it; the corrupted node will still be alive in a fully or partially functional state. This scenario will have to be avoided by reducing the probability of some unfortunately events like:

1. Self-destructive routine was not suitable implemented for node hardware profile;
2. The node's software is modified by the enemy in such way that it doesn't accept incoming messages from the base station, it only sends malicious data;
3. Node battery is almost exhausted at the moment of executing the routine. In this case no memory erasing will be performed. The attacker will replace the batteries and the node will be partially or totally running.

Our solutions for avoiding these incidents are: testing of the self-destruction code on every type of sensor nodes that is included in the WSN; hiding the self-destruction routine into the node's memory; paying attention to the energy consumption on each sensor.

IV. CASE STUDY

For validating our strategy, we used a Crossbow wireless sensor networks, containing 15 Mica2 nodes and one base station in a star topology. The sensors were set to measure the temperature in a room. In order to model an attack upon sensor A, we made the following experiments:

- First case study: at specific moments in time we intervened with an electric incandescent lamp (200 watts) placed very near to sensor A, for disrupting the normal functioning of only one specific node;
- Second case study: at specific moments in time we placed a heat source near to the network. In this case, normal functioning was disrupted for several network nodes, including the sensor node A.

We set up both predictors with the order $n = 3$, the thresholds $\varepsilon_{A(AR)} = 1^{\circ}C$, $\varepsilon_{A(NN)} = 2^{\circ}C$ and the pair $(\alpha, \beta) = (3, 5)$.

We considered that the node A has 8 adjacent neighbors. Past and present values from these adjacent neighbors will be used for neural network prediction of the node A value at a certain moment in time, while past values of node A will be used for autoregressive prediction.

The predictor contains 4 layers: one for input with 24 neurons, two hidden layers and one output layer for the estimated value. The hidden layers have 48 and 24 neurons.

The neural network predictor was trained using three arrays of inputs corresponding to the 8 adjacent neighbors at

three moments of time (predictor order) and a target value for node A (Fig. 11). The training was performed offline, using the Matlab toolbox functions and features (e.g. train function, nntool UI) [13].

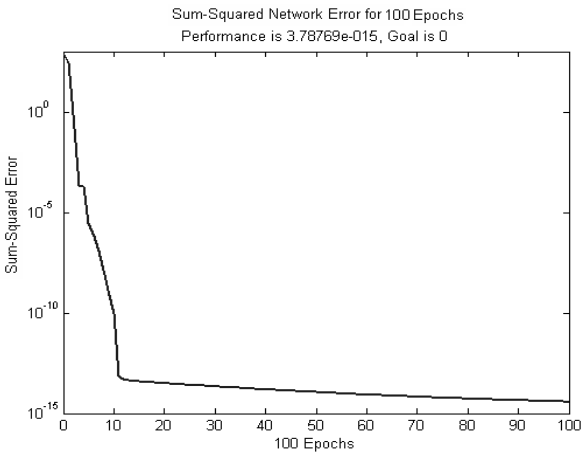


Figure 11. Neural network training session

After the training step was finished, the neural network predictor was installed on the base station node.

A. First case study

In this case study, we simulated an attack over a certain node, by inserting a heated lamp in its close neighborhood, leaving the other nodes unaffected (Fig. 12). We observed the result of both predictors and the system's decision making.

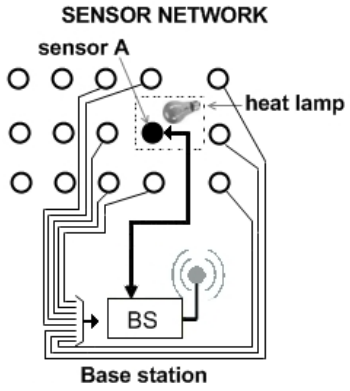


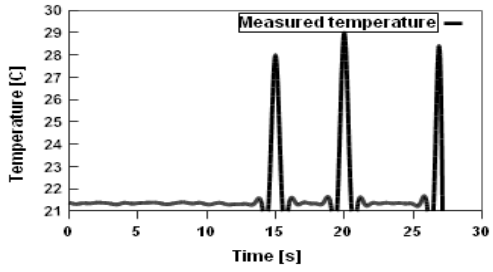
Figure 12. First case study description

In Fig. 13a we presented the sensor's A output time series, including our three "malicious" interventions at instants t=15sec, t=20 sec and t=27 sec. In Fig. 13b and 13c we presented the time variation of the AR and NN predicted time series, and in Fig. 13d and 13e we presented the evolution in time of the errors $e_{A(AR)}(t)$ and $e_{A(NN)}(t)$.

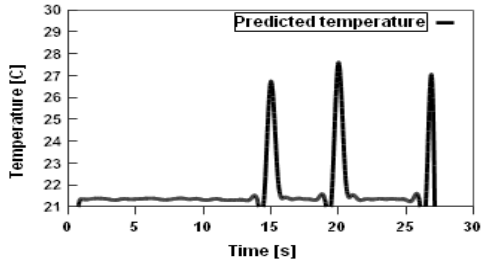
As node A is the only attacked node, the NN predictor estimated value remains in the limits of the adjacent nodes real values, while the AR predictor estimated value is in the vicinity of the attacked node real values, $e_{A(NN)}(t) > e_{A(AR)}(t)$, $t \in \{15,20,27\}$. From the graphics depicted in the Fig. 13d and 13e, we can observe that the trust factors $b_{A(AR)}$ and $b_{A(NN)}$ have the same value, $b_{A(AR)} = b_{A(NN)} = 3$. The decision to engage the self-destruction procedure is made based on a predefined rule:

If ($b_{A(NN)} == \alpha$ AND $b_{A(AR)} == \alpha$)
    Self_Destruction(sensor A);

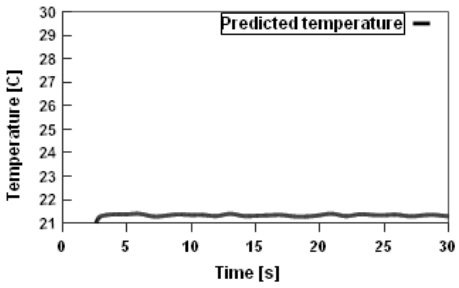
The results are as expected: after exceeding the thresholds $\varepsilon_{A(AR)}, \varepsilon_{A(NN)}$ for three times for both predictors (Fig. 13d, Fig. 13e), the sensor A is expelled from the WSN by starting its self-destruction procedure. This result can be observed in Figure 13 where no more readings are obtained from sensor A after t=27 sec.



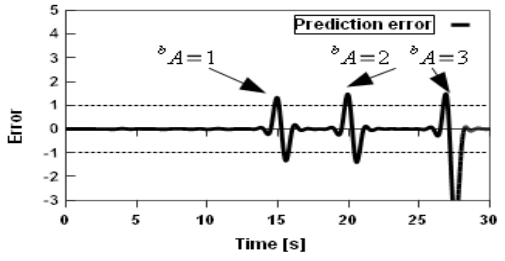
a)



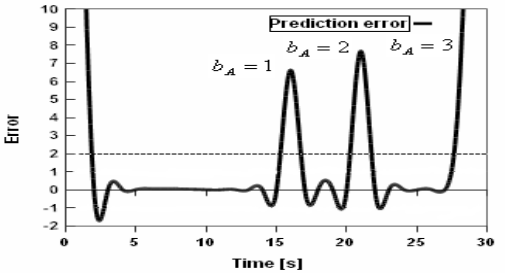
b)



c)



d)



e)

Figure 13. a)The sensor's output time series; b)AR Predicted time series; c) NN Predicted time series; d) AR Prediction error time series; and e) NN Prediction error time series;

B. Second case study

In this case study, a heat source is activated at time moments t=15sec, t=20 sec, and t=27 sec in order to affect all sensor nodes, which represent a normal operating scenario - it is not a simulation of an attack. All network nodes are affected by a hot air wave.

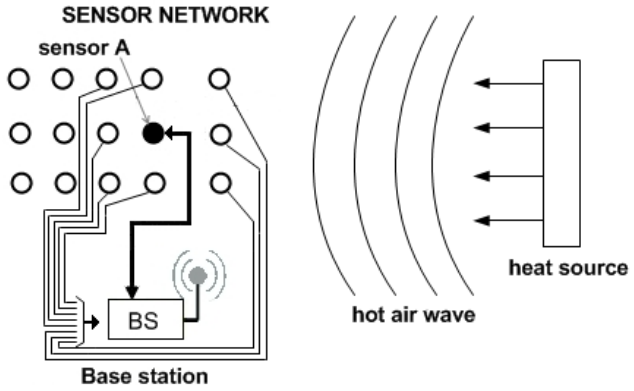


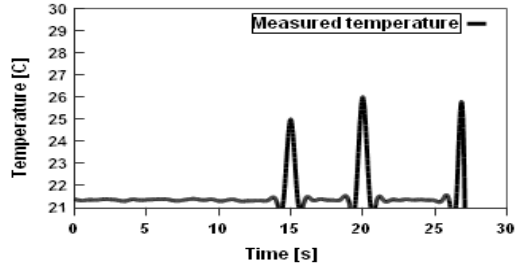
Figure 14. Second case study description

The evolution in time of the AR predicted values and the evolution of the error $e_{A(AR)}(t)$ are similar to the previous case study.

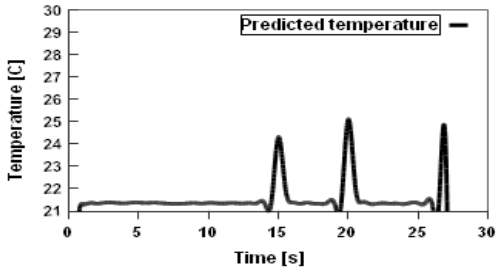
Assuming that node A has 8 adjacent neighbors, since all these neighbors are also affected by this heat wave (Fig. 14), the neural network predicted time series for the node A will have a similar evolution with the AR predictor time series (Fig. 15b, Fig. 15c). The computed error for the NN predictor case is lower than the computed error for the AR predictor case, $e_{A(NN)}(t) < e_{A(AR)}(t)$, $t \in \{15,20,27\}$. Also the error $e_{A(NN)}(t)$ is lower than the threshold $\varepsilon_A = 2^{\circ}C$. The decision to engage or not the self-destruction procedure is made based on a predefined rule:

```
if ( b_A(NN) == 0 AND b_A(AR) == α )
    DoNothing();
```

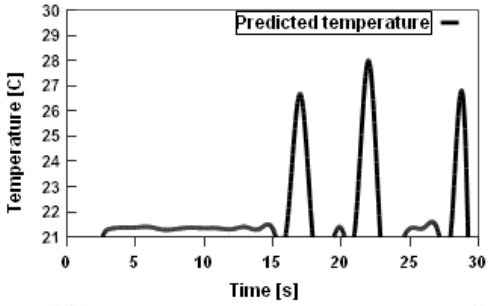
By activating this rule, the Decision Block decided that no attack was performed over the node A.



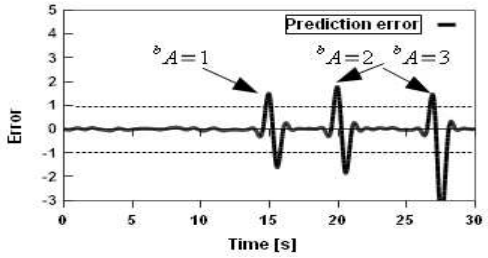
a)



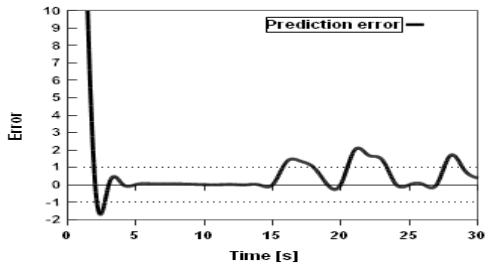
b)



c)



d)



e)

Figure 15. a)The sensor's output time series; b)AR Predicted time series; c) NN Predicted time series; d) AR Prediction error time series; and e) NN Prediction error time series;

Due to the diversity of the attack patterns we proved that the use of the two predictors in parallel is more accurate than the usage of a single predictor. For example, the use of only one predictor (AR predictor) in the second case study can lead to a wrong result – the self destruction of a normal-functioning node.

V. CONCLUSIONS

Security issues related to WSN become more and more an important research area. Detecting abnormal/malicious operation of motes and offering efficient countermeasures represents a difficult task. In this paper we propose a combined strategy that not only detects the corrupted nodes, but also excludes their malicious activity using a self-destruction node technique. Due to the inherent spatial redundancy feature of WSN, applying a self-destruction procedure to the corrupted nodes has no major inconveniences, extending the secure operation of the entire sensor network.

The integrated system presented in this paper has some noticeable advantages: decision for node self-destruction is taken based on two predictions, two errors and two trust factors, therefore the accuracy is better; the whole system takes into consideration not only the evolution of a specific node, but also its neighbors evolution; as the two predictor models have a different nature, different attack patterns could be treated. Our strategy has also a drawback: a bigger computational power is needed at the base station level.

ACKNOWLEDGMENT

This work was developed in the frame of PNII-IDEI-PCE-ID923-2009 CNCSIS - UEFISCSU grant and was partially supported by the strategic grant POSDRU 6/1.5/S/13-2008 of the Ministry of Labor, Family and Social Protection, Romania, co-financed by the European Social Fund – Investing in People.

REFERENCES

[1] D. Curiac, M. Plastoi, O. Banias, C. Volosencu, R. Tudoroiu, A. Doboli, "Combined Malicious Node Discovery and Self-Destruction Technique for Wireless Sensor Networks", Third International Conference on Sensor Technologies and Applications, Athens, Greece, June 18-23, pp. 436-441.

[2] Karlof C., Wagner D., "Secure routing in wireless sensor networks: attacks and countermeasures", Proceedings of the 1st IEEE International Workshop SNPA2003, Anchorage, USA, May 2003, pp. 113-127.

[3] Becher A., Benenson Z., Dornseif M., "Tampering with motes: Real-world physical attacks on wireless sensor networks" Proceedings of the 3rd International Conference on Security in Pervasive Computing (SPC), York, UK, April 2006, pp.104-118.

[4] D.I. Curiac, O. Banias, F. Dragan, C. Volosencu, O. Dranga, "Malicious Node Detection in Wireless Sensor Networks Using an Autoregression Technique", ICNS2007, Athens,Greece, June 2007.

[5] Y. Gao, K. Wu, and Fulu Li, "Analysis on the Redundancy of Wireless Sensor Networks," ACM WSNA'03, San Diego, USA, September 2003, pp.108-114.

[6] J. Feng, F. Koushanfar and M. Potkonjak, "System-Architectures for Sensor Networks Issues, Alternatives, and Directions", Proc. ICCD'02, Freiburg, Germany, Sept. 2002, pp.226-231.

[7] L. Tong, Q. Zhao and S. Adireddy, "Sensor Networks with Mobile Agents", Proceedings IEEE 2003 MILCOM, Boston, USA, October 2003, pp.688-694.

[8] B. Haller, J. Gotze and J. Cavallaro, "Efficient Implementation of Rotation Operations for High Performance QRD-RLS Filtering", ASAP '97 Proc., 14-16 July 1997, Zurich, Switzerland, pp. 162-174.

[9] D.I. Curiac, C. Volosencu, A. Doboli, O. Dranga and T. Bednarz, "Discovery of Malicious Nodes in Wireless Sensor Networks using Neural Predictors", WSEAS Transactions on Computer Research, Issue 1, Volume 2, January 2007, pp. 38-43.

[10] B. D. Ripley, "Pattern Recognition and Neural Networks", Cambridge: Cambridge University Press, 1996, pp. 143-162.

[11] J. More "The levenberg-marquardt algorithm, implementation and theory", In G. A. Watson, editor, Numerical Analysis, Lecture Notes in Mathematics 630, Springer-Verlag, 1977.

[12] A. A. Pirzada and C. McDonald: "Kerberos assisted Authentication in Mobile Ad-hoc Networks", Proceedings of the 27th Australasia.

[13] Howard Demuth,Mark Beale,Martin Hagan "Neural Network Toolbox™ 6 User's Guide", March 2008, pp. 2-71.

A Novel Approach to Indoor Location Systems Using Propagation Models in WSNs

Gomes Gonçalo
Instituto Superior Técnico
Inesc-ID
Lisbon, Portugal
Email: gon.ls.gm@gmail.com

Sarmento Helena
Instituto Superior Técnico
Inesc-ID
Lisbon, Portugal
Email: helena.sarmento@inesc-id.pt

Abstract— **This paper describes a location system for persons and objects in an indoor environment, where wireless nodes can include sensors and provide unique identifiers. The system nodes, using ZigBee technology, can function as RFID tags, having each one a unique EPC identification number. Sensors can be associated with the wireless nodes Zigbee to create applications for home, health and traffic control.**

Existemt location systems are analyzed, with emphasis on indoor location systems.The implemented location algorithm includes a propagation model based on the wall attenuation factor together with triangulation.

A variety of tests were carried out in an indoor environment. Results demonstrate that the location system is viable, showing itself to be effective, flexible and easily adaptable to various locations.

***Keywords - Location, Propagation, Triangulation, WAF, ZigBee, Wireless*.**

I. INTRODUCTION

History shows that progress and technological development are inevitable, actually becoming a need in the globalise world of nowadays. Telecommunications are currently essential in world economy operations of any modern society, being, wireless network systems one of the greatest developments in the area of communications. An example of such a network, is a WSN (Wireless Sensor network), where spatially distributed autonomous devices, equipped with sensors, are used in environment monitoring, traffic control, healthcare, home automation, etc. RFID (Radio Frequency Identification) is also a wireless technology. The development of RFID, besides the advantages in several areas, has also been a starting point to technological evolution in location systems. There has been a rising interest in location techniques that has been motivating allot of research in this area. In location systems there are two types of scenarios: outdoor location and indoor location. Each one of these scenarios presents itself with different characteristics and challenges, being that indoor location systems are the hardest to implement given its complexity and all the factors that have to be considered. RFID and WSN can be used together. Furthermore, generations of RFID tags include sensors, and WSNs that require automatic identification mechanisms [1].

Lately, there has been a growing interest in the integration of RFID and WSN (Wireless Sensor Network) systems. These technologies can originate the development of a variety of applications, particularly in terms of location, in residential environments, corporate environments and critical infrastructures worldwide [1]. ZigBee is a wireless technology that is well suited for WSN. Besides, it can be used to build active RFID tags. The use of RFID technology with ZigBee protocol provides the study of effective, low energy consumption, and financially accessible location systems. In this paper we will explore the use of Zigbee in order to build an indoor location system. Based in this system that locates objects in indoor environments, objects can be identified and sensing variables can be measured.

It's precisely the fusion of RFID with WSN that is the starting point of this work. The goal is make use of the RFID capabilities in automatic identification of people and objects, based on a ZigBee network, to develop an indoor location system. In order to develop a location system it's necessary to study the existing location methods and algorithms, especially those that focus in indoor location. It's important to maximize the advantages offered by the use of RFID and ZigBee together. Knowing that the goal is to achieve a system that is both easily adaptable to various location environments and flexible, it's necessary to give more attention to the location methods that use electromagnetic waves propagation models.

A starting point to the development of an indoor location system is the definition of the location method. In this paper, we propose a location system that uses triangulation together with a propagation model. The main obstacles in using triangulation are: the selection of three network devices and the estimation of the distance between fixed and mobile devices. The goal in the selection of the three network devices is to have them as closest as possible to the mobile device improving the accuracy of the system. In order to estimate the distance between fixed and mobile devices, RSSI (Received Signal Strength Indicator) measures are made so that the distance can be estimated through the use of a propagation model. The use of RSSI leads to a certain degree of inaccuracy due to multipath effect, mainly caused by the indoor environment of a building as well as the distance between a transmitter and a receiver.

The paper is organized as follows. After this introduction,

section 2 presents the state of the art concerning location systems, emphasizing indoor location. The indoor propagation models are described in section 3. Section 4 presents the developed system. In section 5, the system characterization is presented, being the results discussed in section 6. Section 7 presents some conclusions.

II. Location Systems

The success of the GPS system for monitoring and locating objects in outdoor environments, encouraged the application of similar techniques to indoor environments. Unfortunately these techniques are not a valid option for indoor location [2]. These difficulties motivate much research to develop new techniques. With the enormous growth and widespread use of short range wireless technologies, such as Bluetooth for wireless personal area networks (WPAN), WIFI (802.11) for wireless local area networks (WLAN) and ZigBee for wireless sensor networks (WSN), many academic and commercial systems are based measurement of the radio signal propagation.

In [3] present a widely accepted taxonomy to classify location techniques adopted by existent location systems. This taxonomy divides the techniques in three different groups: triangulation, proximity and scenario analysis.

A. Taxonomy

1) Triangulation: Triangulation is a technique of determining the location of a object, based on geometric properties of triangles and mathematical formulation. Two different techniques of triangulation exist: lateration and angulation. Lateration uses distances and angulation uses angles to determine the position of an object in a coordinate system (figure 1).

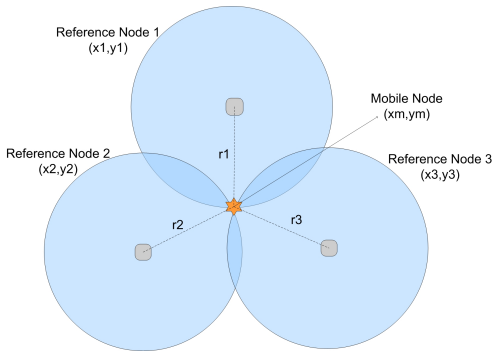


Fig. 1. Lateration

Lateration determines the position of an object by measuring its distance from multiple reference positions. In a bi-dimensional coordinate system distances to 3 non-collinear need to be measured (figure 1). Lateration systems measure distances with different techniques: direct measurement, using a physical action; time-of-arrival, measuring the time it takes a signal to travel between the object and the reference point at a known velocity; or attenuation, measuring the attenuation in the propagation of a radio signal between the object and the reference point. As the strength of radio signals is inversely proportional to the square of the distance from the source, distance is calculated based on send and receive strength.

Angulation uses angles to determine the position of an object. In a 2- dimensional Cartesian coordinate system, two angles and the distance between the reference points are used (figure 2).

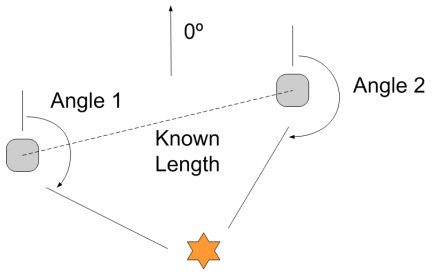


Fig. 2. Lateration

2) Proximity: Proximity based localization techniques obtain the position of an object by identifying a known reference near the object. Detection of proximity is usually achieved by: detecting physical contact with the object, using sensors; monitoring wireless cellular access points, identifying when an object is in the range of an access point; or by observing automatic identification systems. If an object is identified by an automatic system, its location can be inferred.

3) Scene Analysis: Scene analysis techniques are based on the observation of the environment where the objects are to be located. The characteristics of the environment are observed and then compared to the characteristics of a previous observation, in order to determine the location of the objects.

Static scene analysis typically consists of first phase where a pre-characterization of the environment is carried out. Obtained data are stored in a database. Observed characteristics are compared to the stored data to derive the location. Differential scene analysis tracks the difference between successive scenes to estimate location.

4) Location Fingerprinting: Location fingerprinting is a static scene analysis technique that uses radio frequency measurements to characterize the environment. Multiple radio nodes are distributed in the entire location area, usually according to a rectangular grid of points.

During the pre-characterization phase, generally called off-line phase, measurement of the received signal strength (RSS), from multiple fixed nodes, is performed. The collection of RSS values at a point on the grid is called the location fingerprint of that point. Location fingerprints are stored in a database. As RSS values fluctuate along the time, several values are measured and statistical processing is applied to build a reliable database. During the second phase, the on-line phase, the object to be located (mobile node) collects the RSS values collected from multiple fixed points. An algorithm estimates the location of the object and reports the estimated position information. The most common algorithm to estimate the location computes the Euclidean distance between the measured RSS values and the location fingerprints. Other

deterministic, such as nearest neighbor or neural networks and probabilistic algorithms based on statistical learning theory and Bayesian interference are also used.

III. Indoor Propagation Models

In radio frequency indoor location system, time and angle of arrival methods are not used because signals are affected by the multipath effect. Triangulation is implemented based on the attenuation methods, using models to relate the received power with distance.

Indoor electromagnetic waves propagation, especially inside a building, is characterised by reflections, diffractions and dispersion in the internal structures. The transmitted signals arrive to the receiver through multiple paths, originating fluctuations in the received signal. This effect, called multipath propagation, is affected by the type of materials used in the construction of the building and by the surrounding objects. Therefore, it is very difficult to predict the strength of the received signal.

There are models that take into accounty the constructive and destructive nature of multipath to relate the received power with the distance between transmitter and receiver. Rayleigh Fading Model [4] and Rician Distribution Model [5] are widely used, but they present some drawbacks. The Rayleigh Fading Model describes the small-scale rapid amplitude fluctuations in the absence of a strong received component. It assumes that all signals reaching the receivers have equal strength, which is not a realistic approach. A dominant line-of-sight (LoS) component is not accounted for by this distribution. The Rician Distribuition Model takes in account that a strong path exists in addition to the low level scattered path. This model is very appealing, but it is very difficult to determine its parameters, as this requires to physically isolate the direct wave from the scattered components [2].

A. *Wall Attenuation Factor Model (WAF)*

The WAF model [2] is quite attractive, given its ability to describe the slow fading phenomenon and the attenuation in the signal propagation introduced in indoor environments. It derives from the floor attenuation model (FAF) [6], where an attenuation factor is used to estimate the signal intensity in different floors of a building. In the WAF model the attenuation factor permits to predict the behaviour of the signal propagation, when walls are the main obstacle [2]. Equation 1 indicates how the attenuation influences the received signal strength.

$$P(d)_{[dBm]} = P(d_0)_{[dBm]} - 10n\log\left(\frac{d}{d_0}\right) - \begin{cases} nW * WAF & , nW < C \\ C * WAF & , nW \geq C \end{cases} \quad (1)$$

In equation 1, n indicates the rate at which the attenuation of the signal increases with the propagation distance, $P(d_0)$ is the received signal strength at a distance of reference d_0 and d is the distance between the transmitter and the receiver. In the attenuation factor, the C parameter accounts for the number of walls for which the attenuation factor (WAF) stops influencing the signal; nW is the number of obstructions (walls) between the transmitter and the receiver; and WAF is the value for the attenuation of each wall.

B. *The Adjusted Motley-Keenan Model*

The adjusted Motley-Keenan model [7] was developed based on the Motley-Keenan model [8]. This last model, represented by equation 2, is similar to the WAF model, but does not limit the number of walls influencing the signla attenuation. In equation 2, PL_d is the the attenuation at 1 meter of distance between transmitter and receiver, n the attenuation decay rate with distance, N the number of walls between transmitter and receiver, and k_i the number of type i walls, having attenuation of Lw_i.

$$PL(d)_{[dB]} = PL_{d[dB]} + 10n\log(d) + \sum_{i=1}^{N} k_i Lw_i \quad (2)$$

The adjusted Motley-Keenan model also considers the thickness of the walls. In equation 3, L_0 is the attenuation of a reference wall with thickness e_0, and e_i is the thickness of a wall of type i placed between the transmitter and the receiver. The adjusted term substitutes, i in equation 2, the term $\sum_{i=1}^{N} k_i Lw_i$ in order to account for the thickness of the walls originating equation 4.

$$AdjustedTerm = \sum_{i=1}^{N} k_i L_0 2^{\log_3\left(\frac{e_i}{e_0}\right)} \quad (3)$$

$$PL(d)_{[dB]} = PL_{d[dB]} + 10n\log(d) + \sum_{i=1}^{N} k_i L_0 2^{\log_3\left(\frac{e_i}{e_0}\right)} \quad (4)$$

IV. Developed System

The developed system creates a WSN with ZigBee technology to implement an indoor location system. This system was developed in order to have an automatic identification system using RFID. In the WSN one of the devices is mobile being the target of location. The other devices are placed in pre-defined positions. The final objective is to place the mobile device on a person or object in order to determine his location and identify him through WSN and RFID technology.

A. *System Architecture*

The ZigBee protocol emerges as a complement to the IEEE 802.15.4 standard, guaranteeing reliability and safety as well as a low energy consumption [14]. ZigBee using devices shall have a maximum range of 150 meters, depending this value of the environment and energy consumption of the using application.

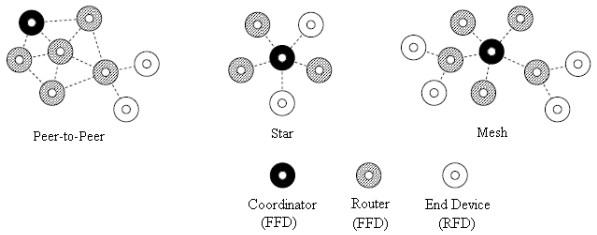


Fig. 3. Network Topology

The IEEE 802.15.4 standard defines three types of network topologies: star topology, peer-to-peer topology and mesh network [14]. Figure 3 shows the three topologies.

The star topologies require at least one FFD (Full Function Device) device functioning as network (WPAN - Wireless Personal Area Network) coordinator. The communication is established between devices and the network coordinator. The WPAN coordinator is normally powered by the electric network and the remaining devices (FFD or RFD (Reduced Function Devices)) by batteries.

In peer-to-peer topologies there also exists a WPAN coordinator, being that in this topology all devices can communicate with each other. The configuration can be found in control and industrial monitoring applications using WSNs. A mesh network is no more than a particular case of the peer-to-peer topology, where most of the devices are FFDs.

Given this, and because ZigBee distinguishes the concept of physical devices (FFD, RFD) using the notion of logical devices, we propose a network that consists of a mobile device that acts as a ZigBee Coordinator and several fixed devices acting as ZigBee End Devices. The ZigBee Coordinator is the first type of logical device, assuming a role much similar to a coordinator in the IEEE 802.15.4 coordinator and is responsible for: initiation, maintaining and manage a network. In the ZigBee hierarchy, next in line is the ZigBee End Device that is the final point in the network structure [15]. The proposed network topology is very similar to the peer-to-peer topology in the IEEE 802.15.4, where the goal is to have all the devices capable of communicate with each other.

We propose a system that is composed of a central server that executes the location algorithm as well as initiates the network. The central server can be located in the mobile device, allowing the device to locate himself, or any other location in the network being that because we are using ZigBee technology all the devices can communicate with each other.

B. Propagation Model

We use a propagation model based on the WAF propagation model [2], making the WAF parameter equal to zero (equation 5). The walls are not always an obstacle between the mobile and the fixed devices. Therefore, we adopt equation 5 instead of equation 1, but with different n and $P(d_0)$ than those used in equation 1.

Figure 1 presents an example where obstructions can exist in a certain direction but not in other directions, in the circumference. Therefore, only at the distances and in the directions where there is a presence of a wall the WAF factor is accounted for. Through this method it's possible to obtain a wide range of different WAF values for the same distances.

During the first phase, where RSS values are collected to determine n and $P(d_0)$ in equation 3, the RSS values that are affected by the presence of a wall are identified (through visual inspection of the location environment).

To those RSS values the value of the WAF factor (previously determined for respective distances and directions) will be subtracted in order to minimize the influence of the walls on the characterization of the RF signal in the location environment. This way the WAF factor is taken into account and it is reflected on the determined n and $P(d_0)$ parameters, instead of being a constant (equation 1) used at determined distances in all directions [17].

$$P(d)_{[dBm]} = P(d_0)_{[dBm]} - 10n \log\left(\frac{d}{d_0}\right) \qquad (5)$$

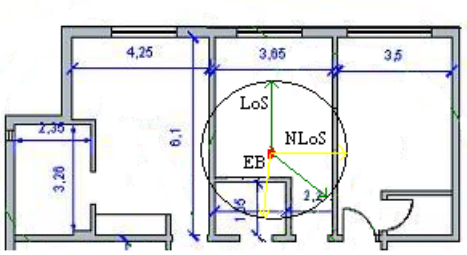


Fig. 4. Line of Sight

C. Location Algorithm

The goal for this location algorithm is to allow the location of the mobile device within certain areas of the indoor environment. The proposed algorithm is not intended for a precise location of the mobile device but to distinguish where in a certain division of a room the device is located.

Knowing that, the developed location algorithm is divided in two steps. Firstly, a propagation model is used to calculate the distances between the fixed devices and the mobile device, based on the received signal strength indicator (RSSI). This phase involves the definition of the propagation models parameters. These parameters are defined based on a previous characterization of the of RF signals in the location environment in terms, by collecting the RSS values. This characterization is much like the off-line phase used in scenario analysis methods [3]. Next, knowing the distances between the fixed and mobile devices and using a triangulation algorithm (see section D), it is possible to determine the location of the mobile device.

D. Triangulation Algorithm

To use the system, it is necessary to define the coordinate system and the position of the fixed nodes (base stations). We adopt a 2D system, assuming that all nodes are at the same height. Knowing the positions of at least three fixed

devices and their distances to the mobile device it is possible to calculate the coordinates (*x*,*y*) where the mobile device is located.

In order to estimate the three fixed devices that are closer to the mobile device, the centroid of a polygonal area as well as RSSI are used. The vertices of the polygon are fixed devices which received a beacon message from the mobile device. A preference index (FDI - Fixed Device Index) is then created so that "best" fixed devices that received the beacon messages can be chosen [16]. The FDI of FD*i* (Fixed Device) is defined by the following equation:

$$FDI_i = (1-\alpha)\left(1-\left(\frac{dist_i^c}{dist_{max}}\right)\right)+\alpha\frac{RSSI_{Ri}}{RSSI_{max}} \quad (6)$$

where $dist_c^i$ are the Euclidian distances between the centroid of a polygon and FD*i* and α is a small number $(0 \leq \alpha \leq 1)$. The values for $dist_{max}$ and $RSSI_{max}$ are defined as:

$$dist_{max} = max_{\forall i}\{dist_i^c\} \quad (7)$$

and

$$RSSI_{max} = max_{\forall i}\{RSSI_{Ri}\} \quad (8)$$

where FD*i* is an FD which received a beacon message from the mobile device. Given FDIs, the triangulation algorithm selects the three FDs with the highest FDI values for triangulation.

We implemented the Dynamic Triangular Algorithm (DTN) [10]. It is well suited for small environments it is allow complexity algorithm a requires a reduced processing time.

The DTN algorithm needs at least three sensor nodes to estimate the location of mobile device. The method discards the worst RSSI values measured by the devices and uses the best three to estimate location. It chooses the fixed device which receives the greatest RSSI value and assumes that the mobile devices location is in the mapping circle of that fixed device. The mapping circle is the estimation distance d1 between the mobile device and the closest fixed device. The DTN finds the angle θ on the mapping circle by using a cost function to pick one that best matches the observed distance. The DTN has the following steps:

1) Generation of the mapping circle: It finds the possible locations of the mobile device (x1+d1cosθ, y1+ d1sinθ) on the mapping circle by using the possible distances d2θ and d3θ between mobile user and the fixed nodes.

2) The distance of the mobile device estimation: Finds the error between estimation distances (d2 and d3) and possible distances (d2θ and d3θ).

3) The coordinates of the mobile device approximation: Determines the cost functions at each angle θ and the θ increases 1 degree each time. The DTN then searches the minimum cost function, and the θ of the minimum cost function is the estimation angle on the mapping circle. The angle θ on the mapping circle is the estimation location of the mobile device. Figure 5 describes the procedure of the DTN location algorithm.

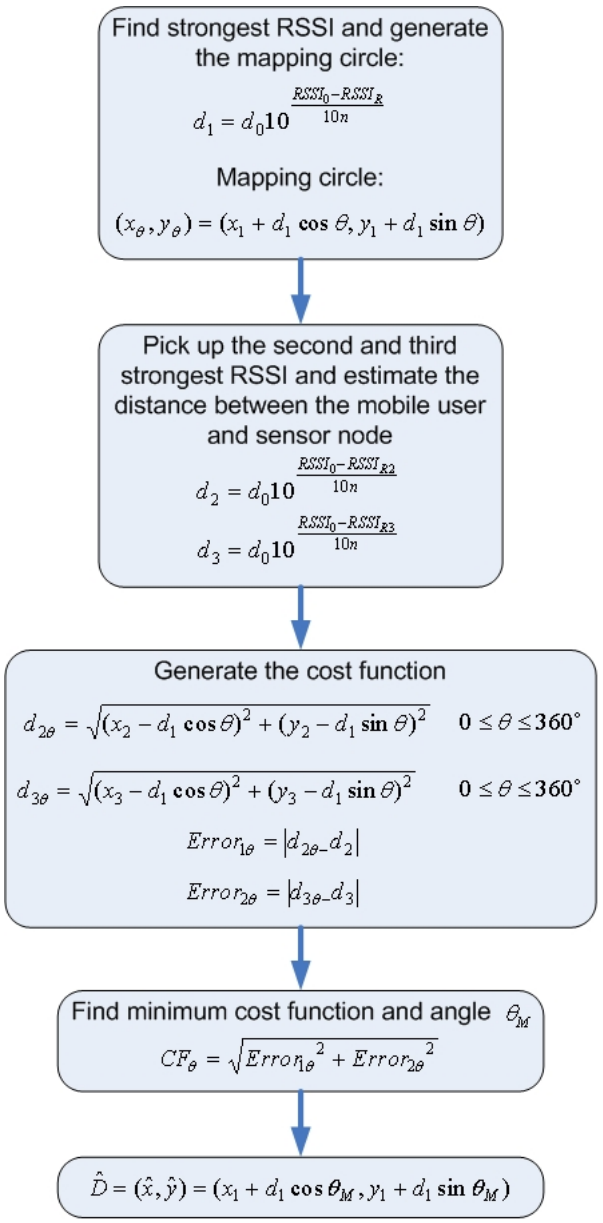


Fig. 5. Dynamic Triangular Algorithm Procedure

E. Prototype Implementation

The final system will include fixed nodes and mobile nodes. Fixed devices will be placed in pre-defined positions. They will be implemented as embedded systems. Mobile nodes can be implemented has embedded systems or objects and persons equipped with ZigBee tags. Mobile devices are the target for location and identification. Nodes will have a unique EPC (Electronic Product Code) identification number [9].

Five ZigBee nodes compose the developed prototype with four fixed nodes and one mobile node. For prototyping purposes, we used a notebook connected to a ZigBee interface through a RS-232 interface.

Fixed nodes and the ZigBee interface of the mobile node are PCB boards supporting commercially available ZigBee

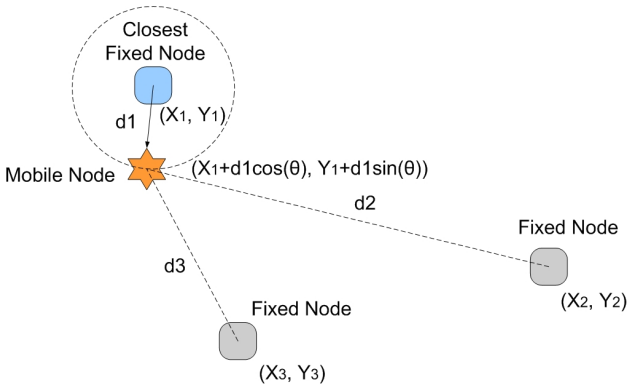


Fig. 6. Dynamic Triangular Algorithm

devices. The developed PCB board also include batteries, a RS232 interface and a USB interface. Programming of ZigBee devices and EPC code assignment is done through RS232. The USB port is used to power the mobile device through a notebook. Fixed nodes are RFD (Reduced Function Device) programmed as ZigBee End Devices. They work independently and are powered by batteries. The mobile device is a FFD (Full Function Device), being programmed as a ZigBee Coordinator. When starting, the mobile device transmits a signal in order to allow the fixed nodes, to detect it. The mobile device is responsible to transmit the received radio signal strength indicator (RSSI) to the software application that executes the location algorithm in order to determine the mobile location.

The software application also initialises the wireless network, establishing the communication between ZigBee devices. Due to hardware limitations we connected the mobile device directly to a notebook, powering the device through the USB port and using the RS-232 port to communicate. A future and more advanced system would be controlled by a server that would execute all the tasks, being the notebook no longer needed.

V. System Characterization

The prototype was installed and tested in a residential environment. The location area is shown on Figures 7 and 8. Table I shows the coordinates for each fixed device. EB1 is located at the origin of the referential.

Fixed Devices	Coordinates (*x*,*y*) in meters
Device 1	(0;0)
Device 2	(4,8;5,8)
Device 3	(0;5)
Device 4	(3,7;0)

TABLE I
Fixed Devices Coordinates

As can be seen on Figure 7, we decided to use a small area. This approach minimizes the location error. In fact, the indoor range of ZigBee devices is limited and when distances increase location errors also increase. Although, it is possible

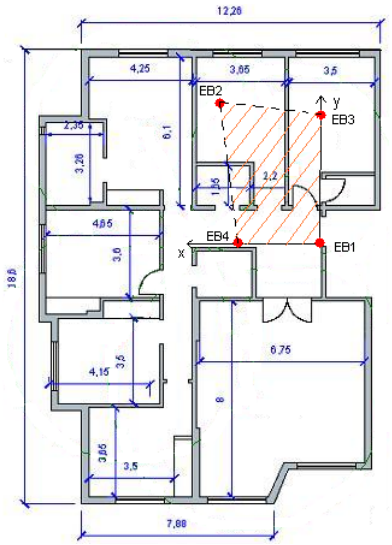


Fig. 7. Location Environment

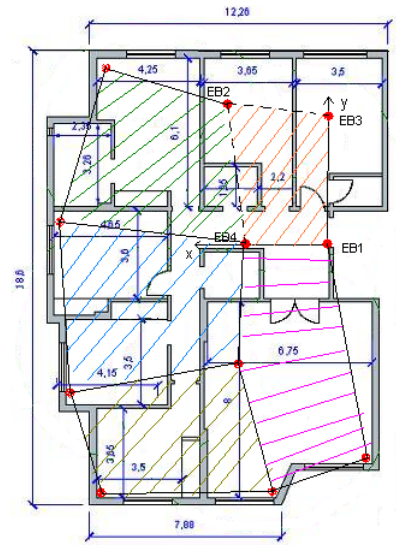


Fig. 8. Set of small location environments

to locate objects in a much larger area, using several small areas, as presented on Figure 8. Using similar small areas with small obstruction also makes easier to characterize the environment. However, the number of fixed devices need to be increased. (The number of devices shown in Figure 8 only has the purpose of illustrating the general idea behind the use of small areas!)

In order to evaluate the capabilities and reliability of the location system we analyse the prototype in different situations: without the WAF factor, introducing the WAF factor and and using an average value for the WAF factor.

The characterization of RF signals is essential to the location system, only doing this it is possible to predict the signal behaviour during propagation and determine the propagation model parameters (n and $P(d_0)$). This characterization is made through the collection of RSS values in a previous

location stage (off-line).

To determine the propagation model parameters, we followed the scenario analysis described in [11]. Measuring the strength of the received signal (RSS) at the fixed devices, transmitted by the mobile device, we repeated this process moving the mobile device across the location area. RSS values were obtained in three fixed devices and, for each one, in three different directions (as presented on Figure 9). The use of three devices instead of all four is enough to determine the propagation model parameters.

Propagation model parameters were calculated for each device individually and for all devices together. Therefore, there is a specific equation for each device and an equation that can be used for all devices.

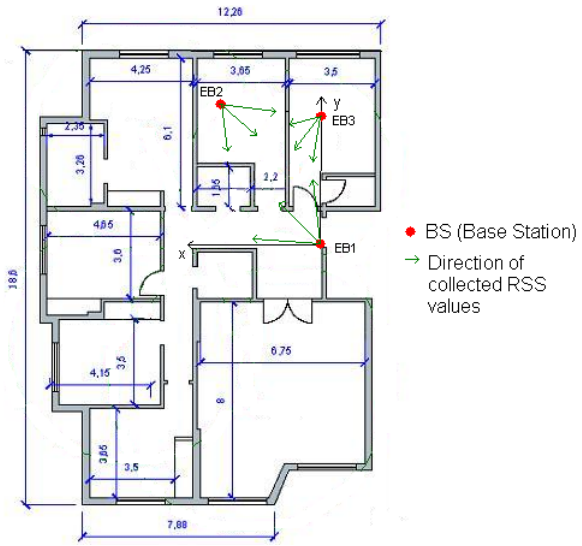


Fig. 9. Direction of collected RSS values

A. Parameter of attenuation WAF

In order to calculate the WAF value, RSS values were measured, for the same distance, with and without a wall between the fixed and mobile nodes. The WAF is the difference between the two RSS values. Table II presents WAF values , for each device and all devices together, obtained for different distances.

Distance (meters)	Device 1	Device 2	Device 3
2,5	-	4,25 dB	-
3	4,42 dB	11,5 dB	13,67 dB
3,5	17,42 dB	-	6,5 dB
4	-	18,08 dB	8,83 dB

TABLE II
WAF PARAMETERS

Table III shows the average value of WAF for each fixed device and for the set of values of all devices together.

B. Characterization of RF signals without WAF

RSS values were collected in each fixed node and in all three different directions. This RSS values permitted, by logarithmic regression, to calculate the attenuation ratio and the received signal strength at the distance of 1 meter. Figure 10 illustrates the use of logarithm regression and table IV presents the values obtained for the fixed devices.

Fixed Devices	Average WAF
Device 1	10,92 dB
Device 2	11,28 dB
Device 3	9,67 dB
All Devices	10,58 dB

TABLE III
AVERAGE WAF PARAMETERS

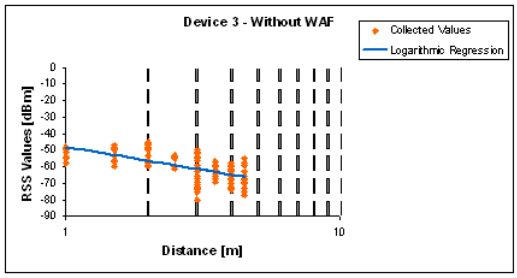


Fig. 10. Collected values for device 3

Without WAF	Attenuation Factor (n)	RSS at 1m ($P(d_0)$)
Device 1	2,382	-45,50 dBm
Device 2	3,637	-40,32 dBm
Device 3	2,806	-48,08 dBm
All Devices	3,027	-44,50 dBm

TABLE IV
CALCULATED PARAMETERS WITHOUT WAF

With the calculated values in the tested situation it was possible to plot the characteristic of the RF signal in terms of the distance using the theoretical model (propagation model) and compare it with the experimental collected RSS values. The following graphics (Figures 11 and 12) compare the theoretical model with the average of the collected RSS values at different distances.

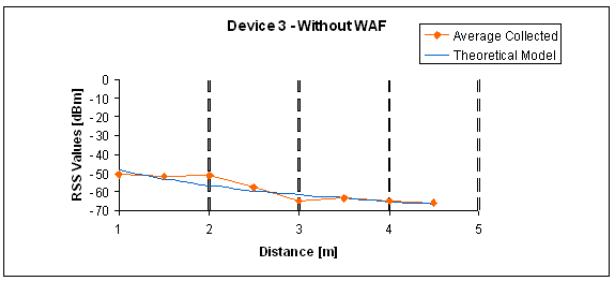


Fig. 11. Average Collected Values VS. Theoretical Model

C. Characterization of RF signals with WAF

This characterization is very similar to the first one and uses the same collected RSS values, but with the difference

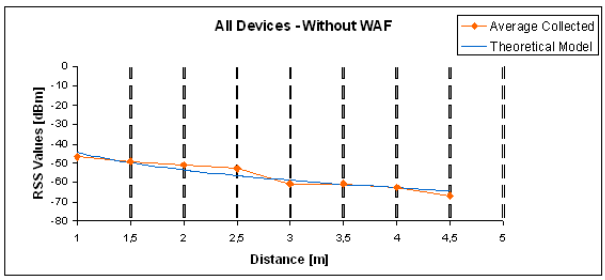


Fig. 12. Average Collected Values VS. Theoretical Model

that WAF values are now taken into account (Table I). Table V shows the calculated values of the propagation model parameters through logarithmic regression and Figures 13 and 14 show the graphics that compare the theoretical model with the average of the collected RSS values at different distances.

With WAF	Attenuation Factor (n)	RSS at 1m ($P(d_0)$)
Device 1	2,012	-45,77 dBm
Device 2	2,803	-41,12 dBm
Device 3	2,391	-48,07 dBm
All Devices	2,255	-45,21 dBm

TABLE V

CALCULATED PARAMETERS WITH WAF

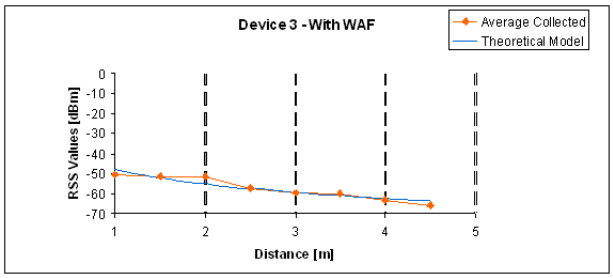


Fig. 13. Average Collected Values VS. Theoretical Model

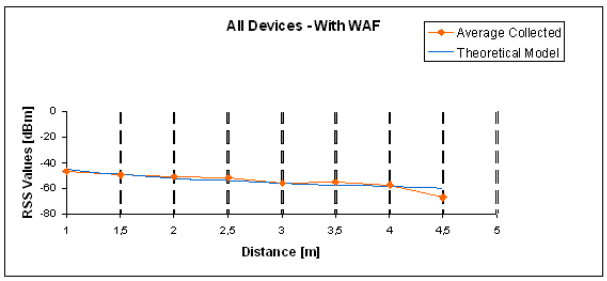


Fig. 14. Average Collected Values VS. Theoretical Model

D. Characterization of RF signals with average WAF factor

This characterization is very similar to the first two and uses the same collected RSS values, but with the difference that the average WAF values are now taken into account (Table II) . The results were not very different then those fo the characterization of RF signals with WAF factor.

VI. RESULTS

After the characterizations the location algorithm was tested. In order to do that the mobile device was placed on two different and random positions. The algorithm was tested without the introduction of WAF, with WAF and with the average value of WAF. The tests were also made for each fixed device separately and for the set of values of all devices together, in this case all fixed devices were characterised by the same propagation model equation. The tests revealed errors from 2,5 to 0,25 meters when using the developed propagation model Tables VI and VII).

Point A (1,8;4) m	Wout/ WAF	W/ WAF	All Wout/ WAF	All W/ WAF
Mob. Node Coord.	(3,14;7,25)	(2,29;6,52)	(6,54;6,50)	(3;-3,63)
Coord. Error	(1,34;3,25)	(0,50;2,52)	(4,74;2,50)	(1,20;-0,37)
Dist. Error	3,51	2,57	5,36	1,26

TABLE VI

LOCATION RESULTS FOR POINT A

Point B (0,7;4) m	Wout/ WAF	W/ Avg. WAF	All Wout/ WAF	All W/ Avg. WAF
Mob. Node Coord.	(0,54;5,65)	(0,27;4,25)	(0,72;5,86)	(0,83;4,23)
Coord. Error	(-0,16;1,65)	(-0,43;0,25)	(0,02;1,86)	(0,13;0,22)
Dist. Error	1,66	0,50	1,86	0,26

TABLE VII

LOCATION RESULTS FOR POINT B

Analysing Figures 11, 13 and 16 we can observe that, for certain locations, the introduction of the WAF parameter increased the error. However, in general, the theoretical model (equation 5) is a good approximation of the collected RSS values. The graphics relative to the deviations of the collected RSS values and theoretical model (Figures 16 and 18) confirm the previous statement. A decrease in RSS values (which translates into a decrease in error), can be observed on Figure 15 when WAF parameter is introduced. This decreased is also noticed when using the average WAF parameter.

Its also verified in the standard deviation of the collected values that at the same distance the fact that the RSS values are not always the same, influences the parameters of the model introducing an additional error. Its also verified that when the set of values of all devices together is used to calculate the model parameters it does not produce very different results of those presented when each fixed device has his own equation.

In general comparing the developed system with other systems, that mainly use location fingerprint techniques, it is

possible to assume that the developed system produces results similar to other systems. Table VIII shows the typical error of other location systems. Although the location environment in which the systems were tested was larger than the one used in this paper, the fact that location fingerprint techniques usually produce more accurate results reinforces the good results achieved by the developed location system.

Location Systems	Algorithm Type	Error
RADAR[2]	Nearest Neighbour	2.13 meters
Youssef[12]	Bayesian	2.13 meters
XIANG[13]	Bayesian, RSS distribution model	1.83 meters

TABLE VIII
COMPARISON WITH LOCATION SYSTEMS

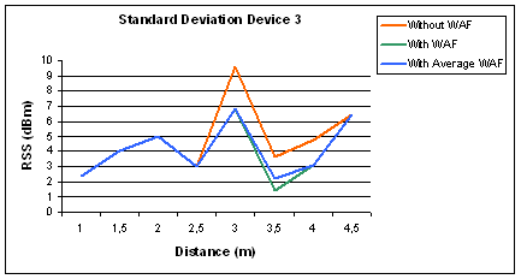


Fig. 15. Standard Deviation of the collected RSS values for device 3

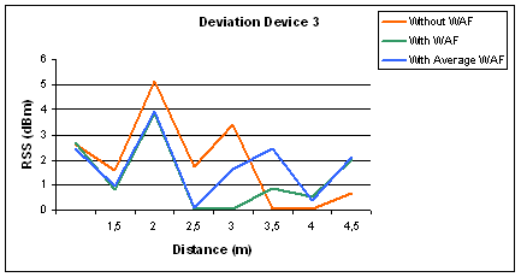


Fig. 16. Deviation between the collected RSS values and theoretical model for device 3

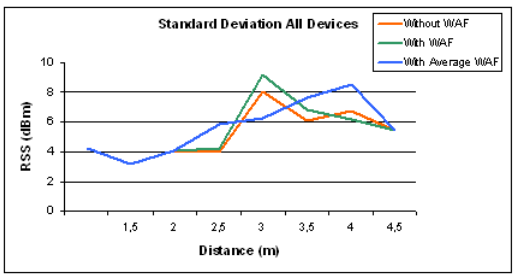


Fig. 17. Standard Deviation of the collected RSS values for all devices

VII. CONCLUSION

This paper describes a location system for persons and objects in an indoor environment, using ZigBee technology.

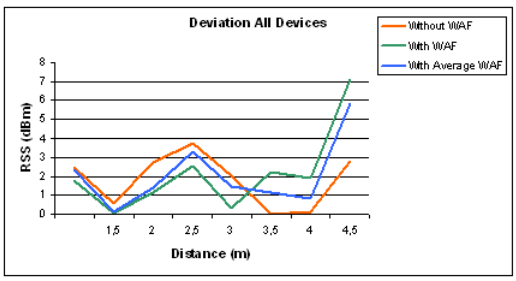


Fig. 18. Deviation between the collected RSS values and theoretical model for all devices

ZigBee nodes can function as RFID tags, having each one a unique EPC identification number. Sensors can be associated with the Zigbee wireless nodes to create applications for home, health and traffic control.

Location is determined, using a propagation model, based on the WAF model, together with a triangulation algorithm. Tests demonstrate an increase in accuracy when the WAF parameter is used in the propagation model. Even the use of an average value for WAF or a set of values proved to be effective. The same equation parameters can be used for each fixed device, avoiding the need to collect RSS values for each fixed device. This becomes very important when a new location system is to be installed in a new environment, where the building architecture is similar.

It is important to remark that signal characterization is crucial to ensure a good performance by the propagation model. The more values of RSS are collected the more accurate the model becomes. On the other hand, it can be demonstrated that with a relative small number of collected values it is possible to develop an effective flexible system, with errors from 2.5 to 0.25 meters when using the developed propagation model.

Results also show that in the moment that the location is initiated, it is important to collect several RSS values in order to obtain an average RSS value in that time interval. The wide range of values that can be collected in the same position can vary enough to generate significant errors. This variation is due to multipath that as the tendency to increase with distance. The closer the devices are of each other the more easily can the propagation model be characterised. This fact proves that the adopted concept of using a small set of location environments that together make a bigger environment, is a valid option. This concept is especially interesting in a ZigBee using system, being that these networks can have thousands of devices associated to them, reducing the distances between the devices, all this at a low cost and low energy consumption.

A more advanced system would be able to track the mobile device instantaneously. The system could be controlled by a central server that managed the ZigBee network and processed all the information needed to the location algorithm. In a future system it would be possible to obtain not only a EPC identification number but a wide range of information regarding the location environment.

REFERENCES

[1] "European Policy Outlook RFID", in Conf. RFID: Towards th Internet of Things, Berlin, June 25-26, 2007.

[2] P.Bahl and V. N. Padmanabhan. "RADAR: An in-building RF-based user location and tracking system. in Proc. IEEE INFOCOM , 2000, pages 775-784.

[3] J. Hightower and G. Borriello, "A Survey and Taxonomy of Location Systems for Obiquitous Computing". Technical Report UW-CSE 01-08-03, University of Washington, Computer Science and Engineering, August 24, 2001.

[4] H. Hashemi, The Indoor Radio Propagation Channel, Proceedings of the IEEE, Vol. 81, No. 7, pages 943-968 July 1993.

[5] S. 0. Rice, Mathematical analysis of Random Noise, Bell Systems Technical Journal, Vol. 23 (1944), Vol. 24 (1945).

[6] Seidel S. Y., Rappaport T. S.,914 MHz path loss prediction models for indoor wireless communications in multifloored buildings. IEEE Transactions on Antennas and Propagation, v. 40, no.2, pages 207-216, February 1992.

[7] A. G. M. Lima and L. F. Menezes. Motley-Keenan Model Adjusted to the Thickness of the wall, pages 180-182.

[8] Mikas F., Zvanovec S., Pechac P. Measurement and prediction of signal propagation for WLAN systems. Czech Technical University in Prague, 2002.

[9] "European Policy Outlook RFID", in Conf. RFID: Towards th Internet of Things, Berlin, June 25-26, 2007.

[10] Ren C. Luo, Ogst Chen and Shi H. Pan. "Mobile User Localization in Wireless Sensor Network Using Grey Prediction Method", Department of Electrical Engineering, Nation Chung Cheng University, Taiwan, 2005.

[11] K. Kaemarungsi and P. Krishnamurthy, Modeling of Indoor Positioning Systems Based on Location Fingerprinting. in Proc. IEEE INFOCOM,2004.

[12] M. A. Youssef, A. Agrawala, and A. U. Shankar, WLAN location determination via clustering and probability distributions", in Proc. IEEE International Conference on Pervasive Computing and Communications (PerCom '03), Dallas, Fort Worth, TX, Mar. 23-26, 2003, pp. 23-26.

[13] Z. Xiang, S. Song, J. Chen, H. Wang, J. Huang, and X. Gao, A wireless LAN-based indoor positioning technology", IBM Journal of Research and Development, vol. 48, no. 5/6, pp. 617-626, Sept./Nov. 2004.

[14] "Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs)", IEEE Standard for Information technology Telecommunications and information exchange between systems, September 8, 2006.

[15] P. Kinney, "ZigBee Technology: Wireless Control that Simply Works", in Conf. Communications Design Conference, October 2, 2003.

[16] L. Yujin, P. Jaesung, "Practical Indoor Positioning System Using Received Signal Strength in IEEE 802.15.4 Networks", 2009.

[17] G. Gomes, H. Sarmento, "Indoor Location System using ZigBee Technology", in Conf. SENSORCOMM 2009, June, 2009.

New Sensing Model for Wireless Sensor Networks

Peter Soreanu, Zeev (Vladimir) Volkovich
Software Engineering Department, ORT Braude College, Israel
speter@braude.ac.il, vlvolkov@braude.ac.il

Abstract -***The paper presents a new sensing model for Wireless Sensor Networks. This model, named Circular Sector Sensing Coverage, was first introduced by the authors at SENSORCOMM 09 conference. It uses circular sectors with variable central angles and radii. The purpose of the model is to minimize the energy consumption of the sensing itself, improving the energy balance of the sensor node (mote). This is especially relevant when the sensing is done remotely, by sending waves to detect intrusion or acquire data. Simulation of this model shows energy savings versus previously published schemes, suggesting the viability and the advantages of this new sensing model. However, no analytical model was yet developed. The present paper expands the scope of the previous one by adding background information about known sensing models, describing in greater details the algorithm used for simulation and by presenting new and significant simulation results. The obtained data confirms the advantages of this new sensing model. The possible implementation of such a WSN may significantly improve the energy-related performance of the WSN, allowing the development of new applications and improving the performance of existing ones.***

Keywords: Wireless Sensor Network; sensing models; sensing coverage; energy efficiency; circular sector

1. Introduction

The paper presents a new model of sensing coverage, supposed to achieve a better efficiency of energy utilization for remote sensing, by using an adaptable radii and angles algorithm. Coverage areas of circular sector shapes, adapted to the residual energy available, are used. The model was first introduced by the authors in [1]. The algorithm is intended to optimize the energy used for sensing, in order to extend the life of the sensor network. It does not assure a complete coverage of the area to be sensed - an NP-hard problem. Although an analytical model was not yet developed, the proposed model is interesting and efficient, and is presented as such to the wireless sensor networks community.

The Introduction section gives a succinct account of WSN-related problems, with emphasize on energy-efficiency related factors: placement and topology, management, coverage, lifetime.

Research in WSN is advancing at a fast pace, as can be seen in recent surveys of this topic - see [2][3]. Simultaneously, more and more actual deployments are implemented. One of the main problems facing WSN applications is the limited life span of the network. A typical node (mote) is powered by a battery, which is generally not field-replaceable. Also, energy harvesting was developed only as a proof-of-concept, and is not yet a viable option for real life implementations. A good survey of energy-related problems may be found in [4].

A typical mote has three energy-hungry subsystems: radio, processing and sensing unit. It is generally agreed that most of the mote's energy is required for communication (ε_c), followed by processing (ε_p) and, finally, sensing (ε_s). The total lifetime of the WSN is a function of their sum Σ, as shown in (1):

$$\Sigma = \varepsilon_c + \varepsilon_p + \varepsilon_s \quad (1)$$

The energy needed for communication ε_c has the same order of magnitude whilst the radio subsystem is transmitting, receiving or in idle state. To reduce the communication costs, two approaches were developed:

- Duty-cycling, i.e., putting the radio in a sleep mode and transmitting only when necessary. This imposes the use of a sleep/wake-up algorithm to coordinate their activities.
- Data-driven approaches, which decrease the amount of sensed data to be transmitted, by optimizing the sampling time, and using data fusion/aggregation/compression schemes.

The power consumption of the processing unit ε_p is generally three-four times of magnitude less than ε_c . Finally, research has shown that the energy consumption of the sensing subsystem ε_s may be significant, even greater than the energy consumption of the radio or processing subsystems [5]. The main factors are power hungry transducers and A/D converters, long acquisition time and the use of active sensors. The class of active sensors contains sensors that use active transducers, as sonar, laser or radar. They cover a dedicated sensing area, by sending out a probing signal. It may acquire data such as remote temperature, information about localization and tracking of moving objects, or simple binary intrusion detection.

Sensing is strongly related not only to energy-efficiency, but also to important problems such as area coverage and connectivity. Both problems may be viewed as a measure of quality of service in WSN. Maximizing coverage and ensuring network connectivity is a difficult task and many solutions were described [6][7]. The coverage of the sensed field is conditioned by an optimum deployment of the motes [8]. Various algorithms and approaches were proposed [9-16]. The lifetime requirements of mote deployment are related to their placement in the site to be surveyed [17]**.** A comprehensive theoretical presentation of the minimum-cost arrangement of the motes in order to achieve the wanted coverage lifetime may be found in [18]. Analytical methods use Linear Programming, Voronoi–diagram based heuristics or Delaunay triangulation in order to achieve an energy-efficient coverage. While the coverage problem is not solved for all cases and methods, good results have been obtained for particular cases. One of the best examples is described in [19] and [20].

The paper is organized as follows. In Section 2 we present various models of sensing area coverage, with emphasize on the disk model and his improvement, the variable radii circular model. The new Circular Sector Sensing Coverage (CSSC) model, which is a further improvement, forms the object of Section 3. The simulation environment for running the algorithm implementing this model is described in Section 4. Section 5 discusses the obtained results, while Section 6 proposes future research paths and concludes the paper.

2. Sensing Area Coverage Models

From a functional point of view, there are two kinds of sensing activities: local sensing and remote (or range) sensing. The first one relates to local measurements, while the late try to detect or measure a change of property of a distant location or range. A typical local measurement may be temperature, humidity, level of radioactivity, etc. in the immediate proximity of the sensor. A typical remote measurement may use ultrasound or a laser ray to detect an intrusion or to track the velocity of a moving object.

Remote measurements make use of physical characteristics of wave propagation. The time-of light method uses pulses of energy transmitted toward the target, and measures the time difference between the transmitted and reflected signals. Another method use the relative phase shift of these two signals. Lasers scanners may send continuous frequency modulated signals, varying linearly with time.

The results of the sensing may be deterministic or probabilistic data. For local sensing, the probabilistic factor is due to the precision range of the sensor. For remote sensing, the probabilistic model takes into consideration the inherent uncertainty of detections or measurements at increasing distance.

2.1 Disk model of sensing

This model defines the sensing area as a circle with radius R_s for a sensor S_i, constant for every *i*. This sensing model is also known as the unit disk model [6][21].

It is generally accepted that the quality of sensing decreases nonlinearly with the increase of the distance, like in (2)

$$S(S_i, P) = \frac{\lambda}{d(S_i, P)^{\alpha}} \qquad (2)$$

where $S(S_i, P)$ is the sensitivity of the sensor s_i at point P, α and λ are parameters, and $d(S_i,P)$ is the distance between the sensor and the point where the measurement is done.

In the deterministic disk sensing model, sensing is done, with a given accuracy, in and only in the area of a disk with radius R_s, around sensor s_i. This is also called a binary sensing model.

In the probabilistic model, the probability of detection (or the accuracy of the measurement) varies as the distance between the sensor s_i, and the point at coordinates (x_i, y_i), $P(x_i, y_i)$ increases. Figure 1 illustrates the different sensing areas, while (3) defines the probability that a node will detect a point P(x_i, y_i). The latest property is called the probabilistic coverage of $P(x_i, y_i)$, denoted $C_{x_i,y_i}(s_i)$.

Note that the quantity R_ε is a measure of uncertainty in detection , $R_\varepsilon < R_s$

$$C_{x_i,y_i}(S_i) = \begin{cases} 1 & if \quad R < R_s - R_\varepsilon \\ p & if \quad R \in [R_s - R_\varepsilon, R_s + R_\varepsilon] \\ 0 & if \quad R > R_s + R_\varepsilon \end{cases} \qquad (3)$$

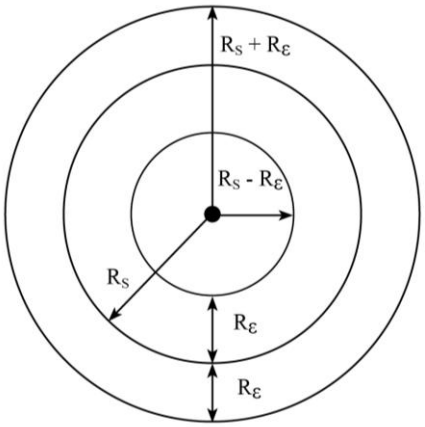


Fig.1 Probabilistic disk sensing model

Generally, p is evaluated as $p = e^{-\omega a^{\beta}}$, where $a = d(S_i, P) - (R_s - R_\varepsilon)$ while ω and β are empirical

parameters that define the detection probability, when an object is situated at a certain distance from the node.

Due to the fact that a point $P(x_i, y_i)$ may lie in the sensing area of more than one node, the total is defined as in (4).

$$C_{x_i,y_i}(\chi) = 1 - \prod_{i=1}^{k} (1 - C_{x_i,y_i}(S_i)) \qquad (4)$$

where χ is the set of nodes whose sensing ranges cover the point $P(x_i, y_i)$, $\chi = \{s_i, i = 1,2,...,k\}$.

This model reflects the sensing behaviour of devices like ultrasound or infrared sensors.

2.2 Variable radii circular model

While the disk model of sensing is simple and the coverage may be relatively easily achieved by using an equal-spaced grid deployment in an obstacle-less environment, this model is not realistic and not energy-efficient. It is not realistic because the variance in sensor calibration and residual battery energy may substantially vary from sensor to sensor during their lifetime. It is not energy-efficient, because the overlapping of sensed areas with more than the necessary sensing radius R_s causes a waste of energy. Another argument is that a large sensing radius will increase the consumption of energy, due to the use of more sophisticated filtering and signal processing methods. The later operation is required to improve the signal-to-noise ratio, in order to achieve an energy-efficient confidence level.

Various models of variable radii circular sensing models were proposed, see [19][22]. They achieve an efficient coverage of the sensed area, while using centralized or local optimization algorithms. The basic idea is represented in Figure 2. It can be seen that by using different radii (R_1, R_2, R_3), the sensed field can be better covered, with less overlapping of sensed areas. Practical implementations/simulations use discrete values for radii.

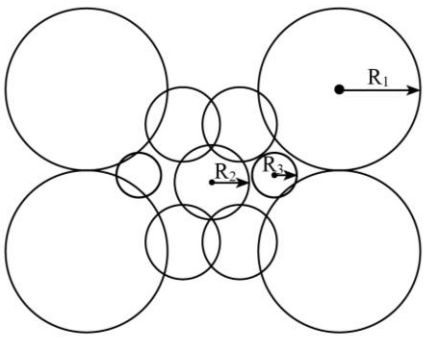


Fig. 2 Different sensing radii

It was shown [6] that this model, when compared to the disk model, presents a more energy-efficient behavior, especially in algorithms which try to solve the coverage problem, i.e., the sensing through all the monitored area. It allows for a more flexible and less redundant deployment policy, while ensuring an optimal approach of solving known coverage problems like

- Minimal exposure path, which is a measure of how well a sensed field is covered for detecting a moving target.
- Maximal breach path, defined as the highest observability path in a sensing field
- Maximal support path, which minimize the distance from any point to the closest sensor.

2.3 Other sensing models

In order to achieve even more energy-efficiency, and to accommodate real-life scenarios, other sensing models were proposed. A brief survey of the most significant ones is given below.

The irregular sensing range model is based on extending the variable radii circular model to a closed polygon. No better energy efficiency is achieved, the sensing range remaining the same. However, the analytical model facilitates a better simulation of the WSN, efficiently identifying fully covered sensors and discovering holes. Centralized or distributed algorithms are used to analyze the coverage problem [23].

Most of the WSN deployments are not truly two-dimensional, but in the majority of cases the height is small enough relatively to the length and width of the network. When simulating the behavior of such a terrestrial network, the third dimension (3D) of the motes may be safely neglected. This is not the case with the underwater, atmospheric or space deployment of WSN. The coverage and connectivity aspects of 3D networks was also researched, and the proposed solutions use generally Voronoi tessellation to partition the space in hexagonal prisms or rhombic dodecahedrons. [24]. These results have energy-related practical consequences. Consider using unmanned aircrafts for airspace surveillance, or underwater unmanned autonomous vehicles for ocean surveillance. Obviously, finding the optimal placement of vehicles, minimizing their number while guaranteeing total coverage of the space, makes a better use of the energy.

A possible approach to mitigate the coverage problem is based on redeploying mobile redundant motes to uncovered areas [25]. Nevertheless, obtaining location information in a GPS-less wireless mobile sensor network demands a lot of energy. This is due to heavy processing and extensive message exchanges. A number of algorithms were devised to alleviate these problems [26].

To be compared, each of the sensing models described in Section 2 would ideally have a complete analytical description. However, the majority of research papers relates to specific aspects, like coverage, connectivity, most exposed path, etc. The energy-effective aspect is generally

taken in consideration mostly by presenting results of running simulations of these algorithms.

3. The Circular Sector Sensing Coverage (CSSC) Model

Our work proposes a new sensing model - the Circular Sector Sensing Coverage (CSSC) - based on circular sectors with variable central angles and radii. The sensing area is the portion of a circle enclosed by two radii and an arc, whose values are set after analyzing the data received from neighbor motes.

In Figure 3 an example of a possible partition of sensing areas between three neighbor sensors (S_j, S_k, S_m) is given. The sensed area between them is dynamically allocated as follows:

- Sensor S_j senses the area covered by the circular sector S_jAB, with radius R_1 and circular angle θ_1 (toward sensor S_k)
- Sensor S_k senses the area covered by the circular sector S_kAB, with radius R_2 and circular angle θ_2 (toward the sensor S_j), and the area covered by the circular sector S_kCD, with radius R_3 and circular angle θ_3 (toward sensor S_m)
- Sensor S_m senses the area covered by the circular sector S_mCD, with radius R4 and circular angle θ_4 (toward sensor S_k).

The same notation S_i is used for the sensor itself and its location. Observe the small overlapping areas between the intersections of circular sectors.

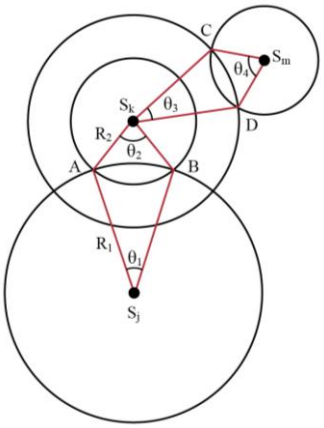


Fig. 3 CSSC model schematics - an example

The CSSC model is based on exchange of residual energy information between neighbor motes. The received data is used to calculate the parameters of the circular sectors covering the sensing areas. The goal is to maximize the coverage, while optimizing the energy consumption. Motes with more residual energy left in their battery may increase the sensing area toward a neighbor with less residual energy. The latest will correspondingly decrease the sensing area in the direction of the neighbor that helps him. The process is iterative, based on negotiations and subject to convergence conditions. It is periodically repeated during the lifetime of the WSN.

The following subsections describe the data exchanged by motes, the implementation of the proposed CSSC algorithm, and the visualization of the whole process.

3.1 Mote communication

In order to establish the sensing areas of each mote, they have to find their neighbors and send relevant information. Finding neighbors is done by broadcasting notification messages and assuming symmetric communication. After that, they send localization data and synchronization control commands. In the next stages, actual energy-related and sensing area information is exchanged. The relevant fields of such a packet contain sender and receiver ID, sender location, operation code, energy level and coverage percentages, maximum and actual coverage distance, supplementary parameters.

The purpose of the communication is to:

- receive enough data to calculate a variable called *StatParam*, used in simulation - See (5).
- notify neighbors about the current sensing area, decided as a function of the value of *StatParam*

$$StatParam = \frac{EnergyLeft}{\frac{CoverageConsum}{TimePeriod} \bullet CrntCoverage + 1} \quad (5)$$

The variables are defined as follows:

- *StatParam* is the relative residual energy of the mote, function of the currently covered sensing area.
- *EnergyLeft* is the absolute value of the residual battery energy.
- *CoverageConsum* is the energy needed to sense the covered circular sector area.
- *TimePeriod* is the time interval between *StatParam* evaluation
- *CrntCoverage* is the area of the circular sensor where this mote does the sensing

If this value is greater (by a predefined threshold) than that of a neighbor node, the sensing area in his direction will be increased.

3.2 Area coverage algorithm

The algorithm used for the simulation of the CSSC model is described, skipping only minor details. The following steps are done for every mote:

Step 1

Broadcast and receive control messages from all sensors. Based on localization information, build a list of neighbors. A neighbor sensor is one which is situated closer than twice the maximum sensing distance (radius). The assumption is that the communication range is greater than

the sensing one, and location information is available and part of the control message. Go to Step 2.

Step 2

Check the number of neighbors in the list. If no neighbors are found, use the *UnitDiskModel* of sensing, i.e., circular sensing with maximum radius, and apply the algorithm to the next mote. If only one neighbor is found, go to Step 3. If more than 1 neighbors are detected, order them according to their angular orientation: sort them beginning with the least angle (from 0), clockwise. Go to Step 3.

Step 3

For every neighbor, check if the maximum coverage distance was received. If not, ignore the neighbor till receiving this parameter. For all other neighbors, calculate the radicals, i.e., the intersection lines of their maximal radius sensing coverage circles. If radicals were found, select and retain only the relevant ones (i.e., those that are not hidden by other neighbors).

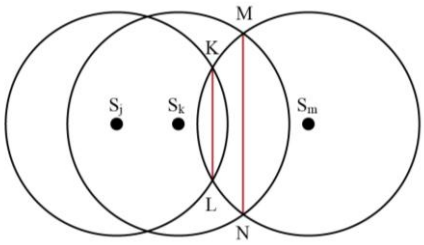


Fig. 4 Discarding radicals example

As can be seen in the example from Figure 4, the radical KL of motes S_j and S_m is situated in the sensing area to be covered by motes S_k and S_m. Consequently, radical KL will be discarded, while radical MN will be retained for further use. Go to Step 4.

Step 4

Calculate the *StatParam* values and compare them. Decide for each neighbor if you can help to preserve its energy reserve by increasing the covered sensing area in his direction. Communicate the changed radical to the neighbor. Do *Step 3* for all neighbors and go to Step 5.

Figure 5 gives an example, showing how two equivalent circular sector sensing areas (S_jPQ and S_kPQ) may be substituted by two asymmetric ones (S_jRS and S_kRS). This allows mote S_k to consume less energy, at the expense of mote S_j.

Step5

Eliminate overlapping of temporary calculated sensing areas for pairs of motes, using the results of the sort done in Step2. Do the sensing according to the newly defined circular sectors.

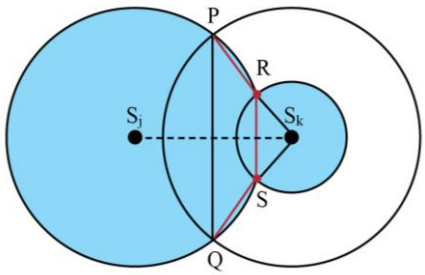


Fig. 5 Moving the radical example

Steps 1 to 5 are repeated periodically (in our simulation, every 5 minutes) for all motes. In order to avoid unnecessary changes, hysteresis was implemented when evaluating the differences of the *StatParam* values.

3.3 Visualization of the process

While running the algorithm described in the previous subsection**,** "housekeeping" calculations are taking place, to find and draw the current circular sectors. The results are used mainly to dynamically draw the sensing areas of the WSN on the screen. Furthermore, the obtained data is also used in energy calculations and as help to check the analytical model in work. These calculations doesn't affect the concept of the CSSC model, being based on specific geometric and programming implementations. The same results may be obtained using different procedures. While running interleaved with the computations needed to implement the CSSC algorithm, these calculations are done mostly during Step 4. Results from [27] were used for clipping arcs.

After eliminating the unneeded radicals, belonging to hidden neighbor motes (Step 3), a tentative temporary new radical, defining the boundary of circular sectors for two neighbor motes, is calculated (Step 4). A collection of intersection segments is saved for each pair of neighbor motes. The mote with the bigger sensing area will save three segments, while the mote with a smaller sensing area will save only one. Referring to Figure 5, mote S_j will save segments PR, RS, and SQ - while mote S_k will save only segment RS.

The next actions are needed to eliminate the ambiguity of areas which may be covered by different pairs of neighbor motes, i.e., they are overlapping. This is done by comparing the relative positions of the saved intersection lines, filtering them and even defining new ones.

Finally, the current circular sectors for the mote, illustrating the received sensing areas, is decided. In the example given in Figure 6, three circular sectors were obtained, with the following parameters:

$\theta_1=2\pi/3$, $_1$
$\theta_2= \pi/4$, r_2
$\theta_3= \pi/4$, r_3, and $r_1> r_2> r_3$

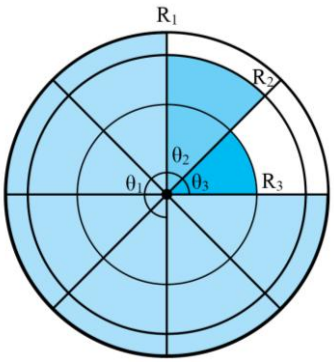


Fig. 6 Coverage example

It has to be noted that no complete coverage may be guaranteed for a given deployment and residual energy of motes. Conditions (6) and (7) hold for every mote:

$$\sum \theta_i \leq 2\pi \quad (6)$$

where θ_i are the circular angles of the sensing circular sectors of a mote.

$$r_i + r_j \leq R_i + R_j \quad (7)$$

where r_i , r_j are the current radii of neighbor motes S_i and S_j, while R_i , R_j are their maximum sensing radii.

The algorithm may be implemented centralized or as a distributed process, at each mote. The energy of processing overhead is obviously smaller than the energy used by emitting light, electromagnetic or sound waves.

4. Simulation

A dedicated simulation program was developed, based on .NET and GDI+ environments. For the simulation data, XML format was used. The simulator allows the programming of the following WSN coverage sensing models: disk unit (i.e., constant radius), circular (i.e., variable radii), and circular sector CSSC (i.e., variable radii and angles). It has the usual features: mote deployment methods, editing mote proprieties, save/import/export scenarios, logs, result analysis, GUI and viewing area, etc. The simulation may be checked in virtual real time, using an adjustable virtual clock, which allows specifying a desired running timeline.

The use of Opnet or ns-2 simulators was also considered. However, when weighting their advantages versus the flexibility of a purpose built dedicated simulator, the last solution was preferred. The main reason was the complexity of the CSSC model visualization.

Special care was given to chose the parameters of the simulation. Data from similar simulations were evaluated. The effects of routing protocols, which were not the object of this research, were minimized (by using single hop connections). A decision to not use, at this stage of the research, heterogeneous motes, was also taken. The planned testbed for WSN, to be built this year, also influenced the choices. During the simulation process, some of the settings and parameters were slightly changed, mainly to achieve better visualization.

The main parameters and data for the simulated network are given in Table 1.

Parameter	Value
Maximum number of motes	100 to 200
Sensed area	60m x 60m
Network type	Homogenous
Mote distribution	Normal/manual
Initial capacity of the battery	1-4Ah
Drain current	40mA
Routing algorithm	Single hop
Communication power/message - Tx	40mW
Communication power/message - Rx	5mW
Packet length (data and management)	1kB
Maximum sensing range	10m
Maximum sensing power (full circle, maximum radius)	20mW
Central angle increments	π/18
Radius increments	0.5m

Table1 Network and mote parameters

Typical timing parameters may be found in Table 2.

Parameter	Value
Location transmission	10s after start
First data transmission	30s after start
Data transmission intervals	10s
Neighbor motes negotiations interval	5s

Table 2 Timing parameters

The main metrics used are enumerated in Table 3.

Metric	Defined as
Data reports	Quantity of data sent by all motes
Redundant data	Measurement already sent by neighbor motes
Application messages	Number of data messages
Management messages	Number of management messages
Number of motes	Current number of active motes

Table 3 WSN simulator metrics

For the purpose of simulation, all messages were defined as having the same number of bits.

The simulator may be relatively easy expanded to process new sensing models, mote parameters, running conditions, measurements, and visualizations. A typical screenshot, representing a partial view of the WSN area may be seen in Figure 7. It illustrates the initial phase of the algorithm. The weight of the red rectangles is proportional to the residual energy of the batteries.

The simulated scenario used a uniform random distribution of motes, while the virtual simulated time was 1000 hours. Three algorithms were implemented: unit disk, circular model, and circular sector (CSSC) model. A variable number of sensors were deployed (between 100 and 200), and performance differences were observed and will be discussed in the next section.

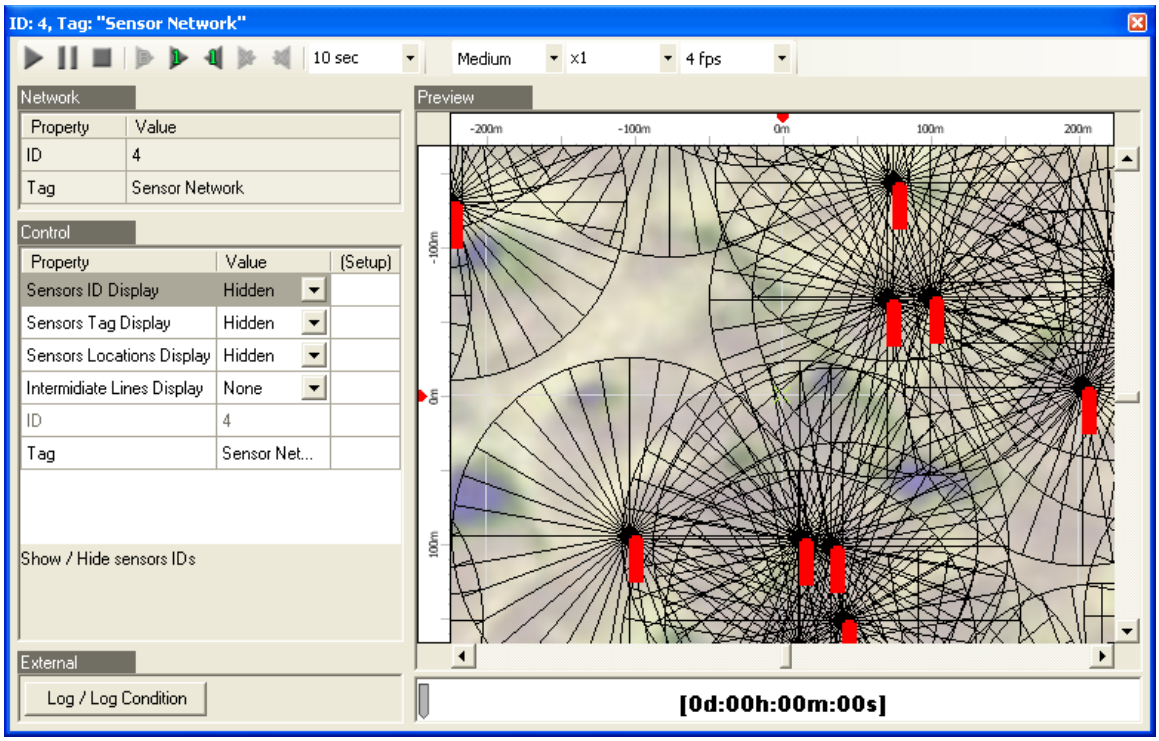


Fig. 7 Network View screen – partial initial field snapshot

5. Results

Extensive running of the CSSC algorithm were done. The results were compared to both the disk unit and the circular model. The life span of sensors, in various deployment situation and energy reserve/consumption assumptions were calculated during the simulations. The CSSC algorithm shows performance improvements of the lifetime of simulated WSNs.

Typical results are presented in Figure 8 and Figure 9. Although different number of motes were used in simulations, the graphs represent the minimum and the maximum number of motes only. Running of algorithms can be seen in two situations: deployment of 100 motes (Fig. 8) and 200 motes

(Fig. 9) in the same field. Both graphs plot the number of active motes, i.e., the motes with still residual battery energy as function of running time. As a consequence of the single-hop communication model adopted, the communication range is supposed to be always operational till the sink, and therefore the possible loss of the transmission connectivity is not an issue. It can be seen that the CSSC model is consistently more energy-efficient.

Interpreting the chart from Figure 9, it can be seen that after 1000 hours (simulated time), the WSN field utilizing the CSSC model had about twice more active nodes left, compared to the circular model. The CSSC model demonstrated, under the same conditions, a 50% increase in the number of active nodes also vs. the circular sensing model.

An interesting performance difference emerged from the simulations obtained by running 100 vs. 200 motes. The chart in Figure 9 shows shorter life-spans of WSN with 200 motes, compared to WSN with 100motes. The advantage of CSSC model, while observed, is less significant. A possible explanation may be the differences in the energy dissipation mechanisms required by the three sensing models. The energy-efficiency of the disk unit sensing model was not influenced by the number of motes, while the circular model was slightly affected and the CSSC model even more.

We have found a strong negative correlation between the density of motes in the WSN field and the energy-efficiency of the CSSC model. This model is less energy efficient for densely populated WSN fields. As can be seen in Fig. 10, a strong positive correlation exists between the motes density and the number of exchanged management messages. The presence of more neighbors in the sensing range of a mote implies an increased number of negotiations to establish the mutual optimal sensing areas. While this density decreases with time, it contributes to the energy consumption balance of the WSN.

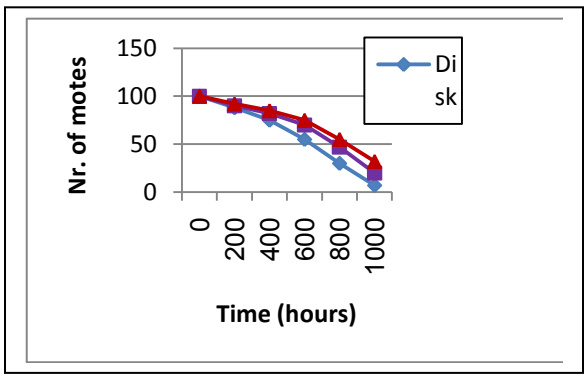


Fig. 8 Sensor field life – 100 motes

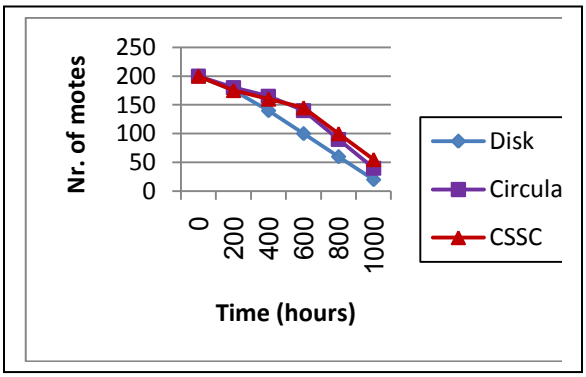


Fig. 9 Sensor field life – 200 motes

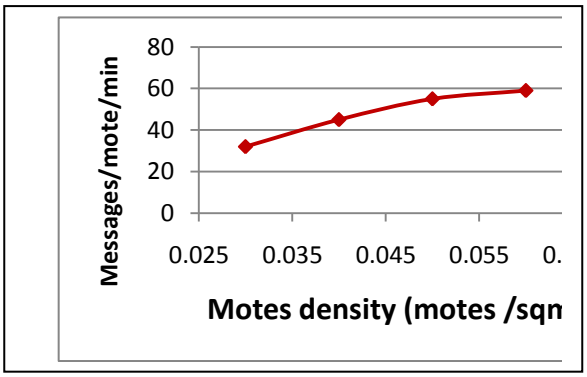


Fig. 10 Number of management messages (CSSC sensing model)

When varying the data transmission intervals, slightly modified simulation results were obtained. As expected, the maximum rate of 5s produced the greater energy consumption, while the minimum simulated rate of 15s presented a much more energy-efficient behavior. The results of these simulations are plotted in Figures 11(a), 11(b), and 11(c). They show the decrease of the number of remaining active motes vs. simulated time, when varying the data transmission intervals, for 100 motes initially in the WSN field. It may be seen that the relative performance of the three sensing models remains invariant, and no significant advantage of CSSC may be reported. At the beginning of the simulation, the increased energy consumption has no influence on motes depletion, due to the still enough existent residual energy.

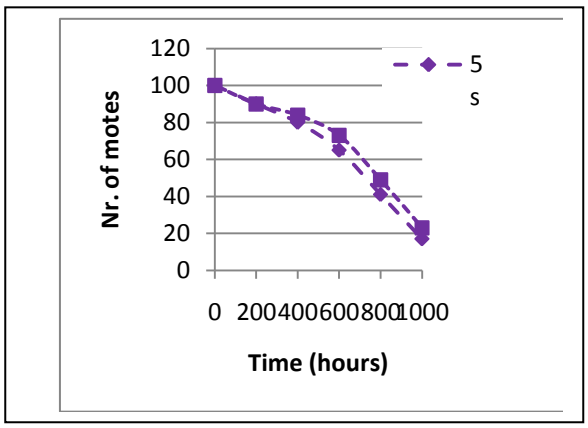


Fig. 11(a) Varying the data transmission intervals (5s to 15s) – Disk model/100 motes

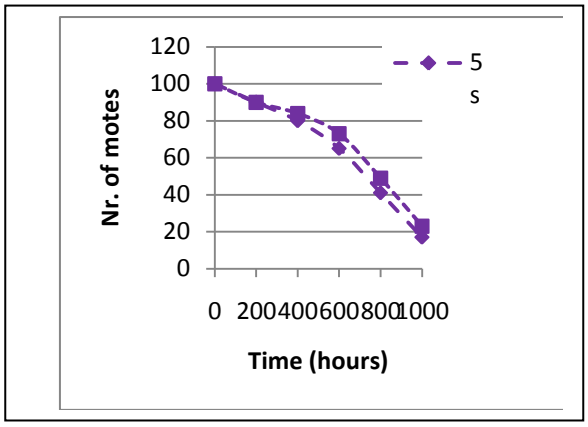


Fig. 11(b) Varying the data transmission intervals (5s to 15s) – Circular model/100 motes

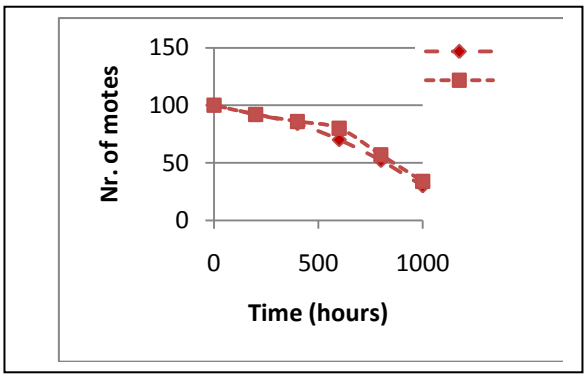


Fig. 11(c) Varying the data transmission intervals (5s to 15s) –CSSC model/100 motes

During the simulation runs, we also changed the rate of negotiation management messages between neighbor motes. For the disk model, where no management messages are used, obviously no performance change occurred. For the circular sensing model, the simulation showed a significant performance worsening when increasing the rate of these messages, as can be seen in Figure 12(a). The increased number of management messages exchanged between neighbor motes, in order to negotiate a possible change of the sensing radii, achieved an increased energy-consumption, resulting in a bigger depletion rate. Simulation showed that the expected improvement of the sensing energy needs was not achieved.

When decreasing the sensed data transmission rate beyond a certain value, a similar increase of the energy consumption was observed. This result is the consequence of using a not actualized and therefore less efficient set of sensing radii. Establishing an optimal rate is obviously needed. It have to be noted that (because of the one-hop communication model implemented) the simulation neglected data fusion scenarios, in which case the negative influence of increased data transmission rates may be alleviated.

A similar behavior was observed for the CSSC model. From Figure 12(b) it can be seen that the influence of varying the rate of the exchanged messages is even greater than for the circular model. The simulation used the same no. of bits for both models. Therefore, this result may be attributed to the following two factors:

- Increased processing time, calculating not only the radius, but also the value of the angle and the position of the arc.
- Less energy-efficient sensing optimization.

While the CSSC model is more energy-efficient than the circular model, it is also more sensible to timing considerations.

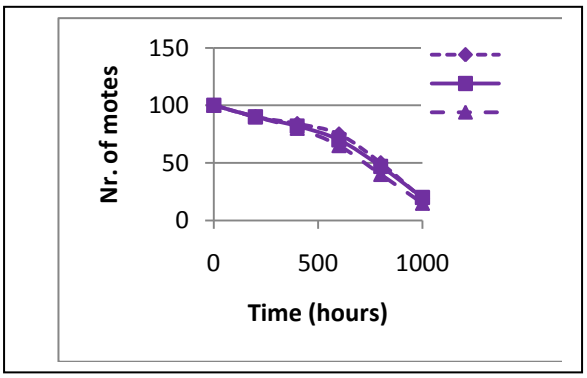


Fig. 12(a) Varying the neighbors' negotiation interval (2s to 12s) – Circular model/100 motes

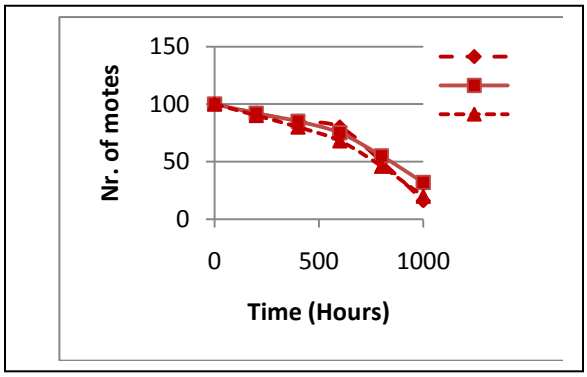


Fig. 12(b) Varying the neighbors' negotiation interval (2s to 12s) - CSSC model/100 motes

From the results of these simulations, it can be seen that the number of active motes remains constantly greater when using CSSC, if the rate of management messages is below a certain level. This level has to be found empirically for every specific WSN using this sensing model.

Comparing graphs for this worst-case scenario, as represented in Figure 13, it can be seen that for improperly chosen parameters, the energy-efficiency of the CSSC model may be even lower than that of the Circular model.

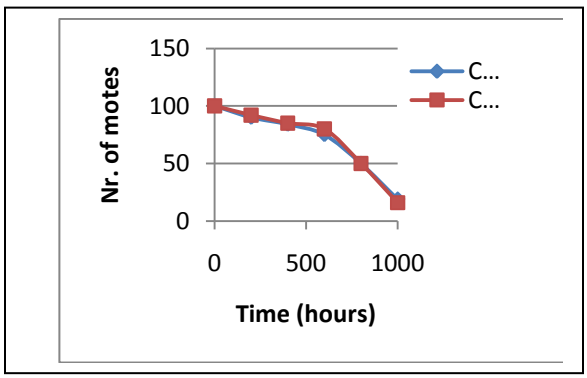


Fig. 13 Circular and CSSC sensing models – worst case scenario

Even if sometimes, at the beginning of the WSN activity, the use of CSSC model seems less energy-efficient than the circular model, toward the end of life-span of the networks it performs better. Significant improvements vs. the circular sensing model were obtained after 1000 simulated hours of sensing/transmitting. The simulation was stopped at this time interval, due to the fact that the remaining number of motes was not enough to effectively cover the sensed field. However, it has to be noted that the simulator has no facilities to check the connectivity of the covered areas. As such, although the lifetime of the WSN was obviously increased, it was not possible to accurately assess coverage data.

6. Conclusion and Future Work

A new model of sensing coverage for active sensors, based on circular sector sensing coverage – CSSC, was developed. The performance of this model was checked vs. previously published schemes, using a dedicated simulation program.. The results of the simulations showed energy savings, suggesting the viability and the advantages of this new sensing model. After 1000 hours of simulated sensing and data reporting, up to a 50% increase in the number of active motes was observed. The best performance, from an energy-efficiency point-of-view, was obtained in a sparsely populated WSN field, using a moderate rate of management messages.

The possible implementation of such a WSN may improve the energy-related performance of the WSN, allowing the development of new applications and improving the performance of existing ones. There are a lot of such applications, civilian or military alike, which implement some remote discovery, localization or tracking activities.

We want to stress again, that no analytical model exists yet for CSSC. However, it is a new sensing model, which as observed during simulations, shows energy-efficiency related performance improvements vs. other known sensing models.

Suggested future possible research topics include:

- Describing analytically the CSSC model.
- Minimizing the required communication overhead, to further increase the energy savings.
- Maximizing the coverage area, while using minimal energy.
- Studying the minimal exposure path, to check its behavior vs. other sensing models.
- Testing the connectivity of sensing areas, in a multi-hop communication environment.
- Researching the behavior of the model in a mobile WSN environment

At this stage, our work concerning the circular sector sensing coverage model CSSC concentrates on:

- Developing an accurate and complete analytical model.
- Defining the optimal rate of the management messages.

- Adding new metrics and functions to the simulator, for a more accurate modeling of sensing models and a deeper understanding of the behavior of the circular sector CSSC model.

.

Acknowledgments

We would like to thank the students who programmed the simulator and implemented the CSSC model, especially Itzik Shoshan and Itay Ronen.

References

[1] P. Soreanu and Z. Volkovich, "Energy-efficient circular sector sensing coverage model for wireless sensor networks", *Proceedings of the 3rd International Conference on Sensor Technologies and Applications*, June 2009, Athens, Greece pp.229-233, doi:10.1109/SENSORCOMM.2009/45

[2] J. Yick, B. Mukherjee and D. Ghosal, "Wireless sensor network survey", *Computer Networks: The International Journal of Computer and Telecommunications*, Elsevier, vol.52(12), August 2008, pp.2292-2330, doi:10.1016/j.comnet.2008.04.002

[3] V. Potdar, A. Sharif and E. Chang, "Wireless sensor networks: a survey", *Proc. 2009 International Conf. on Advanced Information Networking and Appl. Workshops*, May 29-29, Bredford, UK, pp. 636-641, doi:10.1109/WAINA.2009.192

[4] G. Anastasi, M. Conti, M. Di Francesco and A. Passarella," Energy conservation in wireless sensor networks ", *Ad Hoc Networks*, Elsevier, vol. 7(3),May 2009, pp. 537-568, doi:10.1016/j.adhoc.2008.06.003

[5] C. Alippi, G. Anastasi, M. Di Francesco and M. Roveri, "Energy Management in Wireless Sensor Networks with Energy-Hungry Sensors", *IEEE Instrumentation and Measurement Magazine",* IEEE Press, vol.12, April 2009, pp. 16-23, doi:10.1109/MIM.2009.4811133

[6] Ghosh and S. K. Das, "Coverage and connectivity issues in wireless sensor networks ", *Pervasive and Mobile Computing*, Elsevier, vol.4(3), June 2008, pp. 303-334, doi:10.1016/j.pmcj.2008.02.001

[7] D. Zorbas, D. Glynos, P. Kotzanikolaou and C. Douligeris, "Solving coverage problems in wireless sensor networks using cover sets", *Ad Hoc Networks*, Elsevier, vol. 8(4), June 2010, pp.400-415, in press, doi:10.1016/j,adhoc.2009.10.003

[8] M. Younis and K. Akkaya, "Strategies and techniques for node placement in wireless sensor networks", *Ad Hoc Networks*, Elsevier, vol.6(4), June 2008, pp. 621-655, doi:10.1016/j.adhoc.2007.05.003

[9] C.-F. Huang, Y.-C. Tseng and H.-L. Wu, "Distributed protocols for ensuring both coverage and connectivity of a wireless sensor network", *ACM Transactions on Sensor Networks*, vol. 3(1), March 2007, 24 pp , doi:10.1145/1210669.1210674

[10] C.-Y. Chang and H.-R. Chang, "Energy-aware node placement, topology control and MAC scheduling for wireless sensor networks", *Computer Networks: The Int'l Journal of Computer and Telecommunications*, Elsevier, vol.52(11), August 2008, pp. 2189-2204, doi:10.1016/j.comnet.2008.02.028

[11] C.-F. Wang and J.-W. Ding, "The optimum sensor redeployment scheme using the most frangible cluster sets", *Computer Communications*, Butterworth-Heinemann, vol. 31(14), September 2008, pp.3492-3502, doi:10.1016/j.comcom.2008.06.004

[12] B.Wang, C,-F. Hock, B, Lim, "Layered diffusion-based coverage control in wireless sensor networks", *Computer Networks: The International Journal of Computer and Telecommunications Networking*, Elsevier,vol. 53(7), May 2009, pp. 114-1124, doi:10.1016/j.comnet.2008.12.013

[13] C.-F. Huang and Y.-C. Tseng, "The coverage problem in a wireless sensor network", *Mobile Networks and Applications*, Springer, vol. 10(4), August 2005, pp. 519-528, doi:10.1007/s11036-005-1564-y

[14] C.-H. Wu, K.-C. Lee and Y.-C. Chung, "A Delaunay triangulation based method for wireless sensor network deployment", *Computer Communications*, Butterworth-Heinemann, vol. 30(14-15), Oct. 2007, pp.2744-2752, doi:10.1016/j.comcom.2007.05.017

[15] Y.-R. Tsai, "Coverage-preserving routing protocols for randomly distributed wireless sensor networks", *IEEE Transactions on Wireless Communications*, IEEE Press, vol. 6(4), April 2007, pp. 1240-1245, doi:10.1109/TWC.2007.348320

[16] I. G. Siqueira, L. B. Ruiz, A. A. F. Loureiro and J. M. Nogueira, "Coverage area management for wireless sensor networks", *International Journal of Network Management*, John Wiley & Sons, vol.17(1), January 2007, pp. 17-31, doi:10.1002/nem.604

[17] J. Wang and N. Zhong, "Minimum-cost sensor arrangement for achieving wanted coverage lifetime", *International Journal of Sensor Networks*, Inderscience Publishers, Geneva, vol. 3(3), May 2008, pp. 165-174, doi:10.1504/IJSNET.2008.018481

[18] T. H. Lai and S. Kumar, "*Foundations of coverage in wireless sensor networks*", Ohio State University Pub., 2006, ISBN:978-0-542-78170-4

[19] J. Wang and S. Medidi, "Energy efficient coverage with variable sensing radii in wireless sensor networks", *Proc. of the Third IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob 07)*, Oct.2007, White Plains, NY, IEEE Computer Society, vol2, pp. 61-68, doi:10.1109/WIMOB.2007.30

[20] C.T. Vu and Y. Li, "Delaunay-triangulation based complete coverage in wireless sensor networks", *Proc. of the 2009 IEEE International Conference on Pervasive Computing and Communications (PerCom 2009)*,March 2009, IEEE Computer Society, pp.1-5, doi:101109/PERCOM.2009.4912842

[21] S.-C. Lee, S.-B. Shin, H.-S. Hwang and C.-S. Kim, "A Study on the Circular Sensing Model with a Low

Power Profile in Wireless Sensor Networks", *Proc. of 5th ACIS International Conference on Software Engineering Research, management & Applications (SERA 2007)*, 20-22 Aug. 2007, Busan, S. Korea, pp. 616-622, doi:10.1109/SERA.2007.32

[22] M. Cardei, J. Wu and M. Lu, "Improving network lifetime using sensors with adjustable sensing ranges", *International Journal of Sensor Networks*, Inderscience Publishers, Geneva, vol. 3(1/2), January 2006, pp. 41-49, doi:10.1504/IJSNET.2006.010833

[23] A. Boukerche and X. Fei, "A coverage-preserving scheme for wireless sensor network with irregular sensing range", *Ad Hoc Networks*, Elsevier, vol.5(8), pp. 1303-1316, doi:10.1016/adhoc.2007.02.020

[24] S.M.N. Alam and Z.J. Haas, "Coverage and connectivity in three-dimensional underwater sensor networks", *Wireless Communications & Mobile Computing*, John Wiley & Sons, vol8(8), Oct.2008, pp. 995-1009, doi:10.1002/wcm.v8:8

[25] Y.-C. Wang and Y.C. Tseng, "Distributed deployment schemes for mobile wireless sensor networks to ensure multilevel coverage", *IEEE Transactions on Parallel and Distributed Systems*, vol. 19(9), pp.1280-1294, doi:10.1109/TPDS.2007.70808

[26] W.Y. Chang, C.-H. Wang, L. C. Hsu, K. J. Cheng., "Coverage algorithms in GPS-less wireless mobile sensor networks", *Proc. Intl. Conference on Mobile Technology, Applications, and Systems (Mobility '08)*, September 10-12, Yilan, Taiwan, ACM Publisher, doi:10.1145/1506270.1506368

[27] C. J. Van Wyk, "Clipping to the Boundary of a Circular-Arc Polygon", *Computer Vision, Graphics, and Image Processing*, vol. 25(3), March 1984, pp. 383-392

FastM: Design and Evaluation of a Fast Mobility Mechanism for Wireless Mesh Networks

Luís Couto, João Paulo Barraca, Susana Sargento, Rui L. Aguiar
Universidade de Aveiro, Instituto de Telecomunicações, Aveiro, Portugal
{lcouto, jpbarraca} @av.it.pt, {susana, ruilaa} @ua.pt

Abstract

Although there is a large volume of work in the literature in terms of mobility approaches for Wireless Mesh Networks, usually these approaches introduce high latency in the handover process and do not support real-time services and applications. Moreover, mobility is de-coupled from routing, which leads to inefficiency to both mobility and routing approaches with respect to mobility. In this paper we present a new extension to proactive routing protocols using a fast mobility extension, FastM, with the purpose of increasing handover performance in Wireless Mesh Networks. With this new extension, a new concept is created to integrate information between neighbor wireless mesh routers, managing locations of clients associated to wireless mesh routers in a certain neighborhood, and avoiding packet loss during handover. The proposed mobility approach is able to optimize the handover process without imposing any modifications to the current IEE 802.11 MAC protocol and use unmodified clients. Results show the improved efficiency of the proposed scheme: metrics such as disconnection time, throughput, packet loss and control overhead are largely improved when compared to previous approaches. Moreover, these conclusions apply to mobility scenarios, although mobility decreases the performance of the handover approach, as expected.

Keywords: Fast Mobility, mesh networks, MeshDV, neighboring tables, handover signaling.

1. Introduction

Wireless Mesh Networks (WMNs) are dynamically self-organized and self-configured networks, where terminals are connected through routers in a mesh topology. WMNs increase the capabilities of ad-hoc networks, such as robustness, power management, reliable service coverage and optimized node mobility (Figure 1). Coverage increases automatically, allowing a continuous addition of terminals and a self-adapting topology. In the mesh infrastructure there are two types of devices: the Wireless Mesh Routers (WMRs) and the Wireless Mesh Terminals (WMTs). WMRs are devices able to provide multi-hop transport mechanisms enabling communication between the terminals in the same or in different WMNs. A terminal can be any type of device with a wireless interface (typically 802.11a/b/g), whether mobile or stationary. In the particular case of the WIP project [2], where this work was performed, terminals will mostly be comprised of laptops, desktop computers or PDAs, all supporting 802.11a/b/g.

In WMNs, terminal mobility occurs whenever a client associated to an access point (or WMR directly) wants to change its point of attachment. To maintain communication with other terminals, it needs to constantly inform active correspondent nodes about its current location. Any mobility solution designed to these networks must be able to quickly update terminals location information with low overhead yet effectively, creating a reliable, non-interrupted communication between nodes. Cellular technologies, such as the ones used in current GSM and UMTS networks, are able to support seamless connectivity between neighbor points of attachment. WMNs, typically using 802.11, are unable to meet the requirements for voice continuity without further solutions.

In this paper, which is an extended version of the work presented in [1], we propose a new mobility mechanism for WMNs denoted as FastM, Fast Mobility support extension for WMNs, an evolution of MeshDV [3] and Enhanced Mobility Management (EMM) [4], inheriting the basic functional aspects, but using neighboring tables and improved handover signaling to avoid packet loss during the handover process, reduce control multicast packets in the network, save bandwidth and optimize the association and disassociation processes of clients to WMRs. The result is a much optimized and effective solution, able to provide voice continuity over WMNs. The obtained results through simulation show a large increase in performance, for both UDP and TCP communications, in terms of disconnection time, throughput, delays and packet losses, not compromising network overhead and network efficiency.

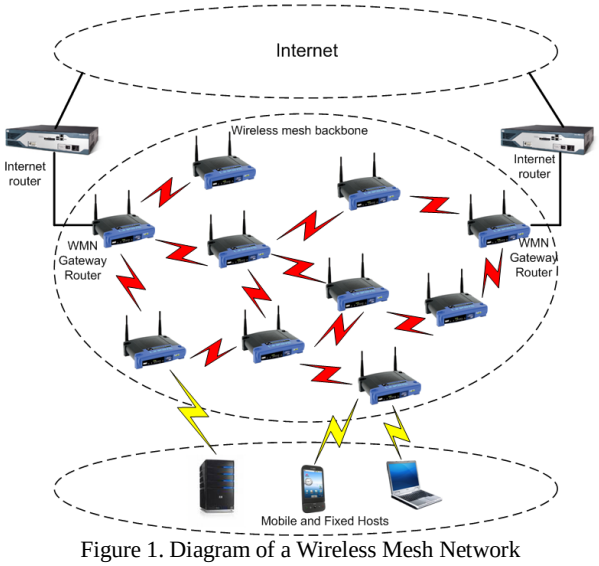


Figure 1. Diagram of a Wireless Mesh Network

The paper is organized as follows. Section 2 presents some of the most relevant mobility mechanisms in WMNs in the literature and their main problems in meeting the requirements of real-time communications. Section 3 introduces the basic routing and mobility mechanism that will be the basis for the protocol enhancements. Section 4 describes the enhanced mobility mechanism, FastM, our proposal for improved mobility in WMNs. Then, Section 5 depicts the simulation scenario and the obtained results, through different scenarios, without and with mobility of nodes. Finally, Section 6 concludes this paper and Section 7 discusses how FastM can be applied to large scale scenarios and describes proposals for future work.

2. Related work

There are already many mobility mechanisms for WMNs in the literature. In this section we describe some of the most relevant mechanisms, stating their benefits and drawbacks.

Ant [5] is a network-based local mobility management scheme for WMNs. Ant introduces some techniques to optimize handovers, such as: a) using the MAC-layer association event as signalling messages, b) maintain IP address of terminals unchanged while moving inside the WMN, and c) pre-establishing tunnels between neighboring WMRs, supporting a list of WMRs neighbors created in each WMR. However, Ant presents some problems: a) the IP address of terminals does not reflect the topology, b) pre-tunnels must be available between every WMR neighbors, which introduces a scaling problem, and c) there is a centralized location server, managing all the location information of the network. In a small scenario with only 4 nodes, handover timing results in the order of 44.5 milliseconds are obtained [5].

MAMP (Mobility-Aware Multi-Path) [6] is a new scheme that uses the interconnection between Serving Access Points (SAPs) and is supported on the existence of a Gateway. It is a multi-path mechanism for packet forwarding, creating a large number of connections between every node, with multiple alternative routes. In this scheme, when a mobile node registers in the network, a message will be forwarded from the correspondent SAP to the gateway, creating routing paths in every SAP that receives the message to the mobile node. Meanwhile, each SAP broadcasts to its neighbors the appearance of a new mobile terminal, and, recursively multi-path routes are created. This solution presents good performance, reducing the handover delay comparing to other techniques, but needs SAPs to have large capacity to deal in a large number of routes. Being this a proactive mobility protocol, it also gives the mobile host the responsibility to trigger the mobility process in the network.

MobiMESH [7] is a WMN mechanism where the network is organized in two sections (backhaul and access), each with a separate IP addressing space. MobiMESH uses a cross-layer mechanism associating MAC and IP layers, making possible to correctly announce associated clients on the backbone routing in a lightweight and fast manner. Results show that, in average, handover using MobiMESH takes 100 milliseconds. However, the association of MAC and IP layers may cause address conflict, and a complete conflict-free strategy may require a central location server or complex interaction between mesh routers.

SMesh [8] uses unmodified WiFi interfaces on terminals. Connectivity and transport is provided by a group of access points, creating the WiFi backbone. Results achieved with SMesh present good performance, with a handover latency time approximately equal to zero (ignoring hardware latency). The main feature contributing to these results in SMesh is the fact that during a handover, traffic to the mobile node is sent by the access points using multicast. However, multicast will consume additional bandwidth. Moreover, in 802.11, multicast data rates are lower than unicast.

Ad-hoc On demand Distance Vector (AODV) Pre-handoff Route Discovery (AODV-PRD) [9] focuses on using concepts of ad-hoc networks, such as the routing protocol (AODV [10]), and optimizes them in a Wireless Mesh Network scenario. The main idea behind AODV-PRD is that a mobile node has always knowledge of the location of an alternative correspondent node, like a backup. The aim is to have a solution with low latency network-layer

handovers and with low overhead, reducing routing discovery process when movement occurs. When a mobile node detects that the SNR value of the current correspondent node falls below a given threshold, it initiates a link-layer scan to detect neighbor wireless mesh routers. From this list, the mobile node selects the wireless mesh router which was detected with the highest SNR value, as its new correspondent node in the handover process. At this time, AODV-PRD is integrated with the signaling scheme of Fast Mobile IPv6 (FMIP) [11] by extending the *Fast Binding Update* and *Handover Initiate* signalling messages with a pre-handoff route discovery request option. This mechanism follows the intention of having mobility mechanism free of changes in the hardware of wireless mesh nodes; however, it also suffers the problem of the adaptation to a wireless mesh network scenario.

QMesh [12] uses a different concept from the one that gave birth to wireless mesh networks. This solution assumes the existence of several gateways, and QMesh uses a common practice to always assign a user to the nearest gateway. When a user moves and associates with a new correspondent node that is closer to a different gateway than its current one, it automatically performs a handover between gateways. This solution is purely location-based, application-transparent, and does not incur a high performance impact, promoting an efficient gateway assignment. One of the main points that QMesh secures is the Quality of Service (QoS) in the mobility management solution. Parameters that incur QoS degradation and additional costs, e.g., network distances and congestion, server (gateway) loads, an optimized gateway assignment algorithm must be taken into account. QMesh has two types of different decisions to manage, one regarding QoS and other regarding mobility management in the wireless mesh network. Mobile nodes can then choose between: migrate between two gateways, and choose a gateway upon a transition. QMesh performs a trade-off between assigning the user to the gateway that provides the best QoS at any given time, and reducing the number of costly gateway handovers. However, it has to be evaluated the costs involved in the monitoring of the QoS parameters, congestion and loads, and if they compensate not to assign a user at its nearest gateway. To manage this solution, some tables need to be created in every router. QMesh maintains two main data structures in each mesh router: a LUC (Local User Cache), which holds the MAC addresses of the mobile nodes whose 802.11 beacons are received by the mesh router, and a GUC (Global User Cache), which holds the mapping of the MAC addresses of the mobile nodes that this node is aware, through the APs that it can be reached at. There is a balanced decision between what is best for the handover process and what is best in terms of QoS.

Geo-mobility [13][14] is an approach that adopts a geographical solution for mobility and location management in spontaneous wireless mesh networks. Like most of the solutions based on locations points, this one needs that nodes know exact geographical positions, by means of GPS or similar, while other nodes can estimate their relative positions. With all this data, it is possible to create a global topologically consistent addressing space. The resulting addressing space is pseudo-geographical, in the sense that the coordinate space is virtual and relative, but anchored in the real world through the exact geographical positions of some routers. Then, an End-Point-Identifier (EID) is used in every mobile node. The EID is a value adopted by the mobile node that remains the same while the mobile node moves around in the wireless mesh network. At a given location, the mobile node uses the address of a nearby router to become reachable from any other location in the WMN. As other solutions, Geo-mobility needs to have some type of location service, where the correspondence of each mobile node between its EID and its current address is made. The location service, being distributed and scalable, is composed by several hash functions giving the robustness needed. Virtual Home Region [15] is a concept adapted to the location service, with the necessary modifications in order to fully adapt to a wireless mesh scenario. The main point that turns this solution adapted to several situations, is the flexibility that is offered based on the movement of a mobile node. If a node moves in short distances, the addresses of nearby routers are topologically close to each other (geographical addressing), and updating addresses can be done in a lazy way (low signaling overhead). The handover performance depends on the update rate of the location service (which intrinsically is involved with the addressing update), and it is assumed that a mobile node moves between closer mesh routers. Re-routing mechanism is performed during the handover through the shortest path. The drawback of this solution is that the non-standard-IP addressing mechanism used can introduce more overhead in the global-state-routing update.

Some other solutions, like Wireless mesh Mobility Management (WMM) [16] try to combine optimization for mobility management and for routing mechanism. Using IEEE 802.11 as the link-layer handover procedures, WMM creates in every wireless mesh router a sort of location service while routing data packets. The location cache brings efficiency to routing packets with mobility of nodes. Every mobile node has a correspondent router; in WMM it is called Serving Mesh Access-Point (SMAP). It is the SMAP that manages all the location information of the mobile nodes assigned and in its radius. When mobility occurs, the SMAP is updated by the location management, combined with a re-forwarding technique of

data packets, using the old and new SMAP in the data flow. Two different cache tables are used to support these procedures, one for routing and other for location. In WMM they are called the routing and proxy table. The first manages the routing paths in the wireless mesh network; the second manages the location information of mobile nodes. Using both tables makes WMM more robust and functional in a dynamically network. Every router has detailed information about the mobile nodes present in the network. One disadvantage of this solution is the overhead and signaling that exists inside the WMN. In large networks, the mesh routes are very solicited, needing to have high performance standards to respond to all solicitations and route correctly in the network.

Finally, although IEEE 802.11r [17] is able to provide fast handover between stations, it is not able to handle the L3 recovery of the network, in terms of routing paths and neighbor information, in order to reduce the latency on the update of the new path in the overall network. Our proposed mechanism will be able to work both with unmodified IEEE 802.11 and IEEE 802.11r, since our approach is performed at the IP layer.

3. MeshDV and EMM

In this section we present two mobility mechanisms in larger detail, MeshDV [3] and Enhanced Mobility Management (EMM) [4], as they will be the basis of our proposal, FastM.

A. MeshDV

MeshDV is a solution proposed for WMNs based on equipments composed by two wireless interfaces, each dedicated to a different sub-network: one offering connectivity to end-user terminals; the other forming a self-organized wireless backbone. The *client* interface is configured as an access point, while the interface used to maintain the wireless backbone, the *mesh* interface, is configured in ad-hoc mode. These two sub-networks will have different routing and addressing mechanisms operating on them. Highly adaptable routing solutions are required in the transport sub-network enabling WMR to route traffic from and to terminals. For this task, it was proposed a routing solution based on the Destination-Sequenced Distance Vector (DSDV) routing protocol [18] running in IPv6. Clients only need to maintain information about their current point of attachment to the network. Traffic is sent towards each correspondent WMR and no modification to the routing protocol is required at the terminals.

Each WMR has a Local and a Foreign Client tables (LC and FC) that keep track of clients present in the network: the LC table contains the list of clients directly assigned to the WMR; the FC table contains the information about clients and their correspondent WMR, which is required in order to allow communication between these nodes and the local ones. MeshDV uses a tunnel-based approach creating a communication channel between end terminals. Terminals only need to know the IP address of the destination client and query the current WMR (using Address Resolution Protocol - ARP - or IPv6 Neighbor Discovery mechanisms). When the client queries the WMR for the location of a given node, the WMR will search its LC table. If the node is not local, it then queries other WMRs in the network and adds this information to the FC table. The client only needs to send packets to its correspondent WMR. The WMR will then create a tunnel for the communication with the correspondent WMR of the destination client.

The module making all this process transparent makes use of the Neighbor Discovery Protocol (NDP) [19], which is ubiquitous in all systems. This way, clients do not need any additional mechanism to communicate, making possible the integration of off-the-shelf equipment without modifications. Traditionally, nodes use NDP to maintain track of the local neighbors and check their local reachability. NDP uses a set of packets and caches to share and maintain information related to nodes in a network. Using MeshDV, the protocol will alter the operation of NDP (at the WMR), allowing impersonation of the remote terminals.

MeshDV introduces several additional messages in order to manage communication, association, and disassociation events. These messages only exist in the backhaul part of the network and are mostly related to the discovery and advertisement of clients:

- MCREQ – Multicast Client REQuest – This type of message is sent by a WMR when the location of a client in the wireless mesh network is unknown.
- UCREQ – Unicast Client REQuest – This is a periodic message that is generated by the WMR to check if the information present in the FC table regarding a particular client is still valid. The purpose is to confirm if reachability still exists.
- CRREP – Client Request REPly – When a WMR receives a MCREQ or a UCREQ and if the client is connected (present and active in the LC table), the WMR answers with this type of message with the requested information.
- CWIT – Client WIThdraw – When a client disassociates from a WMR, this message is sent to all the WMRs that requested information about this node, effectively

notifying them of the withdraw event. Upon receiving this message, a WMR will update his FC table.

- CWREP – Client Withdraw REPly – When a WMR receives a CWIT message and has an entry with the client address in the FC table, it sends a CWREP packet to the WMR that issued the CWIT announcing that the entry was deleted from the table.

In MeshDV, mobility management is based on feedback from the wireless card (MAC layer) and periodic messages (IP Layer). The problem with this approach is that it is affected by the beacon timeout configuration of the wireless driver. When timeouts are considerably long, it is possible that (incorrect) information regarding some node is kept in a WMR for a long time, resulting in connectivity problems. WMRs, as defined by MeshDV, are responsible for all tasks of the handover process, communicating with the other WMRs in order to update caches and maintain information coherent. Standard versions of MeshDV use an approach of self-detection (a predictive approach) where a mobility manager module is responsible for managing the handover process. While being a valid approach, it has poor performance in the real world. This is more noticeable with active communications because, while the association of a client with a new WMR is a fast process, packets will still be delivered to the old location for some time. The result is high packet loss during the handover period until caches expire (a few seconds). This process must be performed in a completely transparent manner to the terminals and consuming the minimum bandwidth. Also, handover must be a fast process with minimal packet loss, giving terminals the possibility of maintaining active communications across different attachment points.

B. EMM

Enhanced Mobility Management (EMM) [4] is a new optimization to MeshDV, designed to improve mobility management, and reduce handover delay. With this solution, a new reactive approach for mobility management is proposed, with the detection of the clients during their movement performed by the new WMR. This solves the refresh delay problem created by the NDP cache [19]. Results show that EMM [4] reduces the handover latency time in MeshDV Network, in some cases from 3 minutes to only a few seconds or less than one second. The main change that EMM adds to MeshDV is a new type of message that is sent when a client changes its WMR association. EMM also proposes modifications to some of the original messages and mechanisms first proposed in MeshDV. Figure 2 depicts de EMM message sequence diagram of a handover process.

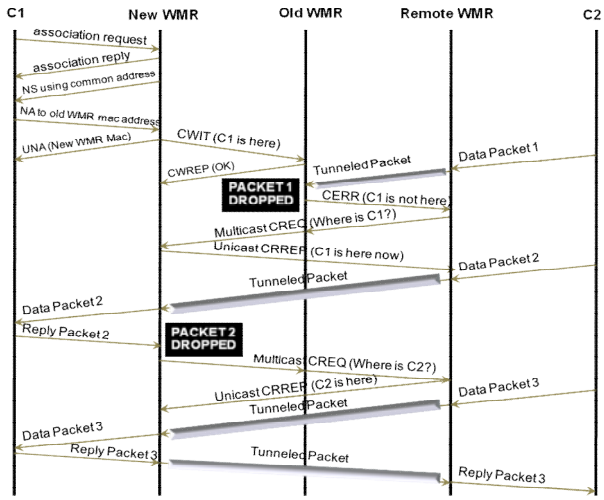


Figure 2. Communication diagram of EMM

One of the messages added is the Unsolicited Neighbor Advertisement (UNA), which is sent by a node to inform its neighbors that its link-layer address has changed, correctly updating the NDP cache of its neighbors with the MAC address of the new WMR. EMM adds an important feature to WMRs: when movement occurs, the old and the new WMRs exchange information related with the old association of the client. This way, the new WMR will receive from the clients the address of the WMR where they were associated (via NDP) and update with its own address (UNA message). Meanwhile, the new WMR informs the old WMR of the association context related to a client. Also, the old WMR will be instantly notified about the new location of its former client and clean local caches.

When a WMR receives traffic from one of its clients with a destination MAC address different than its MAC address, the packet is not discarded. Instead, the packet is tunneled and forwarded to the proper WMR serving the destination address. Then, the WMR sends an UNA message to the client in order to update its NDP cache with the value corresponding to the MAC address of the WMR.

Another message that is added to the MeshDV system is the Client Error message (CERR), which is sent by the destination WMR directly to the source WMR, informing that the client is not associated with it anymore. Receiving a data packet to a node that was removed from the LC table triggers this message.

EMM corrects most of the issues affecting the original MeshDV proposal aiming to be adaptable to wireless mesh networks in general. Results in [4] show reduced disconnection times by a large factor. However, EMM still

achieves results in the order of seconds, thus it is unable to support voice continuity in wireless mesh networks. The problem is that data packets in transit during the mobility process are dropped by the WMR previously used by the client. Another issue is the existence of a large control overhead sent to the network when mobility occurs. The existence of multicast packets during the mobility process for the WMR to be able to locate the sender and receiver client after a handover, creates a large number of unnecessary packets, floods the network and increases the time of the process to locate nodes in the network. Solutions to provide mobility in WMNs must be able to offer better performance with respect to adapt to the actual requirements of nodes, and at the same time, be easy to deploy in real networks. Even being a valid and important improvement, EMM is still not able to support real-time communication during the handover process.

4. FastM: Fast Mobility Support extension

FastM was developed to further optimize mobility in WMNs. It is an evolution of EMM and MeshDV, inheriting the basic functional aspects, but modifying key functionalities, and adding a significant set of effective features. As MeshDV, FastM uses both local and foreign tables, and proposes a new table named Neighbor Client table (NC) used to keep information about neighbor nodes. With the addition of this table, which effectively exploits locality, a new set of control messages is used in order to maintain the coherence of the mesh network. These changes aim to avoid packet loss for the duration of the handover process, reduce control multicast packets in the network, save bandwidth and optimize the association and disassociation processes of clients to WMRs. The result is a much optimized and effective solution providing services like voice continuity over WMNs.

With the neighbor table, all updates made to the local and foreign tables in any WMR are broadcasted to all its neighbors (TTL=1). From real experiments, we notice that handovers are typically performed to neighbor WMRs. In this case, when a data packet reaches an old WMR, the address will be found in the neighbor table and the WMR automatically re-tunnels the packet towards the new location of the client, avoiding packet loss. This produces extremely fewer Client Request packets and speeds the handover process. Our solution does not try to maintain tables consistent, and we assume some incoherence may occur. Nevertheless, if a node has inconsistent information, the algorithm will resort to standard node location mechanisms, guaranteeing proper operation.

A. Signalling Approach

FastM uses a new set of messages to improve the mobility process:

- CLIENT_IS_THERE – When a WMR receives data packets to a client that is in its neighbor table, the WMR checks its foreign table, and a message is sent to the WMRs that had communication with that client. This message contains the new location of the client, enabling FastM to also support mobility that involves other WMRs besides the direct neighbors.
- CLIENT_IS_THERE_CONF – By receiving a CLIENT_IS_THERE message, the WMR is notified that a node moved. Then, this WMR sends to the WMR at the new location, a confirmation stating that a client attached to the first one is communicating with a client attached to the second one. This message acts as a confirmation to the handover process.
- TABLE_UPDATE – This message is broadcasted to neighbors (TTL=1) when the Client table changes. Updates like associations, disassociations, and handovers are then known by every WMR in the neighborhood.
- TABLE_UPDATE_HELLO – In order to periodically refresh node status, and every 30 seconds, all WMRs send a TABLE_UPDATE_HELLO announcing that they are still alive, and that no changes had occurred to the Client table.

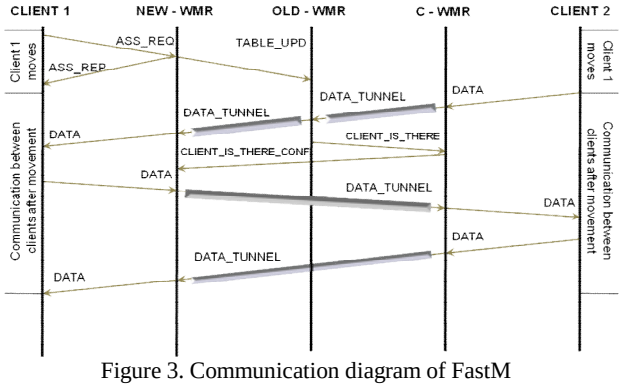


Figure 3. Communication diagram of FastM

Figure 3 depicts the communication diagram of a handover. The changes that characterize this procedure are immediately noticed: there is a tunnel between old and new WMRs, which enables packets to reach their destination until the protocol finally converges; also, no multicast packets are sent to locate clients in the network, in the case of location information is present in the neighbor table. The process occurs as follows:

1) Client 1 issues an ASSOCIATION_REQUEST message to a new WMR. This is a standard 802.11 message. Scanning delays with the new association in

IEEE 802.11 are out of scope of our work. The problem of loosing performance with scanning delays can be resolved using solutions like the one described in [20].

2) The new WMR accepts the association and sends an ASSOCIATION_REPLY message (also standard 802.11) to Client 1. Because the Client table is updated, it broadcasts a TABLE_UPDATE message to all neighbors WMR notifying others about the topology change.

3) When the correspondent WMR forwards packets from Client 2 to Client1, their destination is the old WMR because the correspondent WMR is not informed of the handover process. Data packets arriving to the old WMR are re-tunneled to the new WMR, thus no loss occurs only delay is added. The old WMR will then send a CLIENT_IS_THERE message to the correspondent WMR notifying it that Client 1 has left to the new WMR.

4) Correspondent WMR answers to the new WMR with a CLIENT_IS_THERE_CONF message. This confirms the new location and informs the new WMR about the location of clients that were communicating with the Client 1.

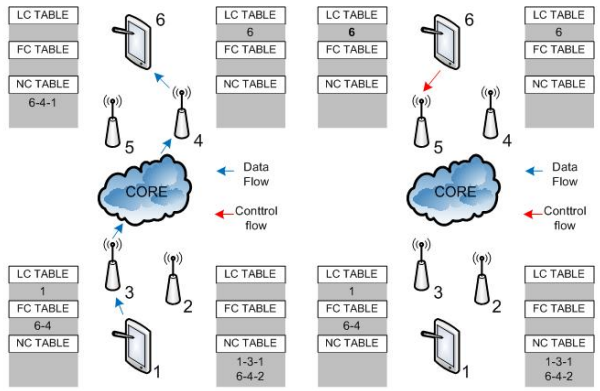
Basically, these are the steps that FastM makes in order to complete a handover process. We can see that in this process the signaling required to perform the handover is mainly between neighbor nodes, which decreases the handover latency and control overhead. There are also changes in the Client Table of each WMR in order to store location information thus predicting future handovers and facilitating them.

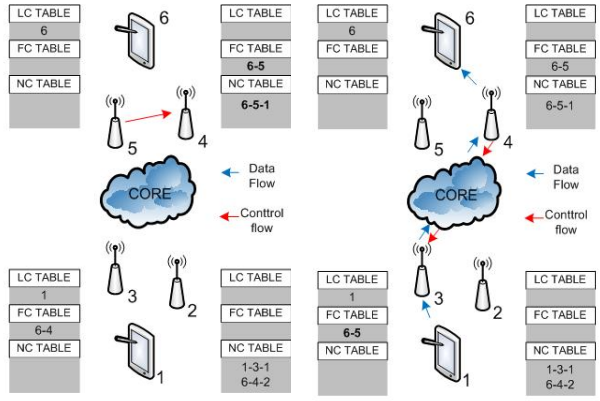
B. Example of FastM Operation

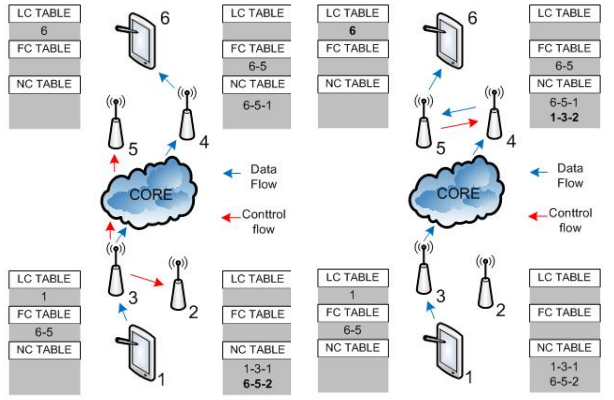
Figure 4 uses an example to show the several steps to reach a successful handover with the FastM mechanism, and to better introduce the changes to the several tables in the handover process. These tables are used to manage neighbor handovers with low latency and without errors.

In Figure 4, in step 1, it is shown the communication between node 1 and 6, and the 3 client tables, local, foreign and neighbor, of the mesh nodes 2 to 5. For example, in step 1, where client 1 is transferring UDP traffic with client 6, node 3 contains: node 1 in its local client table, which is the node directly connected to it; node 6 in its foreign client table (6→4, means that node 6 is connected to node 4), which is the node in the mesh network but not connected to it; and no information in its neighbor client table, since it has no knowledge about neighbors with communication. Node 2 contains no information in local and foreign tables, since this node does not have active communications. However, node 2 has information in its neighbor table of node 1, which is connected to node 3. Since the information in 3 comes from the local table, the third number is 1 (1→3→1). Notice that this third number does not address a specific node, but informs if the information came from a local (1) or a foreign (2) table. Node 2 also contains information in the neighbor table about node 6 that is connected to 4 and whose information came from the foreign table of node 3, and therefore, the third number is 2 (6→4→2). Node 4 is the same case as node 3 in terms of the neighbor table, and contains node 6 in its local table (the node it is communicating with). Finally, node 5 has no information on local and foreign tables, similarly to node 2, and contains node 6 in its neighbor table, which is connected to node 4 and whose information came from the local table of node 4 (and therefore the third number is 1: 6→4→1).

When node 6 moves, it sends an association message to node 5, which will contain now node 6 in its local table (step 2). Then, in step 3, node 5 announces node 4 (neighbor node) that node 6 is performing handover; now node 6 is in the foreign table of node 4 (obtained through node 5), and the neighbor table contains node 6, connected to node 5 and whose information came from the local table of node 4. Then, in step 4, the node 4 forwards the data packets in transit to the new location; it also announces to node 3 the location of node 6. Node 3 updates its foreign table with node 6 connected to node 5. Node 3 then sends information of the new location of the node to the old path of the communication flow, which triggers an update of the neighbor table of node 2, which now contains information from node 5 (step 5). At this stage, all nodes in both new and old communication paths have information on the handover of node 6, and then node 5 confirms to node 4, the previous node associated to node 6, that the handover process is terminated. Node 4 updates its neighbor table with information of node 1 connected to 3, and it will finish the process of forwarding data packets (step 6). Finally, the handover process is finished and the packets are forwarded through the new path (step 7).







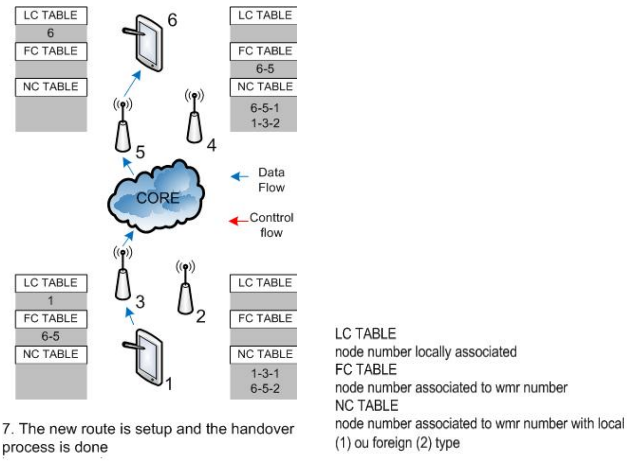


Figure 4. Steps in the handover mechanism of FastM

5. Simulation results

The solution described above was implemented and tested using the ns-2 simulator, version 2.31 [20] on an Ubuntu Linux environment.

Two different scenarios were evaluated: one to compare FastM enhanced MeshDV (MeshDV FMS) with standard MeshDV, and one to provide a more detailed study of the MeshDV with FastM solution in mobile environments with increased mobility. Evaluation of FastM MeshDV in comparison to EMM MeshDV will be addressed in future work.

In order to compare the FastM solution with standard MeshDV, a cloud-type scenario has been chosen with 37 nodes, in a square flat area of 6400 m^2 (800m x 800m). The number of nodes and the topology was chosen to enable the change of several paths during the handover. The MAC layer protocol is the IEEE 802.11 with 11 Mbps of maximum bandwidth, and 150 m of transmission range for every node. This scenario is depicted in Figure 5.

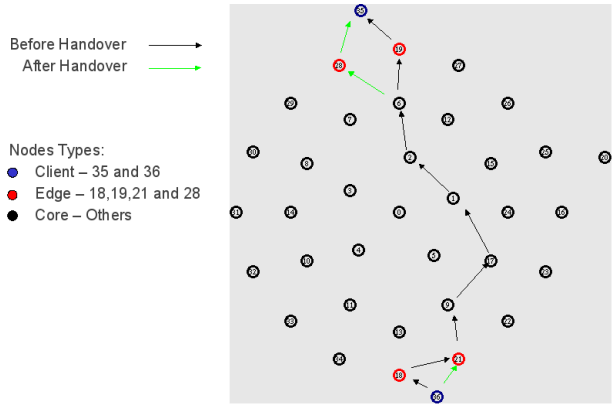


Figure 5. Simulations scenario: comparison between MeshDV and FastM

UDP and TCP flows are generated between two clients nodes as shown in Table 1. In all simulations the receiver node performs handover at t=270 seconds, while the sender node switches attachment point at t=370 seconds. Total simulation time is 450 sec.

Table1. Characteristics of the scenarios

	Configuration 1	**Configuration 2**
Traffic type	UDP CBR	TCP
Packets size	84 bytes	1060 bytes
Sending rate	100 pkt/sec	N/A
Number of flows	1	1

In order to better mimic the real world, artificial delays have been introduced in the WMN. These delays are used to emulate the delay required for the network interface to change channel and the network stack to configure a new address (50 milliseconds). The value is derived from previous work performed [22]. Other relevant aspect is the artificial control delays implemented in ns-2 to approximate even more the simulations to real situations, in what refers to the implementation of MeshDV, both in EMM and FastM. All other values are set to their defaults. Control packets are retransmitted if an answer (or action) is expected and was not detected, with a backoff starting at 1s.

Using this simulation environment, we evaluate MeshDV without and with FastM, according to the following metrics: throughput, packet loss and control overhead.

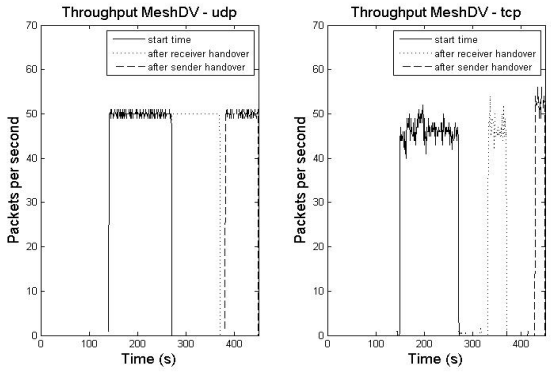


Figure 6. Throughput of MeshDV in the static case and in case of handover (receiver at t=270s, sender at t=370s).

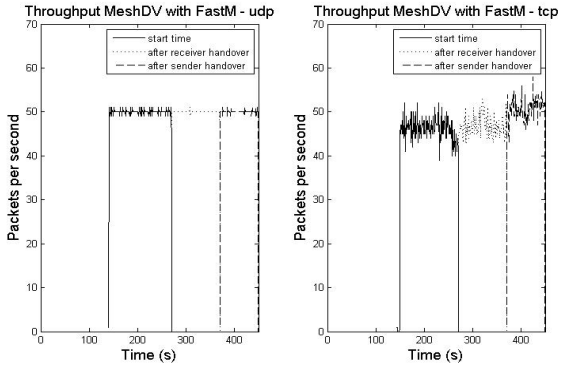


Figure 7. Throughtput of MeshDV with FastM in the static case and in case of handover (receiver at t=270s, sender at t=370s).

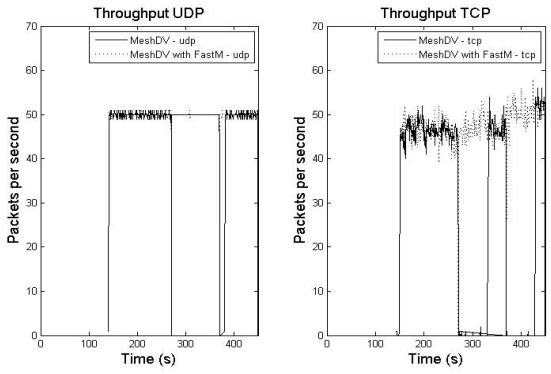


Figure 8. Throughput comparison of MeshDV with and without FastM. The later shows little interruption.

A. Throughput

As depicted in Figures 6, 7 and 8, the results obtained during the handover process, without and with FastM extension, for both configurations, are quite different.

The first curves describe throughput when no handover has occurred. In this situation, and for all cases, throughput is stable. When the receiver changes its point of attachment (second curves), throughput may take some time to reach its previous values if MeshDV is used.

FastM shows a rapid recovery. If the sender moves (last curve), disruption occurs again and can last for several tens of seconds (using MeshDV). Table 2 compares these disconnection delays with physical testbed results, obtained with MeshDV and EMM and previously presented by its authors in [4].

The testbed experimental results of both MeshDV and EMM (Table 2) show handover times in the orders of seconds; even in EMM (which is the best case) the handovers when using UDP sessions require 2 to 4 sec, while the ones of TCP sessions require 35 to 40 sec. In simulations, the values are considerably lower. However, TCP sessions in MeshDV induce disconnection delays in the order of 60s, which is unacceptable for normal communications (even if not real-time). This happens due to inappropriate scanning process, sub-optimal NDP cache handling, and 802.11 scanning. TCP performs worst than UDP due to the rate adaptation algorithm, which takes a long time to recover when packets are being lost.

Table 2. Disconnection time comparison

	Receiver Handover (s)		**Sender Handover (s)**	
	UDP	**TCP**	**UDP**	**TCP**
MeshDV[1)]	15	240	190	185
EMM[1)]	2	40	4	35
ns-2 MeshDV[2)]	0.279	61.287	0.340	59.537
ns-2 MeshDV[2)3)4)]	0.279+ [0...180]	61.287+ [0...180]+ [0...30]	0.340+ [0...180] +[0...30]	59.537+ [0...180]+ [0...30]
ns-2 FastM[2)4)]	0.118	0.344	0.089	0.141

1) Results obtained on a physical testbed (see [4])
2) Results obtained through simulation in ns-2
3) The wireless driver used by the AP's in the physical testbed, has a scanning delay, which varies between 0 and 180 seconds, and increases handover time by ~90s.
4) Due to the use of NDP in clients, there's a cache update delay, which ranges between 0 and 30 seconds. This will increase handover time by ~15s.
(Inclusion of notes 3) and 4) aims at providing results closer to the ones expected in real world scenarios)

Comparing the disconnection times in Table 2 between MeshDV and FastM (through simulation), the values obtained in FastM are significantly lower. With UDP, even when both sender and receiver clients move, traffic values are reduced and the timeout imposed by the wireless driver and NDP are suppressed; in this case, the handover time is reduced from 300 milliseconds to 100 milliseconds, a 3 times improvement. Using TCP traffic, the differences are even more evident: the techniques implemented in FastM (which minimize packet loss) are able to lower the disconnection time to milliseconds (between 100 and 300 milliseconds) compared to the 60 seconds of MeshDV (an improvement of more than 200 times). This large disconnection delay happens due to TCP congestion avoidance mechanism. When disconnection occurs, and both delay and loss figures increase, TCP will reduce the packet rate in order to minimize loss. Because disconnection time spans for several seconds, the exponential backoff will increase to high values, further increasing disconnection time for TCP applications. Ultimately, sessions may be terminated and then restarted. When using FastM, because disconnection time takes only a few hundreds of milliseconds, backoff never reaches high values, and TCP recovers more rapidly. Moreover, TCP creates two flows requiring routing, which greatly increases disconnection time.

B. Dropped Packets

Figure 9 shows the amount of dropped data packets in every scenario. The results using FastM in MeshDV show a large decrease of dropped packets during handovers, both in UDP and TCP traffic.

Using UDP traffic this difference is more noticeable due to a 10 seconds gap in which the communication between the clients is non existing due to NDP session timeout (as the handover occurs at t=370s, it only needs 10 seconds for the NDP timeout, since it is issued every 30 s). In this period, there is a large number of dropped packets, as the session in the client is not updated to the address of the new WMR. When the receiver handover takes place (t=270s), there is a small number of dropped packets, during the period the receiver changes attachment point, and the new WMR searches the location of the correspondent client (sender node).

With respect to FastM, there are no dropped packets when the receiver changes attachment to a new WMR. This is due to the re-tunnel of data packets in transit. When the server handovers to a new WMR, some packets are lost during the time it takes to disassociate and associate to a new WMR. An improvement of 97.3% (from 546 to 15 packets) is obtained with the use of FastM in MeshDV. Using TCP, FastM also shows great improvements, in this case of 35.1% (from 57 to 31 packets). As it can be shown in Figure 10, there are fewer packets generated when FastM is not used in MeshDV, which will also result in less packets being dropped. This happens because the location of clients is unknown after handovers. With FastM, packet generation is constant and some losses exist due to the TCP characteristics, such as drop links and full queues, during the simulation period. During handover, only 7 packets are dropped (4 in the first, 3 in the second handover) between the disassociation and association times of clients to a new WMR.

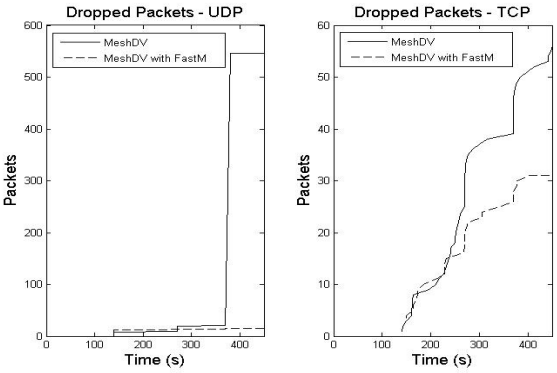


Figure 9. Sum of dropped packets after stabilization

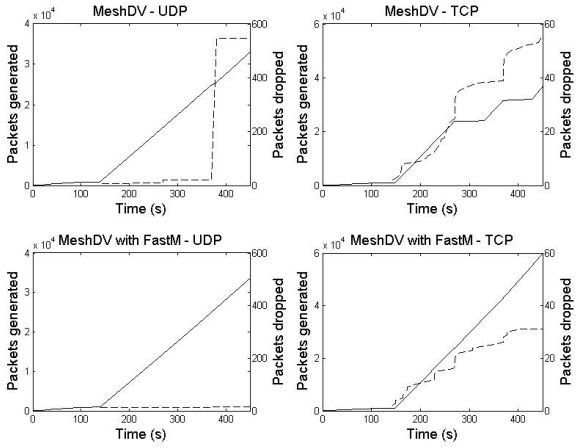


Figure 10. Comparison between generated and drops packets

C. Control Messages

In what refers to control messages (Figure 11), there are significant changes using MeshDV with and without FastM. While analyzing the performance of MeshDV without FastM, we see a typical and coherent ladder shaped process during the simulation. This is present either using UDP or TCP traffic. The initial control packets (t=10s) are due to the initial associations of the clients to the WMRs. At t=140s, when data transfer starts, some control packets are generated in order to locate the correspondent clients in the WMN. With TCP traffic, there are two client location processes (the second takes place at approximately t=143s), due to the packet flows of TCP (data and ack). Then, in each handover, control packets are generated in order to locate the clients. In TCP, due to the loss of links, some exchanges of packets are performed after the handover takes place, causing a disconnection time during this period.

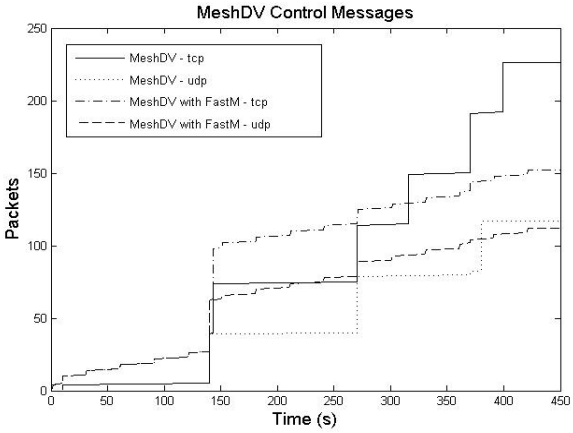


Figure 11. Sum of control packets sent to the network.

Using FastM extension in MeshDV, when mobility takes place, the number of control packets is reduced. Due to the existence of the neighbor client table, there are TABLE_UPDATE_HELLO messages every 30 seconds. During the handover, FastM reduces the control packets from 117 packets to 112 packets in UDP traffic, and from 226 packets to 152 packets in TCP traffic. With this traffic, an improvement of 32.7% in control packets is obtained. This will be even more significant in a scenario with larger number of mobile nodes and more frequent handovers. Please consider that the improvement we achieve is expressed in terms of number of packets sent to the network, which is more appropriate considering that the medium is shared. A single message sent by a node will lead to the generation of multiple packets (one for each forwarding node, plus one sent by the origin). By better exploring locality, from one side communications are reduced, and from the other they involve nodes in closer proximity, thus reducing the impact of the routing protocol.

D. Mobility Speed

In this section we analyse the performance of the FastM solution with respect to the speed of the mobile client. The scenario is similar to the previous one, but nodes are placed on a square grid instead of a circle, as shown in Figure 12. This square scenario was used in order to simplify the mobility pattern and the estimation of the position (and point of attachment) of the mobile node. All WMR nodes are fixed, while one mobile node moves around the others at speeds ranging from 4m/s to 16m/s. This node (node 26), starts its movement at the lower left corner, and moves first to the right, then up, left and down, stopping near to node 2. In this scenario, the mobile node is communicating with one of the fixed nodes either using TCP or UDP. When using TCP, it sends packets with 1000 bytes of data (1040 bytes including TCP and IP headers); bandwidth is limited only by TCP internal

contention mechanisms. When using UDP, the node sends a CBR flow with packets of 1000 bytes and a constant periodicity of 10 milliseconds. As in the previous case, total simulation time is 500s and the flows start at t=100s.

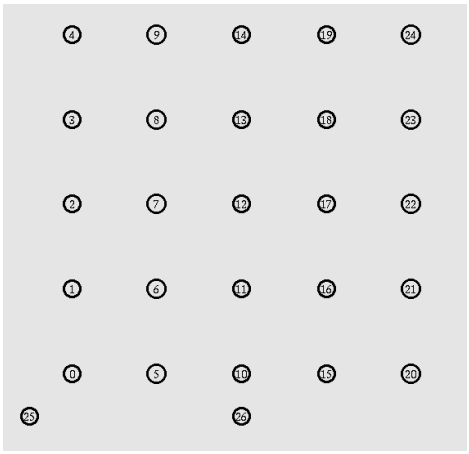


Figure 12. Simulations scenario: FastM with increased mobility

Concerning delivery rate in Figure 13, we observe that the total number of packets received decreases with the increase of velocity. These results are inline with our expectations and reflect the proper operation of the routing protocol and handover approach in the mesh network: mobility decreases the time between handovers, and with more frequent handovers, the losses are increased, which consequently decreases the throughput. One interesting aspect is that delivery rate seems to decrease more rapidly at lower speeds, reaching stabilization at higher speeds. Above 10 m/sec, results show that the protocol is able to maintain some minimal delivery rates. This shows the effectiveness of the neighboring tables: with higher handover frequency, there is less time for tables to be disseminated and normal MeshDV handover process occurs (the effect and enhancement of FastM is minimized). More aggressive table maintenance strategies could help increasing the performance; still, the resulting higher overhead would probably nullify the overall performance improvement. As expected, UDP achieves higher delivery rate than TCP due to the inexistent flow control mechanisms in UDP; however, the trend is similar to both types of communications.

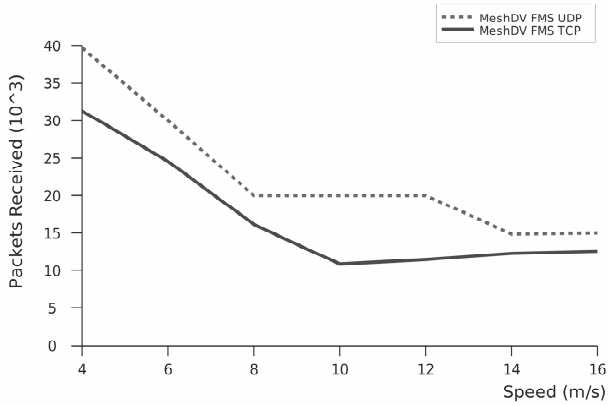


Figure 13. Comparison between throughput in FastM in TCP and UDP as a function of mobile node velocity

According to the results presented in Figure 13, we would expect that overhead would be higher when neighbor tables are being maintained (more maintenance in small mobility scenarios), and this is shown in Figure 14. This graphic depicts the number of control packets sent to the network as a function of speed of the mobile node. For lower mobility, as nodes slowly hop between the WMRs, neighbor tables are filled and propagated to their neighbors, thus producing a higher overhead. As the nodes start to move with a higher speed (and we realize from the graphic that the critical point is at 10 m/s), the neighbor client tables construction becomes more inefficient, since there is less time to propagate the tables to the neighbors, and this reduces total overhead.

Related to the difference between UDP and TCP curves, TCP has, in most of the cases, increased overhead, as in the first scenario depicted in Figure 11. For increased speed, the difference in overhead is not so noticeable: this is due to the mobility effects on TCP sessions.

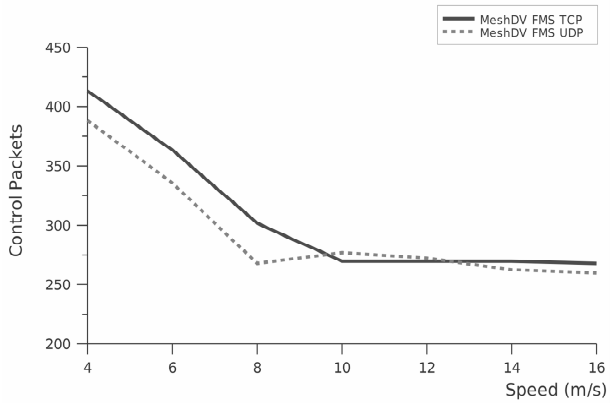


Figure 14. Number of control packets sent to the network as a function of node velocity

6. Conclusions

The support for VoIP applications in current and future WMNs is considered to be vital to its success. However, the wireless medium and multi-hop routing protocols are frequently unable to meet the requirements of seamless mobile nodes handover, while maintaining uninterrupted calls: efficient mobility mechanisms are required to enhance the performance of the communications in these networks in mobile scenarios.

In this paper, we proposed a novel extension to improve the mobility process, denoted as FastM, that integrates mobility and routing mechanisms, and that brings a new way to deal with neighborhoods, using other nodes to maintain information about the organization of the WMN. Results obtained with ns-2 prove the efficiency of the solution and its effectiveness in meeting the requirements of low packet loss, disconnection times, and control overhead, even in scenarios with increased mobility. The disconnection times were reduced to values around 100 milliseconds, being able to meet the real-time communications requirements.

7. Future Work

This mobility mechanism is based in principles that are transversal to other mobility mechanisms that exist nowadays. In this paper this solution was applied to MeshDV; however, its concepts are applicable to most proactive routing protocols. Its support in EMM will be addressed as future work.

New routing processes use concepts related to k-neighborhoods where the algorithm itself organizes the network in several groups, having a number of nodes with a global perspective of the overall network, one for each group. Adapting this concept to the principle behind FastM is intuitive. FastM is supposed to be a mobility management in small perspective, using the relations between neighbors to re-forward packets and adjust the topology when mobility occurs. This way, implementing FastM inside every neighborhood to support micro mobility management and in every group leader to support macro mobility management, it will be created a system with fully support of micro and macro mobility management integration with considerably gains. This integration brings two main advantages: larger routing capacity in large networks and capacity of maintaining k-neighborhoods with high performance. FastM contributes with good results on handovers: no packet loss, low timeout and high performance on route convergence. These two techniques combined add a great robustness to a network when handovers occur. Applying the procedures in very large networks will have no influence in the global performance, as the responsibilities and overloads are distributed between all the k-neighborhoods.

Other future work in this area also concerns the implementation of the mobility approach and the comparison of simulation and experimental results, to assess the behaviour of these mechanisms in real environments. Moreover, a comparison should be made with different mobility solutions.

Acknowledgments

This work was partially supported by the European Commission project IST-WIP under contract 27402.

8. References

[1] L. Couto, J. P. Barraca, S. Sargento, R. L. Aguiar, “FastM in WMN: a Fast Mobility Support extension for Wireless Mesh Networks”, The Second International Conference on Advances in Mesh Networks (MESH 2009), Athens (Greece), June 2009.

[2] IST WIP – An All Wireless Mobile Network Architecture, http://www.ist-wip.org.

[3] L. Iannone and S. Fdida, “Meshdv: A distance vector mobility-tolerant routing protocol for wireless mesh networks”, *IEEE ICPS Workshop on Multi-hop Ad hoc Networks: from theory to reality (RealMAN’06)*, July 2005.

[4] M. Bezahaf, L. Iannone and S. Fdida, “Enhanced Mobility Management in Wireless Mesh Networks”, *Journées Doctorales em Informatique et Réseaux (JDIR08)*, January 2008.

[5] H. Wang, Q. Huang, Y. Xia, Y. WU, Y. Yuan, “A Network-Based Local Mobility Management Scheme for Wireless Mesh Networks”, *In Proc. of WCNC 2007.*

[6] Y. Fan, J. Zhang, X. Shen, “Mobility-Aware Multi-Path Forwarding Scheme for Wireless Mesh Networks”, *In Proc. of WCNC 2008.*

[7] A. Capone, M. Cesana, S. Napoli and A. Pollastro, “MobiMESH: a Complete Solution for Wireless Mesh Networking”, *IEEE*, 2008.

[8] Y. Amir, C. Danilov, et al., “Fast Handoff for Seamless Wireless Mesh Networks”, *In Proc. of MobiSys’06*, June 2006.

[9] S. Speicher, C. H. Cap, “Fast Layer 3 Handoffs in AODV-based IEEE 802.11 Wireless Mesh Networks,” In Proc. of 3rd International Symposium on Wireless Communications Systems (ISWCS),Sept, 2006.

[10] C. Perkins, E. Belding-Royer, S. Das, “Ad hoc On-Demand Distance Vector (AODV) Routing,” RFC 3561, 2003.

[11] G. Koodli, “Fast Handovers for Mobile Ipv6,” RFC 4068, Jul., 2005.

[12] E. Bortnikov, T. Kol, A. Vaisman, “QMesh: a QoS Mesh Network with Mobility Support,” In Proc. of ACM SIGMOBILE Mobile Computing and Communications**,** vol. 12, no. 1, Jan., 2008.

[13] F. Rosseau, Y. Grunenberger, V. Untz, E. Schiller, P. Starzetz, F. Theoleyre, M. Heusse, O. Alphand and A. Duda, “An Architecture for Seamless Mobility in Spontaneous Wireless Mesh Networks”. *in Proc. MobiArch'07, Kyoto, Japan*, August 2007.

[14] F. Rousseau, F. Théoleyre, A. Duda, A. Krendzel, M. Requena-Esteso, J. Mangues-Bafalluy, "Geo-mobility and Location Service in Spontaneous Wireless Mesh Networks," In Proc. of ICT-MobileSummit, Stockholm, Sweden, June 10-12, 2008.

[15] X. Wu, "VPDS: Virtual Home Region based Distributed Position Service in Mobile Ad Hoc Networks," In Proc. of IEEE International Conference on Distributed Computing Systems (ICDCS), June, 2005.

[16] D.Huang, P. Lin, C. Gan, "Design and Performance Study for a Mobility Management Mechanism (WMM) Using Location Cache for Wireless Mesh Networks," IEEE Transactions on Mobile Computing, vol. 7, no. 5, May, 2008.

[17] IEEE Standard for Information Technology-Telecommunications and Information Exchange Between Systems-Local and Metropolitan Area Networks-Specific Requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 2: Fast Basic Service Set (BSS), July 2008.

[18] C.E. Perkins and P. Bhagwat, "Highly Dynamic destination-sequenced distance vector routing (DSDV) for mobile computers." *in Proc. ACM SIGCOMM 94, London, UK*, Oct. 1994, pp. 234-244.

[19] T. Narten, E. Nordmark, W. Simpson, "RFC 2461 – Neighbor Discovery for IP Version 6 (IPv6)".

[20] I. Ramani, S. Savage, "SyncScan: Practical Fast Handoff for 802.11 Infrastructure Networks", *Infocom 05,* March 2005.

[21] The ns-2 Manual (Oct. 2, 2006), http://www.isi.edu/nsnam/ns.

[22] L. Couto, J. Barraca, S. Sargento, R. Aguiar, "Fast Mobility in proactive routing protocols", *ICT-MobileSummit 08,* July 2008.

www.iariajournals.org

International Journal On Advances in Intelligent Systems
ICAS, ACHI, ICCGI, UBICOMM, ADVCOMP, CENTRIC, GEOProcessing, SEMAPRO, BIOSYSCOM, BIOINFO, BIOTECHNO, FUTURE COMPUTING, SERVICE COMPUTATION, COGNITIVE, ADAPTIVE, CONTENT, PATTERNS
issn: 1942-2679

International Journal On Advances in Internet Technology
ICDS, ICIW, CTRQ, UBICOMM, ICSNC, AFIN, INTERNET, AP2PS, EMERGING
issn: 1942-2652

International Journal On Advances in Life Sciences
eTELEMED, eKNOW, eL&mL, BIODIV, BIOENVIRONMENT, BIOGREEN, BIOSYSCOM, BIOINFO, BIOTECHNO
issn: 1942-2660

International Journal On Advances in Networks and Services
ICN, ICNS, ICIW, ICWMC, SENSORCOMM, MESH, CENTRIC, MMEDIA, SERVICE COMPUTATION
issn: 1942-2644

International Journal On Advances in Security
ICQNM, SECURWARE, MESH, DEPEND, INTERNET, CYBERLAWS
issn: 1942-2636

International Journal On Advances in Software
ICSEA, ICCGI, ADVCOMP, GEOProcessing, DBKDA, INTENSIVE, VALID, SIMUL, FUTURE COMPUTING, SERVICE COMPUTATION, COGNITIVE, ADAPTIVE, CONTENT, PATTERNS
issn: 1942-2628

International Journal On Advances in Systems and Measurements
ICQNM, ICONS, ICIMP, SENSORCOMM, CENICS, VALID, SIMUL
issn: 1942-261x

International Journal On Advances in Telecommunications
AICT, ICDT, ICWMC, ICSNC, CTRQ, SPACOMM, MMEDIA
issn: 1942-2601