Hierarchical Tree Alternative Path (HTAP) Algorithm for Congestion Control in Wireless Sensor Networks

Charalambos Sergiou, Vasos Vassiliou, Aristodemos Paphitis

Computer Science Department
University of Cyprus,
1 University Ave., 2019 Nicosia, Cyprus
Email: {sergiou, vasosv, csp7pa2}@cs.ucy.ac.cy

Abstract

Recent advances in wireless sensor networks (WSNs) are leading to applications with increased traffic demands. Research is evolving from applications where performance is not considered as a crucial factor, to applications where performance is a critical factor. There are many cases in the fields of automation, health monitoring, and disaster response that demand wireless sensor networks where performance assurances are vital especially for parameters like power, delay and reliability. Due to the nature of these networks the higher amount of traffic is observed when the monitored event is taking place. Exactly at this instance, there is a higher probability of congestion appearance in the network. Congestion in WSNs is tackled by the employment of two methods; either by reducing the load ("traffic control"), or by increasing resources ("resource control"). In this paper we present the Hierarchical Tree Alternative Path (HTAP) algorithm, a "resource control" algorithm that attempts through simple steps and minor computations to mitigate congestion in wireless sensor networks by creating dynamic alternative paths to the sink. HTAP is evaluated in several scenarios in comparison with a state of the art "resource control" algorithm (TARA), as well as with a "traffic control" algorithm (SenTCP) and the case where no congestion control applies in the network ("no CC"). Results prove that HTAP is a simple and efficient algorithm capable of dealing successfully with congestion in WSNs, while preserving the performance characteristics of the network.

Keywords: Wireless Sensor Networks, Congestion Control, Alternative Path Routing, Energy Efficiency

1. Introduction

A wireless sensor network (WSN) is a network composed of many sensor nodes capable of sensing a phenomenon, transforming the analog data to digital and transmitting them to destination nodes (usually called sinks). Due to severe power constraints their computation capability, as well as their transmission range, are limited. Thus, for the transmission of data from a source (the node that sensed the phenomenon) to a sink (the end-node that receives the data), the wireless sensor nodes that lie over between them , form a "path" and data are transmitted through them in a hop-by-hop manner. Frequently, sensor nodes are densely deployed near the event sources and sinks in a redundant manner [1]. A WSN comprises of a potential large set of nodes that may be distributed over a wide geographical area indoor or outdoor. These networks enable numerous sensing and monitoring services in areas of vital importance such as efficient industry production, safety and security at home, and traffic and environmental monitoring. Traffic patterns in WSNs can be derived from the physical processes that they sense. WSNs typically operate under light load and suddenly become active in response to a detected or monitored event. Depending on the application this can result in the generation of large, sudden and correlated- synchronized impulses of data that must be delivered to a small number of sinks without significantly disrupting the performance (i.e fidelity) of the sensing application. This high generation of data packets is usually uncontrolled and often leads to congestion (overflowed buffers or packet collisions in the medium). There are a number of protocols proposed in literature for congestion control in Wireless Sensor Networks. A detailed analysis of a number of them can be found in [2] and [3].

In this work we present in detail the Hierarchical Tree Alternative Path (HTAP) algorithm, discuss the improvements we did over its original form and analyze its performance under varying conditions. HTAP is an algorithm that attempts to face congestion in WSNs by employing alternative paths for the avoidance of congested areas (or nodes) in the network. The HTAP algorithm was firstly presented in [4], while a performance study of the energy utilization of HTAP under different node placements has been presented in [5]. The bigger advantage of HTAP algorithm as presented in [4] and [5] is the fact that it is a very simple algorithm which actually does not add any significant overhead to the already heavy loaded network in case of congestion. On the other hand, a small problem of this algorithm

is the fact that the delay for the transmission of packets to sink through alternative paths is increasing under specific cases. In order to overcome this problem we have designed and implemented two specific features, in this new version of the algorithm, which renders HTAP as a complete and robust congestion control algorithm that can face congestion situations effectively and efficiently. Specifically, the first feature added to this algorithm is a topology control scheme based on a variation of the Local Minimum Spanning Tree algorithm (LMST)[6] where each node builds its local minimum spanning tree independently, using Prim's algorithm [7] and keeps as neighbors in the tree only nodes that are one hop away. The maximum number of nodes that a node can be connected to, in our implementation, is six. The second important feature that is added to HTAP is the ability of nodes to recognize "deadlocks". If a node recognizes that there are no other nodes at a level higher than itself (closer to the sink) able to forward packets to the sink, it informs, through a "negative ACK" packet (NACK), the nodes using it as a relay node for this situation. Therefore, certain nodes are removed from the routing tables and the risk of transmitting packets to dead-end paths is avoided. Furthermore, HTAP algorithm introduces a novel adaptive method for inferring congestion in the network. This adaptive method, uses buffer occupancy as a first indication of congestion occurrence and then it employs the ratio of out/in data rate in order to trigger the alternative path creation.

To confirm the improvement in the performance of HTAP, the algorithm has been extensively tested compared to an established algorithm that also employs a "resource control" method for the transmission of packets from source to sink, called Topology- Aware Resource Adaptation (TARA) [8]. Simulation results prove that the HTAP algorithm presents a better performance attitude, in the range of 2-4%, in all examined parameters in comparison with TARA, while it is evidently much easier to be implemented and introduces significantly less overhead than TARA. For completeness, we also consider a "legacy" traffic control algorithm [9] and the case where no congestion control is used. We believe that this work presents a significant advancement in the area of congestion control in WSNs since it provides a simple, robust, effective and efficient solution to this problem.

2. Background

2.1. Congestion in WSNs

Congestion in WSNs is more complicated in comparison to conventional networks since it may appear in different types and different places. Concerning the type of congestion this can be categorized in two types [8]:

Type H1 Congestion - In a particular area, many nodes within range of one another attempt to transmit simultaneously, resulting in losses due to interference and thereby reducing the throughput of all nodes in the area. We note that explicit local synchronization among neighboring nodes can reduce this type of loss, but cannot eliminate it completely because non-neighboring nodes can still interfere with transmission.

Type H2 Congestion - Within a particular node, the queue, or buffer used to hold packets to be transmitted overflows. This is the conventional definition of congestion, widely used in wired networks. This is also the main cause of packet losses. Type H2 assumes the existence of an effective MAC protocol which is able to transmit packets from different sources without collisions.

Concerning where in the network the problem happens, three categories exist [10]:

Type W1 Congestion - Hotspot near the source (transient) - Source Congestion. Densely deployed sensors generating data events during crisis state will create persistent hotspots very close to the sources (e.g., within one or two hops. In this scenario, localized, fast time scale mechanisms capable of providing backpressure from the points of congestion back to the sources would be effective. Also local de-synchronization of sources would be effective too.

Type W2 Congestion - Hotspot near the sink (Persistent) -Sink Congestion. Even sparsely deployed sensors that generate data at low data rates can potentially create transient hotspots anywhere in the sensor field, but more likely farther from the sources and near the sink. Fast time scale resolution of localized hotspots using a combination of localized back-pressure and packet dropping techniques would be more effective, in this case. Source nodes may not be involved in the backpressure because of the transient nature of the problem in this situation. Also an effective way of alleviating sink congestion is to deploy multiple sinks that are uniformly scattered across the sensor field, and then balance the traffic between these sinks.

Type W3 Congestion - Forwarder Congestion. A sensor network will have more than one flow (sink-source pair), and these flows will intersect with one another. The area around the intersection will likely become a hot spot. In a tree-like communication paradigm, every intermediate node in the tree can suffer from forwarder congestion. Compared to TYPE W1 and W2, forwarder congestion is far more challenging because it is very difficult to predict the intersection points due to the network dynamics. In this case even sparsely deployed sensors generating data will create both transient and persistent hotspots distributed throughout the sensor field. A combination of fast time scale actions to resolve localized transient hotspots, and closed loop rate regulation of all sources that contribute toward creating persistent hotspots seems to be effective. Resource provisioning techniques could be used when "traffic control" methods cannot meet application's requirements.

2.2. Congestion Control and Mitigation Methods in WSNs

Generally, the presence of congestion means that the load is (temporarily) greater than the resources (in a part of a sensor network) can handle. In such a case the following control schemes may be used: decrease the load, increase the resources, or employ MAC layer enhancements. MAC layer enhancements could help more in the direction of Type H1 congestion. If the packet generation rate is sufficiently small, simultaneous transmission of packets becomes independent of the rate. Rather, it depends on the time at which each node generates the packet. A good way to reduce this type of congestion is to perform phase shifting, an observation made by the authors in [11] . Small amounts of phase shifting can be performed by introducing slight jitters at the data-link layer. In [11] also the application layer itself introduces phase shifts. While jittering at the data-link layer aims to cause small transmission variations between neighboring nodes, we think, that phase shifting at a higher layer can be achieved on a larger time scale. To handle Type H2 congestion one may employ the other two methods, a) decreasing the load (traffic control) or b) increasing resources (resource control) as these would help in emptying the buffers of intermediate sensor nodes. An analysis of these two methods is presented below:

Traffic Control Method: By employing a traffic control method, affected nodes, inform the nodes that forward packets to them, to reduce the rate with which they are transmitting packets, since they will soon become overloaded. To achieve data rate reduction, affected nodes, use backpressure messages which are relayed backwards to the data sources until the data rate

is controlled. While this method can be effective in cases where the total load in the network is heavy, or in specific scenarios like Type W1 and Type W2, it cannot be effective in the case of Type W3 congestion. In W3 congestion it is possible, a number of flows with low data rates to transmit data at a specific intermediate node and congest it. By acting on every flow individually, flows will be committed to reduce the, already low, data rate they transmit with, while the remaining network will be under-loaded. Another significant disadvantage of this method is the fact that in WSNs a heavy load is usually produced when an event is taking place. In such a case, reducing the data rate could negatively affect the mission of the network. Traffic control is the method employed by the majority of congestion control algorithms in WSNs [10] [11] [12] [9] [13] [14] [15] [16] [17] [18].

Resource Control Method: The resource control method operates in a different way in comparison with traffic control. In this method, due to the redundant deployment of nodes in the network, affected sources attempt to employ nodes which are not involved in the current flow for sending packets, from a source to a sink, in order to maintain the rate with which data are flowing. The advantage of this method is especially obvious when Type W3 congestion is experienced, since flows that are merging to a specific node are possible to use other paths to the sink. Thus, reducing the data rate of all sources may not be the best solution. However, if different paths are chosen for one or more of the interacting flows, the data rate will not be reduced and the final network throughput will be increased. Of course, in this method it is not possible for each node to arbitrarily decide which flows to reject and for rejected flows to randomly choose which nodes to join. This situation is certain that it will worsen the situation. Therefore, a number of parameters, including the type of congestion and the type of the application, must be taken into consideration before applying a resource control method. This method has not attracted as much attention as the traffic control counterpart. The most representative work that employs "resource control" method is [8]. Also [19], [20], [21], [22] employ in different manners a "resource control" scheme. The HTAP algorithm which is presented in this paper is also a "resource control" algorithm.

3. Network Model and Problem Description

We assume that there is a densely deployed WSN, where nodes are initially deployed uniformly. One sink is located at a specific point in the net-

work topology, while the number of source nodes is variable since an event is possible to be random and captured by more than one nodes. We also consider that each node knows its position and the position of the sink node(s) in the grid. Localization and positioning in WSNs is a subject that attracted a lot of research work [23] and it is beyond the scope of this work. In our network deployment we consider that all nodes beside sink are identical and CSMA/CA is employed as the MAC protocol.

The problem that we attempt to counter in this paper is to render a network capable to deliver to the sink(s) all (or almost all) of the data packets which have been created without intervening on the rate with which sensor nodes are injecting these packets in the network. This means that this network should be able to detect possible congestion nodes (hot spots) and to avoid them. To achieve this target we need to employ nodes which are not in the initial route from the source to the sink. For this purpose we have designed the Hierarchical Tree Alternative Path (HTAP) algorithm. HTAP is a dynamic algorithm which, besides congestion control, also attempts to maintain the network's performance characteristics which are also critical for the network's successful operation.

4. Hierarchical Tree Alternative Path (HTAP) Algorithm

In this work we propose and evaluate a congestion control algorithm, which bases its functionality on the creation of alternative paths from sources to sinks in order to prevent congestion from happening. HTAP is a dynamic congestion control algorithm that bases its path switching decision on local information, such as the congestion state of its neighbors.

HTAP consists of four different schemes:

- Topology control
- Hierarchical Tree Creation
- Alterative Path Creation
- Handling of Powerless (dead) nodes

These four schemes are described below:

4.1. Topology Control

Topology control is crucial in WSNs since it can handle issues arising from the redundant number of nodes and their dense deployment. Problems like interference, maximum number of possible routes, use of maximum power to communicate to distant nodes directly e.t.c. are possible to arise. Since HTAP is an algorithm which attempts to employ the network's extra resources (unused nodes), it is obvious that guaranteeing a redundant number of paths is essential. In order to maintain the performance characteristics of the network in case of congestion, these paths must be carefully selected. Topology control is, therefore, the first scheme applied in the HTAP algorithm.

An effective topology control algorithm should be able to preserve connectivity with the use of minimal power, while maintaining an optimum number of nodes as neighbors to each node. In this work we employ, with a variation, the Local Minimum Spanning Tree algorithm (LMST) [6] as the initial topology control that runs on the network. LMST is an algorithm capable of preserving the network connectivity using minimal power, while the degree of any node in the resulting topology is restricted to six. As it is analytically explained in [6], this feature (6 neighbors per node) is desirable because a small node degree reduces the MAC-level contention and interference. In LMST each node builds its local minimum spanning tree independently, using Prim's algorithm [7] and keeps on the tree only those neighboring nodes which are one hop away. In addition, the resulting topology is possible to use only bi-directional links, a matter which, as we will explain later, is valuable for the successful operation of HTAP.

The variation introduced to LMST by HTAP relates to the selection of the neighbor list. Instead of selecting any node that fulfils the criteria as neighbor (maximum six), the modified LMST used by HTAP keeping as neighbor nodes only those that reside one level closer to the sink than itself.

4.2. Hierarchical Tree Creation

The hierarchical tree creation algorithm runs over the topology control algorithm and only at the moment, where a node becomes a source (starts sensing a phenomenon). This algorithm consists of two main steps:

4.2.1. Path Creation

In this step a hierarchical tree is created beginning at the source node. After the end of the topology control phase each node is able to be connected

to at most six nodes which are only one hop away from itself. During this phase each node that is becoming a source node is self-assigned as level 0 and sends a *level_discovery* message to the six neighbors selected during topology control phase. Nodes that receive this packet are considered as children to the source node and are set as level 1. Each of these nodes broadcast again the *level_discovery* packet, and the pattern continues with the level 2 nodes etc. This procedure iterates until all nodes are assigned a level and stops when the *level_discovery* packets reach the sink.

When the procedure finishes it is possible that the sink receives more that one *level_discovery* packets from different nodes and each packet may have a different level value. This is an indication that disjoint paths are reaching the sink.

The hierarchical tree algorithm is also able to identify and rectify some issues that are possible to arise. Specifically, it is possible for a node to be the last one that receives the *level_discovery* packet, since there are not any other nodes upstream able to forward that packet. In such a case this node broadcasts a "negative ACK" packet (NACK) indicating that for this reason it cannot route any packets. When the node, that forwarded the packet, receives the NACK it is aware that it will not route any packets through this node.

An example of the operation of hierarchical tree algorithm, and placement of nodes in levels, is illustrated in Figures 1a and 1(b).

Figure 1(a) represents the networks connectivity after the LMST topology control algorithm applies.

After topology control algorithm applies, level placement procedure takes as input this topology and attempts to place nodes in levels from each source to sink.

In this example, node 1 is considered as the source. Thus, when node 1 becomes source, it broadcasts a message to the nodes that it can connect to, according to the results of the topology control algorithm. In this case, nodes 2,3 and 4 receive this message and are assigned as level 1 nodes for this source (node 1). Then these nodes (2,3 and 4), transmit this message to the nodes that they are connected to, and these nodes are assigned as level 2 nodes (5,6,7 and 8). This procedure iterates until packets reach sink.

During this procedure, if a node receives a packet from more that one nodes, then it keeps connection only with the node that assigns it the higher level. For example node 7 is connected, after the end of topology control

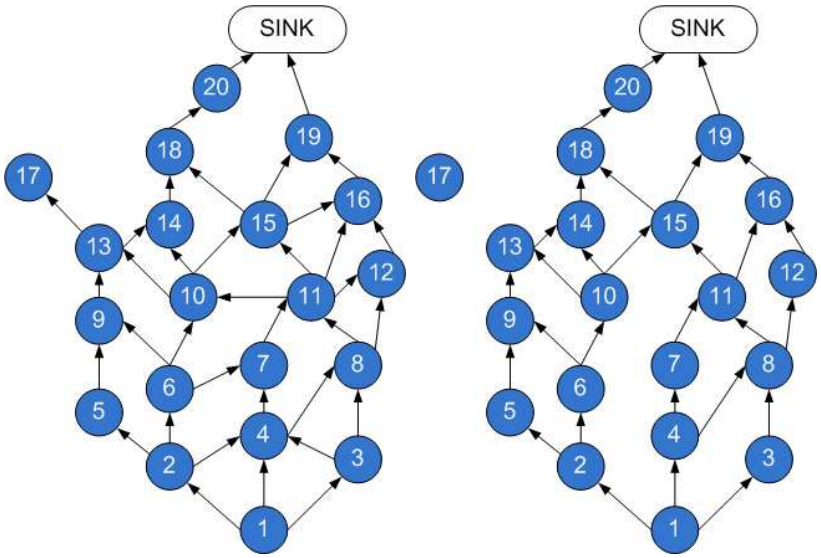

Figure 1: (a) Network Connectivity after Topology Control, (b) Level Placement Procedure

phase, with nodes 6 and 4. During level placement procedure, it will receive a packet form node 6, that will ask it become a level 3 node and from node 4, that will ask it to become a level 2 node. In this case, node 4, will become a level 2 node and will ask from its neighbor nodes to become level 3 nodes.

Finally, in this figure, node 17 represents the case where a node does not have any upstream to transmit a packet. In such a case, level placement algorithm removes this node from the table of upstream nodes of node 13.

4.2.2. Flow Establishment

A connection is established between each transmitter and receiver pair using a two-way handshake. Packets are exchanged between each transmitter and receiver in the network, in order to get connected. Through this packet exchange, the congestion state of each receiver is communicated to the transmitter. Let us consider again Figure 1, where node 1 is the source and nodes 2, 3, and 4 are receivers. Firstly, node 1 sends a packet to node 2. When node 2 receives this packet, it sends an ack packet back to 1. In this ack packet, node 2 piggybacks its current congestion state. This exchange makes the source node aware of the congestion state of all its next hop neighbors that can overhear. When the congestion state of children reaches a pre-specified limit these nodes update their congestion state and inform node 1 using the

bidirectional link.

4.3. Alternative Path Creation

This algorithm runs when congestion is possible to occur at a specific node in the network. Since the employed topology control algorithm is able to counteract collisions in the medium by choosing the smallest transmission power (one hop nodes), congestion is still possible to happen when a node receives packets with a higher rate than it can transmit (buffered based congestion). In a wireless sensor network where all nodes, except the sink, are exactly the same, this can happen if a node is receiving packets from at least two flows, or if the nodes to which it has to transmit packets to, cannot accept any more packets.

When the buffer of a node starts filling, this node has to take action. In such a care each node is programmed to run locally a lightweight congestion detection (CD) algorithm. When the buffer reaches a buffer-level threshold value, the CD algorithm starts counting the rate with which packets are reaching the node. Since each packet is identified by the NodeID in its packets header, the CD algorithm is aware of the nodes that are transmitting packets through this node, as well as their data rate.

$$\sum_{i=1}^{k} Rx_i \geq Tx_{max} \tag{1}$$

By using Equation 1, the CD algorithm is able to calculate the total receiving rate ($\sum_{i=1}^{k} Rx_i$) and compare it with the maximum rate that is able to transmit (Tx_{max}). When this ratio is large (and above a certain percentage) the node sends a backpressure message to the nodes that keep transmitting packets through it to search for an alternative path. The selection of these nodes is performed firstly from those that transmit with the lower rate. The purpose of this tactic is to maintain the performance characteristics of the network (keep the throughput of nodes at the maximum possible level without packet drops) and minimize the impact of the change to the network.

When a node is informed through the bi-directional link to stop transmitting through the specific node, it searches in its neighbor table and finds the next available node with the higher level (in comparison with the congested node) and starts transmitting through it. If this node is in the transmitting

range of the congested node, is already aware of its condition and does not use it either. This tactic, results in the relief of a congested node.

An advantage of HTAP in comparison with similar schemes like [8] and [24] is the fact that it does not employ specific nodes as distributors and mergers. Through the advantage that is offered from the use of the topology control algorithm and the source based hierarchical tree, each node is able to inhibit the transmission of packets through itself and also it is able to join the first available shortest path, after path alternation.

4.3.1. Congestion Threshold

As mentioned above, a key point to the operation of the HTAP algorithm is the value of the congestion threshold. Nodes, using equation 1, calculate the total receiving rate and compare it with the maximum transmission rate they have at the moment. The issue that arises is that both parameters (receiving and sending rate) are heavily dependant on the current network situation. Thus, it is possible for a node to receive a large number of packets in a short period of time and then to keep receiving packets at a lower (normal) rate. A node is then congested (in terms of occupied buffer space) but actually is not experiencing any problems that force it to control this overload situation. Therefore, the parameter that needs to be tuned is not just the value of the threshold, but also the duration where this threshold is exceeded (burst period). If the duration is set too low then the "alternative path creation" algorithm will be triggered often. In cases when the situation is transient the creation of alternative paths would add unnecessary overhead to the network (delays and power consumption). On the other hand, if the burst period is very big, buffer overflows will occur and nodes will react overdue to this situation.

HTAP handles this issue by using an adaptive method. Initially, buffer monitoring begins when the buffer occupancy of each node reaches 50% of the total. At this point the affected node counts the number of nodes from which is receiving packets. Then it assumes that each node is transmitting with the maximum data rate and calculates the time until the buffer occupancy will reach the 85% limit. When this time elapses it checks again the occupancy of the buffer. If it is between 80 and 85% it considers that, indeed, is receiving packets with a higher rate than it can transmit and it triggers the "alternative path" algorithm, in order to avoid congestion. If the buffer occupancy is less than 80% it re-calculates the remaining buffer and adjusts accordingly the time, which obviously is severely reduced. If at the next measuring epoch

the buffer occupancy is still below 80%, the first threshold is adjusted from 50% to 70%. Such behavior can happen in areas where a permanent event is taking place. When a permanent event is affecting the network, is expected, that by setting the buffer to just 50% will contribute to the overhead.

Let us consider that our network consists of nodes with a buffer size (B) of 128Kbytes and a maximum data rate (r) of 128Kbps. We also consider that a node is receiving data from five different nodes (n). In this case when the 50% of the buffer (64Kbytes) is full, node begins the process. It counts the number of nodes from which is receiving data (in this case is five) and considers that they transmit with maximum data rate.

Thus, it calculates the time t which is t=B/(n*r)=64Kbytes/(5*128kbps)=0.8s. This means that in 0.8s the CD algorithm will check again the buffer occupancy. If the buffer occupancy is between 80 and 85% it will trigger the "alternative path" algorithm. If it is less e.g 65% then it will measure again the number of nodes that transmit packets and will adjust the time accordingly. Thus, if we consider that now is receiving packets from four nodes instead of five, it will calculate the remaining buffer up to 85 % which is now 20% as well as the maximum data rate of 4 nodes. This will give a new time and algorithm will check buffer occupancy after this time elapses. If it remains below 70% but higher than 50% it will set the first threshold to 70% and will stop monitoring this node until buffer occupancy exceeds this threshold (70%). By employing this extended but lightweight congestion detection scheme the HTAP algorithm is able to face both permanent and transient congestion situations successfully.

4.4. Handling of Powerless (dead nodes)

Special care is taken in the HTAP algorithm concerning the nodes which are power exhausted. These nodes cause major problems to the network in case they act as sources or relay nodes. Thus, when a node is about to get power exhausted, it should immediately be extracted from the network and the tables of its neighbor nodes should be updated. This procedure should be as simple as possible due to the fact that this can happen when the network is in a crisis state. When the power of a node reaches the "power extinction" limit, it immediately broadcasts this fact to the nodes around it. The nodes that receive this packet extract this node ID from their neighbor list. If this node is a part of an active path (a path that is relaying packets to sink), the nodes that are sending packets to this node and receive "power extinction"

message, apply the "alternative path" algorithm and find another path to forward packets to the sink.

5. Description of Evaluated Algorithms

Before starting the performance evaluation section (section 6) we provide a short description of the algorithms considered against HTAP. These algorithms are TARA [8] and [9]. TARA is chosen as the most representative congestion control algorithm in WSNs that employs a "resource control" method for congestion control. On the other hand SenTCP [9] is an established congestion control algorithm in WSNs that employs the "traffic control" method.

TARA: Kang et al. proposed Topology Aware Resource Adaptation (TARA) protocol. TARA focuses on the adaptation of network's extra recourses in case of congestion, for alleviating intersection hot spots. TARA copes with buffer occupancy as well as channel loading. In TARA, congestion alleviation is performed with the assistant of two important nodes. These are the distributor and the merger. Between them a"detour path" is established, starting at the distributor and ending at the merger. The distributor, distributes the traffic coming from the hot spot between the original path and the detour path, while the merger merges the two flows. In case of congestion and creation of a hot-spot, traffic is deflected from the hot-spot through the distributor node along the detour and reaches the merge node, where the flows are merged. When congestion is alleviated the network stops using the detour path. For quick adaptation, the distributor node keeps track of which neighbor is on the original path.

SenTCP: Wang et al. proposed SenTCP. SenTCP is an open-loop hop-by-hop congestion control protocol with two special features: 1) It jointly uses average local packet service time and average local packet inter-arrival time in order to estimate the current local congestion degree in each intermediate sensor node. The use of packet arrival time and service time not only precisely calculates the congestion degree, but effectively helps to differentiate the reason of packet loss occurrence in wireless environments, since arrival time (or service time) may become small (or large) if congestion occurs. 2) It uses hop-by-hop congestion control. In SenTCP, each intermediate sensor node will issue feedback signal backward and hop-by-hop. The feedback signal, which carries local congestion degree and the buffer occupancy ratio, is used for the neighboring sensor nodes to adjust their sending rate in the transport

layer. The use of hop-by-hop feedback control can remove congestion quickly and reduce packet dropping, which in turn conserves energy.

6. Performance Evaluation

To evaluate the performance of HTAP algorithm a series of simulations have been performed using the Prowler [25] simulator, a probabilistic wireless network simulator. Prowler provides a radio fading model with packet collisions, static and dynamic asymmetric links, and a CSMA MAC layer. In the first series of simulations, we evaluate the performance of HTAP in comparison with three other algorithms. The first is "no congestion control" (No CC) algorithm which represents the case where "no congestion control" is employed in the network. The second is TARA [8] and the third is SenTCP [9]. Also we present results in comparison with the initial version of HTAP the HTAP v1.

In the second series of simulations we evaluate the performance of HTAP, when network is deployed under different placements. HTAP is an algorithm that depends on the number of possible alternative paths in order to mitigate congestion. Node placement is a parameter that can assist in this effort. The employed placements are Grid, Random, Biased Random and Simple Diffusion placements. These placements will be explained later in the paper.

6.0.1. Simulation Environment and Setup

To perform our simulations we have used the Prowler simulator [25]. The radio propagation and transmission models are given by:

$$P_{rec,ideal}(d) \leftarrow P_{transmit}\frac{1}{1+d^{\gamma}} \tag{2}$$

where, $2 \leq \gamma \leq 4$ and

$$P_{rec}(i,j) \leftarrow P_{rec,ideal}(d_{i,j})(1+a(i,j))(1+\beta(t)) \tag{3}$$

where $P_{transmit}$ is the signal strength at the transmitter and $P_{rec,ideal}(d)$ is the ideal received signal strength at distance d, a and β are random variables with normal distributions $N(0,\sigma_a)$ and $N(0,\sigma_\beta)$, respectively. A node j can receive packets from node i if $P_{rec}(i,j) > \Delta$ where Δ is the threshold.

In our simulations we use the following default simulator parameters:

- $sigma_a = 0.5$

- $sigma_\beta = 0.03$
- $p_{error} = 0.05$
- $\Delta = 0.1$

The rest parameters reflect Mica-Z node specifications, the most important of which are presented in Table 1.

Table 1: Simulation Parameters

Max Data Rate (kbps)	250
Transmission Power (dbm)	2
Receive Threshold (dbm)	-74
Transmission Current (mA)	17.4
Receive Current (mA)	19.7
Fragment Size (bit)	1024
Buffer Size(Bytes)	512K
MAC layer	CSMA/CA

6.1. Performance Evaluation of HTAP in comparison with the other Algorithms

In the following scenarios 100 nodes were uniformly (randomly) deployed on a 500m x 500m grid. The sink was set in the upper right corner of the grid while sources where set in bottom left corner of the grid in order to create congestion conditions. Each simulation run has been performed 20 times and average results have been extracted. In the results we present the standard deviation limits of each point. Also, as we stated above, in this series of experiments we present the performance of the initial version HTAP presented in [4] for comparison purposes. We denote this initial version of HTAP as "HTAP v1".

6.1.1. Percentage of Successfully Received Packets

The first performance metric we have examined is the percentage of successfully received packets (Eq. 4). This metric is particularly important for critical/emergency applications, where every packet has to be received by the sink, and provides the ability of algorithms to control congestion.

$$ReceivedPktsRatio(\%) = \frac{SuccessfullyReceivedPkts}{TotalPktsSent} \quad (4)$$

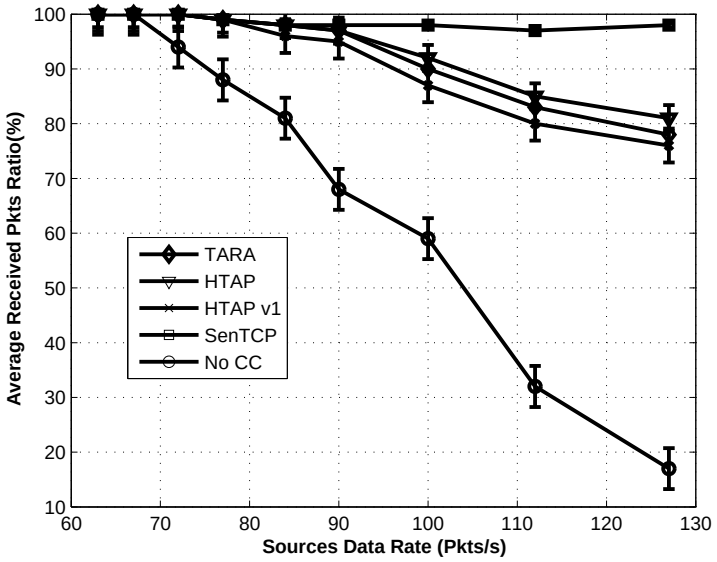

Figure 2: Percentage of Successfully Received Packets

Analyzing the results of Figure 2 we notice that SenTCP which is a "traffic control" algorithm maintains the percentage of successfully received packets near 100% while HTAP and TARA start reducing their percentage after a specific point (when the source data rate exceed 90 packets/s). On the other hand, if no congestion control algorithm is applied, the number of successfully received packets decreases dramatically as the load increases in the network. This fact is expected since the network becomes overloaded and packets are dropped either due to collisions or due to full buffers. Concerning HTAP and TARA we suspect that the reason they start presenting this performance is due to the fact that all resources in the network are fully utilized. This leads to improper operation of resource control algorithms due to lack of resources. In this specific simulation series we note the performance of HTAP algorithm is at an average of 2.5% better than TARA while the performance of HTAP v1 was initially worse than TARA.

6.1.2. Network Throughput

To further study this attitude we count the actual number of packets that are received by the sink (throughput). This metric is a strong indication of the ability of the algorithms to transmit a satisfactory number of packets to sink. Results are presented in Figure 3.

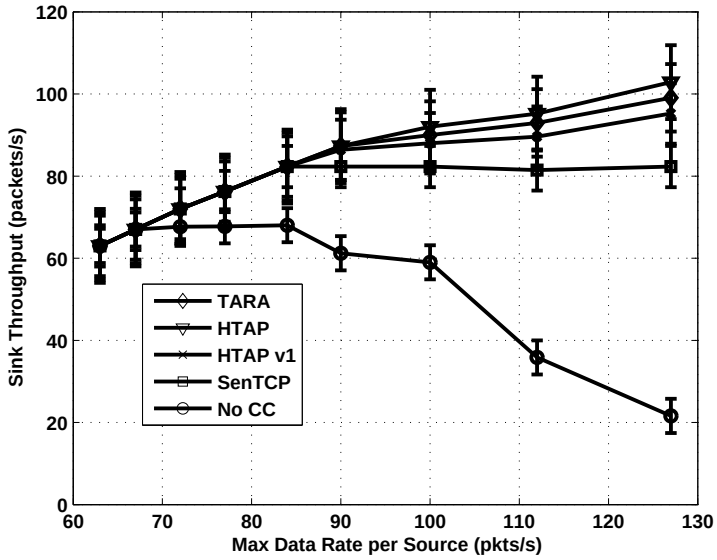

Figure 3: Thoughput

This figure indicates that although the percentage of received packets is decreasing for HTAP and TARA, compared to SenTCP, the actual number of packets that is received by the sink is increasing. This fact is a strong indication that "resource control" algorithms can provide the sink with a large number of data packets even when the load in the network is very high. On the other hand, SenTCP lowers the number of packets transmitted to the sink even if an application requires all data to be delivered in order to work properly.

6.1.3. Increasing Resources

In order to check how the number of resources affect the performance of "resource control" algorithms we keep the node data rate at 128 packets/second and in each run we increase the number of nodes. Results are presented in Figures 4 and 5.

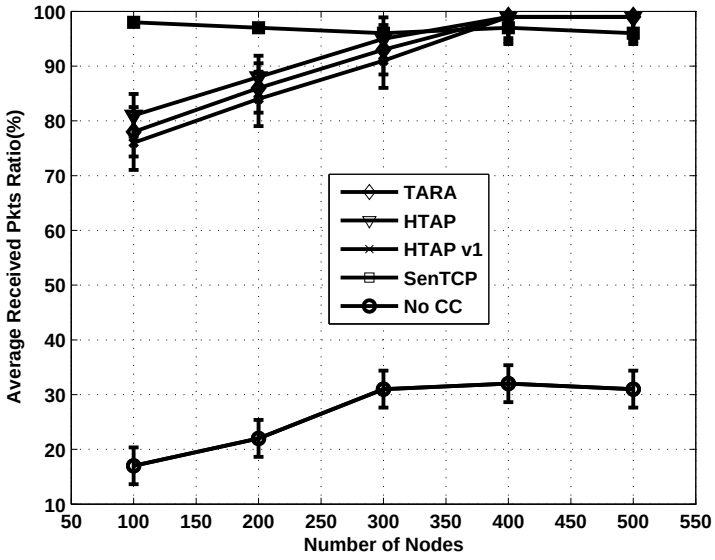

Figure 4: Average received rackets ratio (%) with increasing number of nodes

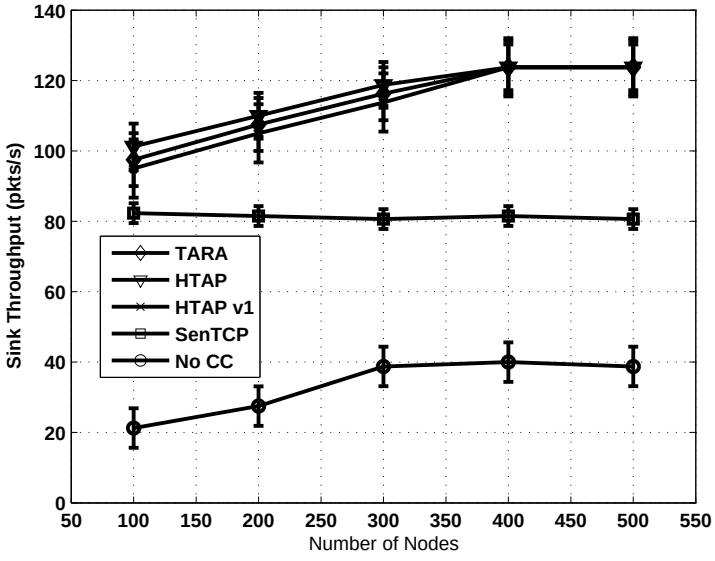

Figure 5: Average throughput with increasing number of nodes

Figures 4 and 5 indicate that as the number of nodes in the network increases "resource control" algorithms improve their performance and are able to deliver a bigger number of packets to sink in comparison with "traffic control" algorithms like SenTCP. We also notice that as resources are increasing, HTAP and TARA present similar performance, which is an indication that their "resource control" mechanisms can operate properly, giving maximum performance. The "no congestion control" case is also slightly improved but again its performance is very low.

6.1.4. Average Hop-by-Hop Delay

The next metric that we evaluate is the average hop-by-hop delay. This metric is an indication of how algorithms handle inter-path interferences, link layer retransmissions, and the overhead introduced by control packet exchanges.

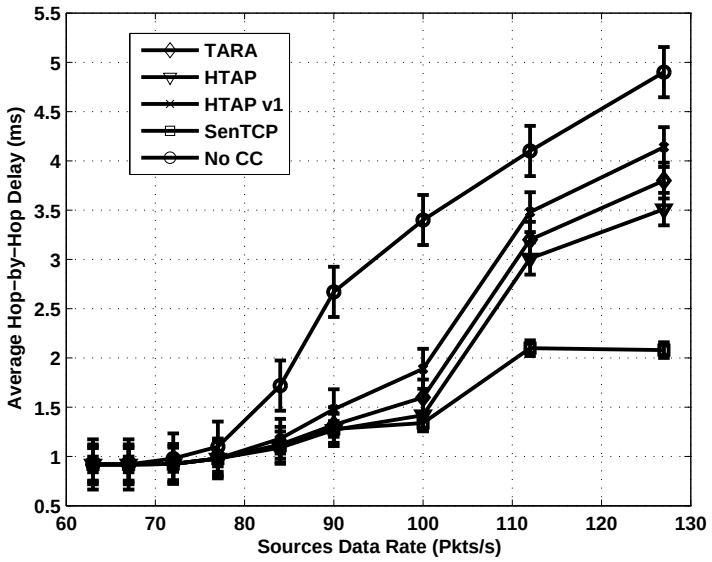

Figure 6: Average hop-by-hop delay

Results in Figure 6 depict that, when congestion occurs (near 100 packets/s) SenTCP starts reducing the data rate in the network (i.e. performs traffic control) and the delay is getting stable. This reaction is expected since the load in the path is not increasing. On the other hand HTAP and TARA seem to have an increased delay after 100 packet/s since the load in

the network is increasing and an attempt is made so that all packets reach the sink. Packets drops that happen at this phase (also according to Figure 2) justify this attitude. Between HTAP and TARA we notice that HTAP presents less delay in the range of 2-3%. This happens because in HTAP, nodes exchange less control messages in comparison with TARA and the algorithm introduces less overhead. The performance of HTAP is also much better than HTAP v1. This means that topology control algorithm improved the performance of HTAP v1 algorithm.

6.1.5. Energy Consumption

Then we study total energy consumption of nodes during a congestion period.

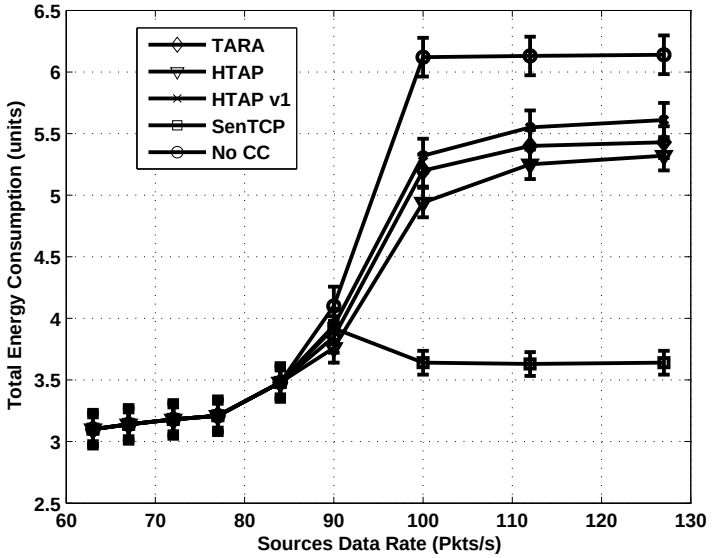

Figure 7: Total energy Consumption

As it is expected SenTCP presents less energy consumption since less packets are injected in the network. SenTCP presents its bigger energy consumption, at the point where congestion occurs. This fact is justified by the operation of SenTCP, since until congestion is controlled, the nodes' overhead in increasing. When there is no congestion control the consumed energy increases quickly and then stabilizes, since the network has reached

its maximum capacity. On the other hand, TARA and HTAP present increased energy consumption since many more nodes are now employed for the transmission of packets from the source to the sink. As we can see from Figure 3 this energy consumption is rather negligible in comparison with the achieved throughput. Again, between the two "resource control" algorithms HTAP is 2-3% better than TARA and better than HTAP v1.

6.1.6. Network Remaining Energy

Finally we study the percentage of the network's remaining energy (network lifetime) after the network is not able to transmit a single packet anymore from the source to the sink due to a lack of available paths. This metric is an indication of the effectiveness of algorithms to utilize uniformly the resources of the network. In this simulation series each algorithm run over the same topology for different numbers of nodes (100, 200, 300). In each set the simulations were running until the network became disconnected.

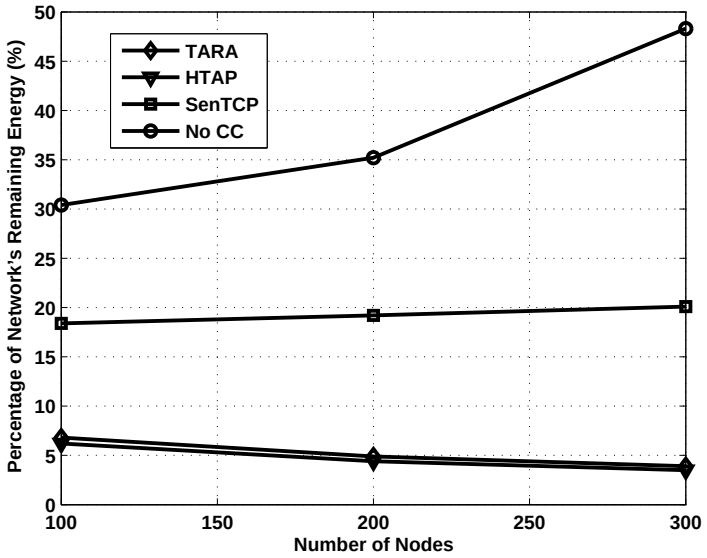

Figure 8: Network's Remaining Energy (%)

It is clear, from Figure 8, that "resource control" algorithms (HTAP and TARA) can uniformly utilize network resources in comparison with "traffic control" algorithms like SenTCP. When SenTCP is employed the network

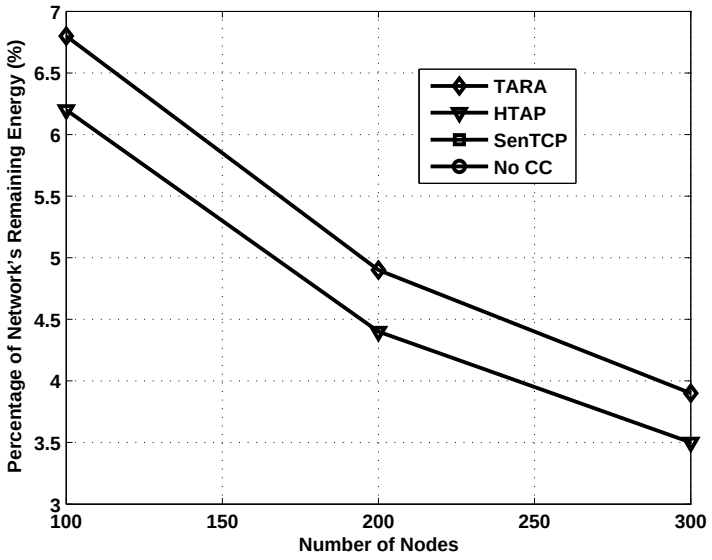

Figure 9: Network's Remaining Energy (%)(zoom on the lower part of Figure 8

stalls when the remaining energy in the network is near 20% in comparison with "resource control" algorithms where the network stalls at less than 7%.

If we zoom at the two "resource control" algorithms we recognize that their performance increases as the number of nodes is increasing. This fact is expected since more nodes provide more resources. Moreover, we notice that the performance of HTAP is better than TARA. This is an indication that HTAP can utilize the network resources more efficiently and can provide increased lifetime.

6.2. Evaluation of Congestion Threshold Adaptive Method

As we stated in section 4.3.1 the adaptive method that is employed in HTAP algorithm is important for the improved performance of the algorithm. In order to validate this statement we have designed and implement specific scenarios. Specifically we kept all the simulation parameters the same as in the previous section and we varied the congestion threshold. We have set four values for this congestion threshold. The first value is static but low (30%), the second is static but high (85%), while the third is static again and set to 50%. Finally the last one is the adaptive method we have used.

The first parameter we have examined is the percentage of successfully

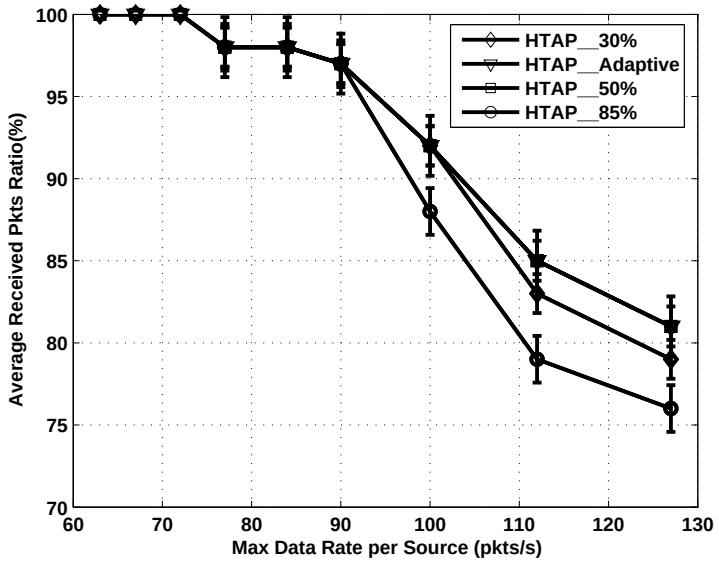

Figure 10: Percentage of Successfully Received Packets)

received packets. As we notice in Fig. (10) when the threshold is set to a low percentage of buffer occupancy(30% and 50%) the network presents almost the same performance as when adaptive method is being used. This means that when the nodes trigger the alternative path mechanism very early, the network avoids congestion and the only reason that packets are lost is because the capacity of the network is exceeded, as it happens when the source data rate is high. A small variation exists when the threshold is set to a very low value (30%), when the max data rate per source is over 100 packets per second. This can be explained by the fact that when the data rate is very high, a big number of nodes in the network have their buffer 30% filled and some packets are lost in the procedure of finding alternative paths.

On the other hand when the congestion threshold is set to a static high value (85%) the network starts dropping packets when the data rate exceeds 90 packets/s per source. This is an indication that when the alternative path creation is triggered, congested nodes cannot "absorb" the packets which are "in transit". Therefore, until the backpressure message reaches the sending nodes their buffer is already full and packets are getting lost.

To reinforce the findings of the previous figure, we study the Total Energy that is consumed during a congestion epoch. Figure 11 indicates that when

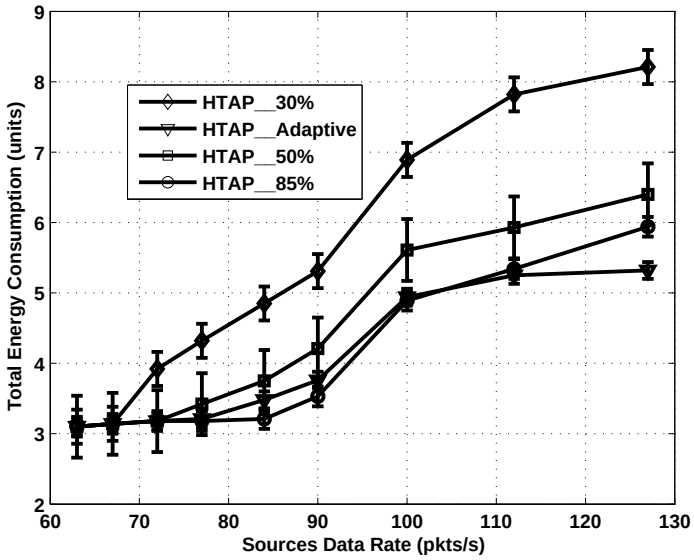

Figure 11: Total Energy Consumption

the threshold is set to 30% and 50% the energy that is consumed during a congestion epoch is high. We also notice that the energy consumption starts even when the load in the network is not very high (67 packets/sec per source for $HTAP30\%$ and 72 packets/sec per source for $HTAP50\%$). Since according Fig. 10 packet drops are limited at these values, the energy that is consumed is due to the alternative path creation. Thus, these values affect negatively the performance of the algorithm. When the value is set to 85% we notice that the energy consumption is similar to the "adaptive" case and it just increases when the load in the network is very high (128 packets/sec per source). Similar findings exist for the average hop-by-hop delay (Fig. 12.

6.3. Summary

As we conclude the performance evaluation section, we can state that the HTAP algorithm provides an increased performance at the range of 2-3% in comparison with TARA. Also it outperforms in certain cases the performance of SenTCP, which is "traffic control" algorithm. The topology control algorithm based on Local Minimum Spanning Tree (LMST) that was added in this version of HTAP has proven crucial for the successful operation of it. Specifically, this enhancement improved the average hop-by-hop delay,

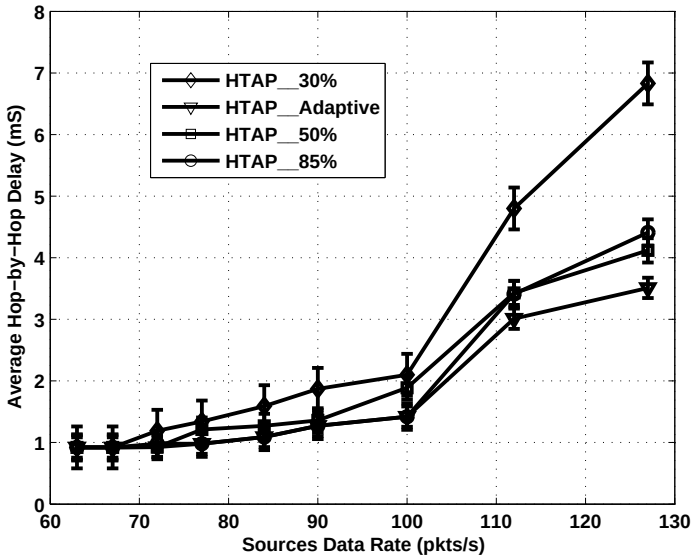

Figure 12: Average hop-by-hop delay

while the avoidance of "deadlocks" improved packet drops and retransmissions. We can also state that, while HTAP remains simple in its operation its performance has been significantly better than TARA which is a comparable scheme.

7. Performance Evaluation of HTAP through Different node Placement

As stated above, HTAP's successful operation relies on the number of available paths than are created between the source(s) and the sink. Different node placements are possible to provide a variable number of paths, which can improve or reduce the performance of HTAP [26]. In this section we study how the performance of HTAP algorithm is affected by different node placements.

Placement of nodes in a network can be divided into three major categories concerning the way nodes are placed in the area. In this work we choose to place nodes in four different placements: a deterministic placement (Grid), a semi-deterministic (Biased Random), and two non-deterministic (Simple Diffusion and Random).

7.1. Deterministic Node Placement

In deterministic node placement methods, nodes are placed on exact predefined points on the grid or in specific parts of the grid. Usually, deterministic or controlled node placement is specified by the type of nodes, the environment in which the nodes will deploy, and the application. Therefore, in applications like Indoor Surveillance Systems or Building Monitoring, nodes must be placed manually [27] (either by hand or by robots).

Grid Placement: In this placement nodes are placed strictly on the lines of a Grid (Fig. 13).

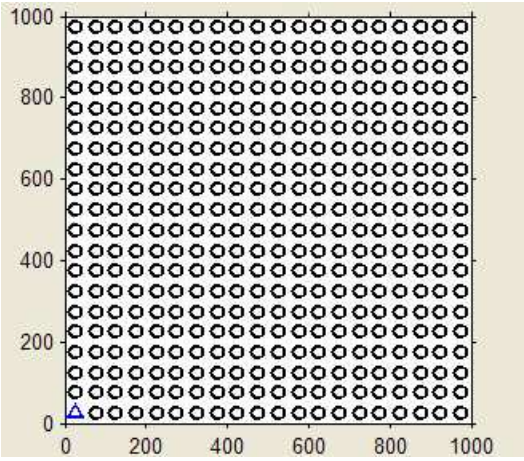

Figure 13: Grid Placement

7.2. Semi-Deterministic Node Placement

Semi-deterministic placement is the placement, where, although individual nodes are placed in a non-deterministic way on the grid (e.g random) the areas where nodes are going to be spread are pre-determined. This means that in a microscopic way the placement of nodes is non-deterministic, while in a macroscopic way the placement is deterministic. In this paper we employ biased-random placement, where nodes are placed in two specified areas (near source and near sink). Note that the actual node placement is performed in a random way in these areas (Fig. 14).

7.3. Non-Deterministic Node Placement

Deterministic Placement is not so realistic when many sensor nodes are placed in a large area. In such situations, stochastic placement is needed.

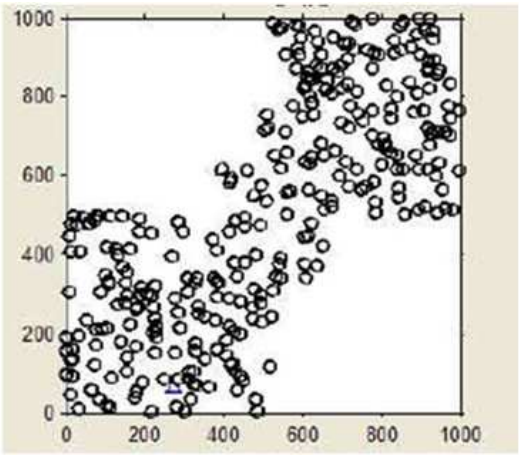

Figure 14: Biased Random Placement

In this paper we employ two stochastic placements: Simple Diffusion and Random Placement.

Simple Diffusion: This node placement emulates the distribution of nodes when they are scattered from air e.g from airplane (Fig. 15). Simple diffusion is analytically explained in [28].

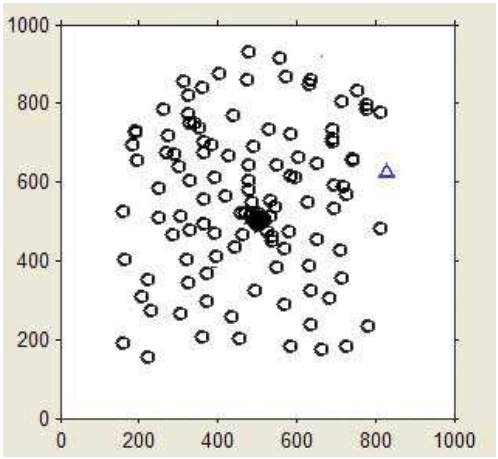

Figure 15: Simple Diffusion Placement

Random Placement: This is a commonly used topology and sensor nodes are placed so that their density is uniform (Fig. 16). This is the topology that has been used for the simulations in Section 6.1, and in many

WSN-related research.

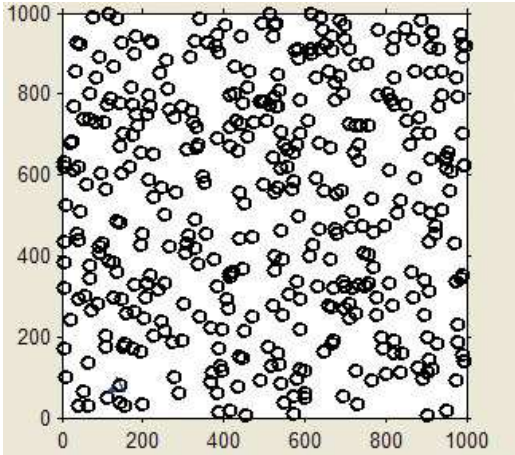

Figure 16: Random Placement

7.4. Simulation Parameters

For the evaluation of HTAP under different placements we employ the same simulation parameters as in the previous section. Again, 100 nodes were uniformly (randomly) deployed in 500m x 500m grid. The sink was set in the upper right corner of the grid while sources were set in the bottom left corner of the grid in order to create convergence situation and congestion conditions. Each simulation run has been performed 20 times and average results have been extracted.

7.5. Results

The first metric we study is the percentage of successfully received packets.

As we notice in Figure 2 where the placement is random, at the maximum data rate (128 packets/s) the percentage of received packets was near 80%. With exactly the same simulation parameters and different placements we notice that this percentage changes. When nodes are placed under a "Biased-Random" placement the percentage is much higher. Even when "Simple Diffusion" applies, the percentage is higher than Random placement, while "Grid" placement presents the worst results. The reason obviously lies on the number of paths which are created when diverse placements are employed.

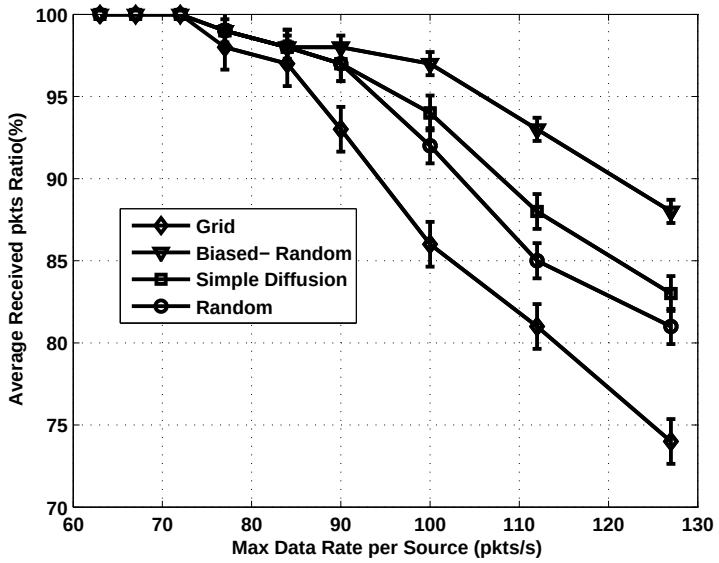

Figure 17: Percentage of Successfully Received Packets

When the "Biased-Random" placement is employed, nodes are placed near the sources and the sink, the places with the higher probability of congestion occurrence. Subsequently, more paths are available in order to extract and transmit data to sink. On the other hand, when the "Grid" placement is used, the number of paths is limited. This means that network capacity is less and the results are the worst.

The next metric we consider is the actual number of packets that is received by sink (throughput). Also in this figure the "Biased Random" placement favors the network. This means that the number of packets that are received by the sink is increased. "Simple Diffusion" also presents better results than "Random" placement, and "Grid" placement presents the worst results.

Next we study the Hop-by-Hop Delay and the Total Energy Consumption.

Also in this case the results are in correlation with the results in Figures 17 and 19. Hop-by-Hop delay and Total Energy Consumption are strictly related with packet drops and link layer retransmission and the results again vary under different placements.

Finally we study the Percentage of Network Remaining Energy at the point where the network is unable to transfer a single packet from source to

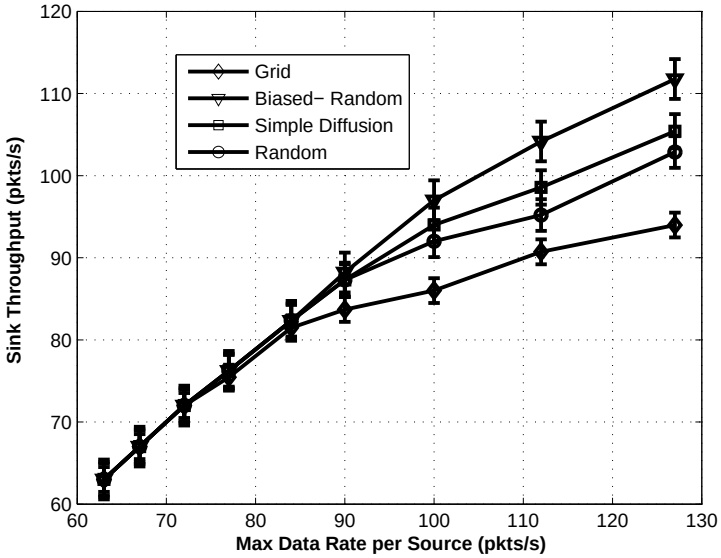

Figure 18: Throughput

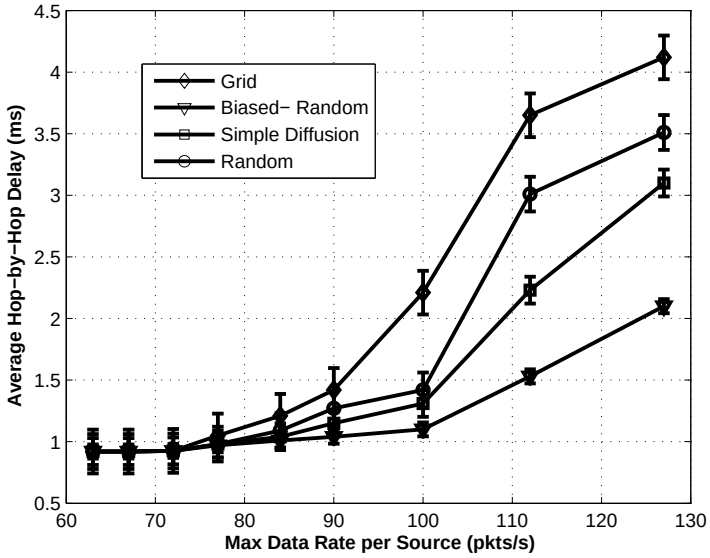

Figure 19: Average Hop-by-Hop Delay

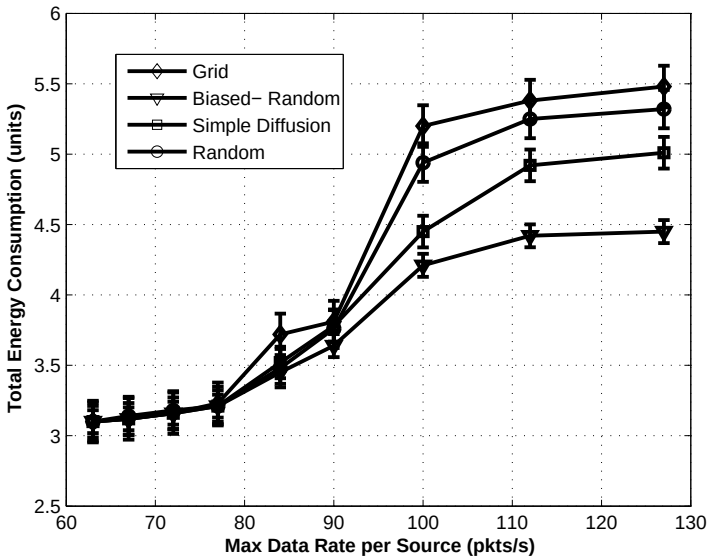

Figure 20: Total Energy Consumption

sink.

Analyzing these results we notice that when the nodes follow the "Biased-Random" approach the percentage of the network's remaining energy is the best, indicating that under this placement, the utilization of network resources is more uniform than the other placements. We also notice that the number of nodes in the network is also a very important factor for the uniform utilization of network's resources.

8. Conclusions

In this paper we presented the Hierarchical Tree Alternative Path (HTAP) algorithm. HTAP is an algorithm that manages to control congestion in WSNs using a "resource control" method. When congestion appears in the network the HTAP scheme transmits the excess packets to the sink through alternative routes, employing nodes which are not in the initial path from the source(s) to the sink. HTAP's successful and efficient functionality lies on a topology control scheme that creates the initial connectivity in the network and in a hierarchical tree scheme which discovers all possible upstream routes from the sources to the sinks when an event occurs. HTAP employs

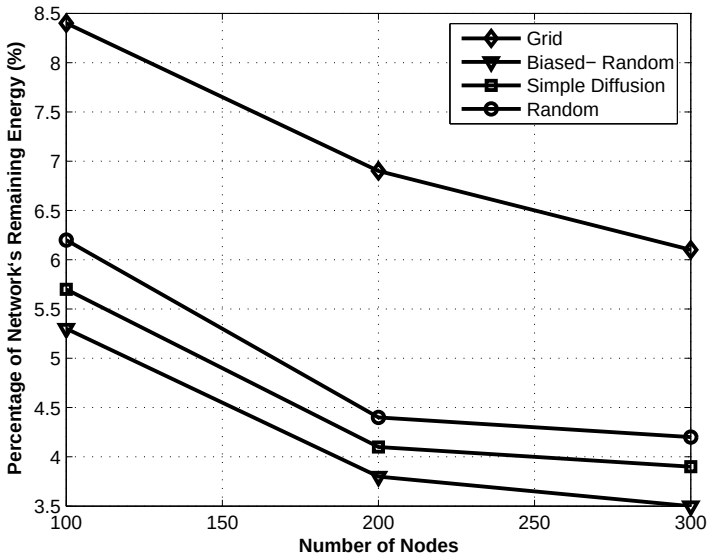

Figure 21: Network's Remaining Energy (%)

an adaptive "congestion threshold" that renders it capable to skip transient congestion situations, while it is efficient enough to create alternative paths to the sink in order to control persistent congestion situations. The major advantage of the HTAP algorithm is its simplicity, which adds minimal overhead to the already heavy loaded networks that is intended to operate onto. HTAP has been evaluated and its performance was compared with another "resource control" algorithm (TARA), a "traffic control" algorithm (SenTCP), and finally the "no congestion control" case. Simulation results show that HTAP is an efficient and simple solution for facing overload situations in densely deployed WSNs. HTAP has also been evaluated under different node placements. Results show that HTAP is affected by different placements and its performance improves when nodes are densely deployed near possible hot spots like sources and sinks.

Acknowledgment

This work has been partly conducted under the European Union Project GINSENG funded under the FP7 Program (FP7/2007-2013) grant agreement no 224282.

References

[1] S. Toumpis, L. Tassiulas, Optimal Deployment of Large Wireless Sensor Networks, IEEE Transactions on Information Theory 52 (7) (2006) 2935–2953.

[2] C. Wang, M. Daneshmand, B. Li, K. Sohraby, A Survey of Transport Protocols for Wireless Sensor Networks, IEEE Network, May/June 20 (2006) 34–40.

[3] C. Wang, K. Sohraby, B. Li, W. Tang, Issues of Transport Control Protocols for Wireless Sensor Networks, in: Proceedings of International Conference on Communications, Circuits and Systems (ICCCAS), 2005.

[4] C. Sergiou, V. Vassiliou, A. Pitsillides, Reliable Data Transmission in Event-Based Sensor Networks During Overload Situation, in: WICON '07: Proceedings of the 3rd International Conference on Wireless Internet, Austin, Texas, ISBN 978-963-9799-12-7, 1–8, 2007.

[5] C. Sergiou, V. Vassiliou, Energy Utilization of HTAP Under Specific Node Placements in Wireless Sensor Networks, in: European Wireless Conference (EW),, 482 –487, doi:10.1109/EW.2010.5483490, 2010.

[6] N. Li, J. Hou, L. Sha, Design and Analysis of an MST-based Topology Control Algorithm, in: INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies, vol. 3, ISSN 0743-166X, 1702 – 1712 vol.3, doi: 10.1109/INFCOM.2003.1209193, 2003.

[7] R. C. Prim, Shortest Connection Networks and Some Generalizations, Bell System Technology Journal 36 (1957) 1389–1401.

[8] J. Kang, Y. Zhang, B. Nath, TARA: Topology-Aware Resource Adaptation to Alleviate Congestion in Sensor Networks, IEEE Transactions on Parallel and Distributed Systems 18 (7) (2007) 919–931, ISSN 1045-9219, doi:http://doi.ieeecomputersociety.org/10.1109/TPDS.2007.1030.

[9] C. Wang, K. Sohraby, B. Li, SenTCP: A Hop-by-Hop Congestion Control Protocol for Wireless Sensor Networks, IEEE INFOCOM (Poster Paper) .

[10] B. Hull, K. Jamieson, H. Balakrishnan, Mitigating Congestion in Wireless Sensor Networks, in: SenSys '04: Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems, ACM, New York, NY, USA, ISBN 1-58113-879-2, 134–147, doi: http://doi.acm.org/10.1145/1031495.1031512, 2004.

[11] A. Woo, D. E. Culler, A Transmission Control Scheme for Media Access in Sensor Networks, in: Proceedings of the 7th annual International Conference on Mobile Computing and Networking (MobiCom'01), ACM Press, New York, NY, USA, ISBN 1-58113-422-3, 221–235, doi: http://doi.acm.org/10.1145/381677.381699, 2001.

[12] C. Lu, B. M. Blum, T. F. Abdelzaher, J. A. Stankovic, T. He, RAP: A Real-Time Communication Architecture for Large-Scale Wireless Sensor Networks, Tech. Rep., Charlottesville, VA, USA, URL `http://portal.acm.org/citation.cfm?id=900537`, 2002.

[13] Y. Sankarasubramaniam, O. Akan, I. Akyildiz, ESRT: Event-to-Sink Reliable Transport in Wireless Sensor Networks, in: MobiHoc '03: Proceedings of the 4th ACM International Symposium on Mobile Ad Hoc Networking & Computing, ACM, New York, NY, USA, ISBN 1581136846, 177–188, doi:10.1145/778415.778437, URL `http://dx.doi.org/10.1145/778415.778437`, 2003.

[14] C. T. Ee, R. Bajcsy, Congestion Control and Fairness for Many-to-One Routing in Sensor Networks, in: SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems, ACM, New York, NY, USA, ISBN 1-58113-879-2, 148–161, doi: http://doi.acm.org/10.1145/1031495.1031513, 2004.

[15] C.-Y. Wan, S. B. Eisenman, A. T. Campbell, CODA: Congestion Detection and Avoidance in Sensor Networks, in: SenSys '03: Proceedings of the 1st international Conference on Embedded Networked Sensor Systems, ACM Press, New York, NY, USA, ISBN 1581137079, 266–279, URL `http://portal.acm.org/citation.cfm?id=958523`, 2003.

[16] K. Sundaresan, V. Anantharaman, H.-Y. Hsieh, R. Sivakumar, ATP: A Reliable Transport Protocol for Ad Hoc Networks, IEEE Transactions on Mobile Computing 4 (2005) 588–603, ISSN 1536-1233, doi: http://doi.ieeecomputersociety.org/10.1109/TMC.2005.81.

[17] F. Stann, J. Heidemann, RMST: Reliable Data Transport in Sensor Networks, in: Proceedings of the First International Workshop on Sensor Net Protocols and Applications, IEEE, Anchorage, Alaska, USA, 102–112, URL http://www.isi.edu/ johnh/PAPERS/Stann03a.html, 2003.

[18] C.-Y. Wan, A. T. Campbell, L. Krishnamurthy, P.S.F.Q: A Reliable Transport Protocol for Wireless Sensor Networks, in: WSNA '02: Proceedings of the 1st ACM international workshop on Wireless Sensor Networks and Applications, ACM Press, New York, NY, USA, ISBN 1581135890, 1–11, doi:10.1145/570738.570740, URL http://dx.doi.org/10.1145/570738.570740, 2002.

[19] C.-Y. Wan, S. B. Eisenman, A. T. Campbell, J. Crowcroft, Siphon: Overload Traffic Management using Multi-Radio Virtual Sinks in Sensor Networks, in: SenSys '05: Proceedings of the 3rd international conference on Embedded networked sensor systems, ACM, New York, NY, USA, ISBN 1-59593-054-X, 116–129, doi: http://doi.acm.org/10.1145/1098918.1098931, 2005.

[20] P. Antoniou, A. Pitsillides, T. Blackwell, A. Engelbrecht, Employing the Flocking Behavior of Birds for Controlling Congestion in Autonomous Decentralized Networks, in: A. Tyrrell (Ed.), 2009 IEEE Congress on Evolutionary Computation, IEEE Computational Intelligence Society, IEEE Press, Trondheim, Norway, –, 2009.

[21] L. Popa, C. Raiciu, I. Stoica, D. S. Rosenblum, Reducing Congestion Effects in Wireless Networks by Multipath Routing, in: ICNP, IEEE Computer Society, ISBN 1-4244-0593-9, 96–105, URL http://doi.ieeecomputersociety.org/10.1109/ICNP.2006.320202, 2006.

[22] T. He, F. Ren, C. Lin, S. Das, Alleviating Congestion Using Traffic-Aware Dynamic Routing in Wireless Sensor Networks, in: SECON, IEEE, 233–241, 2008.

[23] J. Hightower, G. Borriello, Location Systems for Ubiquitous Computing, Computer 34 (2001) 57–66, ISSN 0018-9162, doi:http://dx.doi.org/10.1109/2.940014, URL http://dx.doi.org/10.1109/2.940014.

[24] W.-w. Fang, J.-m. Chen, L. Shu, T.-s. Chu, D.-p. Qian, Congestion Avoidance, Detection and Alleviation in Wireless Sensor Networks, Journal of Zhejiang University - Science C 11 (2010) 63–73, ISSN 1869-1951, URL http://dx.doi.org/10.1631/jzus.C0910204, 10.1631/jzus.C0910204.

[25] Prowler: Probabilistic Wireless Network Simulator, URL http://www.isis.vanderbilt.edu/Projects/nest/prowler, 2003.

[26] V. Vassiliou, C. Sergiou, Performance Study of Node Placement for Congestion Control in Wireless Sensor Networks, in: New Technologies, Mobility and Security (NTMS), 2009 3rd International Conference on, 1 –8, doi:10.1109/NTMS.2009.5384782, 2009.

[27] M. Younis, K. Akkaya, Strategies and Techniques for Node Placement in Wireless Sensor Networks: A Survey, Ad Hoc Networks 6 (4) (2008) 621–655, ISSN 1570-8705.

[28] M. Ishizuka, M. Aida, Performance Study of Node Placement in Sensor Networks, ICDCSW '04: Proceedings of the 24th International Conference on Distributed Computing Systems Workshops (2004) 598–603.