Master's Thesis

CONGESTION CONTROL IN WIRELESS SENSOR NETWORKS

Charalambos Sergiou

UNIVERSITY OF CYPRUS

DEPARTMENT OF COMPUTER SCIENCE

June 2007

UNIVERSITY OF CYPRUS

DEPARTMENT OF COMPUTER SCIENCE

CONGESTION CONTROL IN WIRELESS SENSOR NETWORKS

Charalambos Sergiou

Supervising Professor

Dr. Andreas Pitsillides

This thesis is submitted to the Graduate Faculty of University of Cyprus in partial fulfillment of the requirements for the Degree of Master of Science at Computer Science Department

June 2007

Acknowledgments

I would deeply like to thank my advisor, Dr. Andreas Pitsillides, for his invaluable guidance and great support through my study and research.

I am also thankful to Dr. Vasos Vassiliou and Mr. Yiannos Milonas for their valuable time, comments and technical support. Also I would like to thank my friend and colleague Timotheou Timotheou for his valuable help.

At last I am grateful to my parents Theodoros and Georgia and my wife Christiana for their moral support and help, during the research and writing of this MSc Thesis.

Summary

In wireless sensor networks (WSNs), nodes have very limited power due to hardware constraints. Packet losses and retransmissions resulting from congestion, cost precious energy and shorten the lifetime of sensor nodes. This problem motivates the need for congestion control mechanisms in WSNs.

In this research we concentrate on event-based networks. In event-based networks, data packets are produced only upon the observation of specific events that satisfy pre-specified conditions. When a specific condition is satisfied, there may be a huge transmission of data packets, which, if not controlled, can easily lead to congestion. Congestion, especially in these networks, is highly undesirable, not just because of the obvious power consumption but also due to the fact that is able to destroy the networks' mission. The data packets which are produced when an event- based network satisfies the pre- specified condition are critical for the success of the applications.

The target of this Thesis is the development of a specific algorithm which is able to transmit these critical packets to the sink, safely, without throttling the data rate.

Based on the above, a congestion control scheme is proposed. The target of this scheme is, during a congestion situation, to maintain the reliability of information flow to the sink. In order to maintain the data rate, we try to take advantage of the plethora of sensor nodes which are in the network and are not used during a session (i.e increase resource provisioning).

Specifically the algorithm consists of three main parts: (i) Hierarchical Flooding which is used initially for the network discovery (ii) Hierarchical Tree Alternative Path (HTAP) algorithm in order to deal with the expected congestion situation and to safely forward the data packets to sinks (iii) Handling of powerless (dead) nodes. The HTAP (Hierarchical Tree Alternative Path) algorithm tries to solve a congestion situation locally "by- passing" the congested node through the creation of alternative paths form the source to the sink.

This scheme has been implemented in MATLAB. Simulations have been conducted to evaluate this algorithm. Results show that HTAP algorithm can cope with congestion and maintain the reliability of data packets transmission to the sink. In addition it achieves good performance in terms of energy dissipation, latency and transmission efficiency.

Table of Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation

Wireless sensor network (WSN) is a wireless network consisting of spatially distributed autonomous devices using sensors to cooperatively monitor physical or environmental conditions, such as temperature, sound, vibration, pressure, motion or pollutants, at different locations[1][2].

Research in WSN was originally motivated for military applications, especially for monitoring the battlefield. In recent years, due to the advances in low power units and improvements in radio technologies, wireless sensor networks technologies have emerged. Applications using sensors are increasing fast. A wide range of them is now deployed in civilian areas, like habitat observation [3], [4], health monitoring [5], object tracking [6], [7].

In recent years intense study has been carried out concerning many aspects of WSN especially physical layer [8], [9], MAC layer [10], [11], [12], and network layer [13], [14],

[15], [16]. Having addressed many of these fundamental problems, nowadays the problem **of congestion control and avoidance** begins to attract a lot of attention [Chapter 2].

Congestion in WSN is expected to occur in many cases due to the high density of sensor nodes in a network and the variable data rates that they exhibit. Congestion usually appears in two major "patterns".

The first "pattern" is the congestion which appears when a sensor node receives more data than it can forward. When this happens, the sensor node must buffer the excess data. A node when it receives more data than it can store the excess data must be dropped. Bearing in mind that the buffer space of these nodes is limited and the production of data, especially when an event is taking place, is high, congestion can happen easily.

In Fig.1, three nodes W, Z, Y are transmitting simultaneously packets to node X. Node's X buffer will soon fill up and excess data will be dropped.

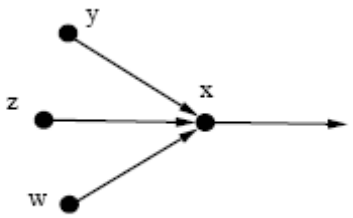

Fig. 1. 1 Simultaneous transmission causes buffer overflow

In addition the same event can occur if the source node transmits data packets in higher rate than the receiving node can transmit. An example is presented in Fig.1.1. In this figure if node Z transmits data packets in a higher rate than the ones node X can forward, the buffer of node X's will soon come in a congestion state. Exactly this type of congestion we try to counter in this Thesis.

The second "pattern" of congestion occurs when a particular area is densely deployed. In such a case, when a number of nodes attempt to transmit simultaneously, triggered from a specific event, interference is caused at the listening node. One solution for this problem is exponential random back-off and virtual carrier sensing **[17].**

In this thesis, we will focus on buffer- based congestion which still remains an open problem in WSN.

1.2 WSNs unique features

WSNs belong to the family of ad-hoc networks. The main feature of ad-hoc networks is the lack of infrastructure and the ability of self configuration. Sensor networks obey to these characteristics, but they also have some distinguishing features.

The most important of them are the following:

a. Application Specific: WSN are targeted to a specific application. Depending on the application their characteristics change. For example, their deployment could be different, from very dense to very sparse, their communication protocol could be different, even their size. It is impossible that there will be a "one-size-fits-all" solution for all the applications.

b. Data rates: Due to their interaction with the environment, WSN exhibit very different traffic characteristics compared to other networks. So, for significant amount of time their data rate could be very low and in case that a specific event takes place, very high data rate may appear.

c. Power: Due to the small size and the low cost, the energy in these nodes is a scarce resource. The entire architecture of these networks is based on the power limitations, so as to maximize their lifetime for applications, which may be deployed in hazardous environments.

d. Self- Configuration: Most of the sensor networks applications demand the self configuration of the sensor nodes to networks. Self configuration is also a requirement which arises due to the limitations of power. Limitation in power often leads to nodes failure and the network needs to be configured again.

e. Cooperation between nodes: WSNs operate as typical ad- hoc networks as far as the transmission of data. So, each node may sense and transmit data by itself or it can act as a router, transmitting other nodes' packets (hop by hop).

f. QoS: Quality of service in WSN is application depended. This means that the QoS is anticipated by the success of the mission and not out of the number of packets drops etc.

1.3 Sensor Node Details

The main components of sensors consist of a sensing unit, a processing unit, a transceiver, and a power unit, as shown in Fig. 1, adapted from [18]. Each component is described below.

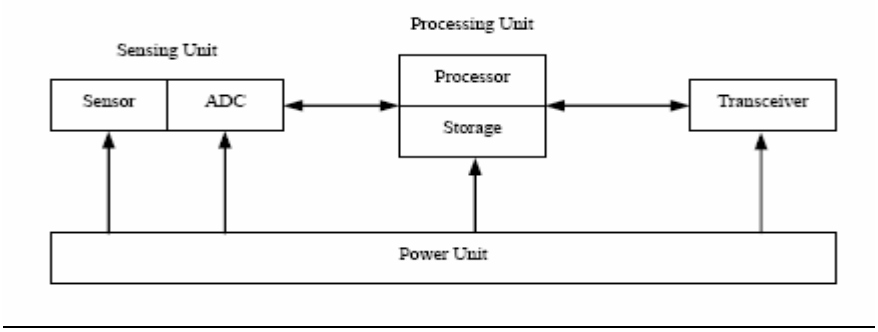

Fig. 1. 2 Sensor Node's Hardware Units

a. Sensing Unit

The main functionality of the sensing unit is to sense or measure physical data from the target area. The analog voltage, which is generated by the sensor corresponding to an "event'', is then digitized by an analog-to digital converter (ADC) and then delivered to the processing unit for more analysis.

b. Processing Unit

The processing unit takes a major role in managing collaboration with other sensors to achieve the predefined tasks. There are currently several families of this unit including microcontrollers, microprocessors, and field-programmable gate arrays (FPGAs).

The Non-volatile memory and interfaces such as ADCs can be integrated onto a single integrated circuit [19] [20]. The processing unit needs storage for tasking and to minimize the size of transmitted messages by local processing and data aggregation [21]. Flash memory is widely used due to its low cost and storage capacity.

c. Transceiver

There are three deploying communication schemes in sensors, including optical communication (laser), infrared, and radiofrequency (RF). RF is the easiest to use, but requires antenna.

d. Power Unit

Power consumption is a major weakness (problem) of sensor networks. Batteries used in sensors can be categorized into two groups; rechargeable and non-rechargeable. Often, in harsh environments, it is impossible to recharge or change a battery.

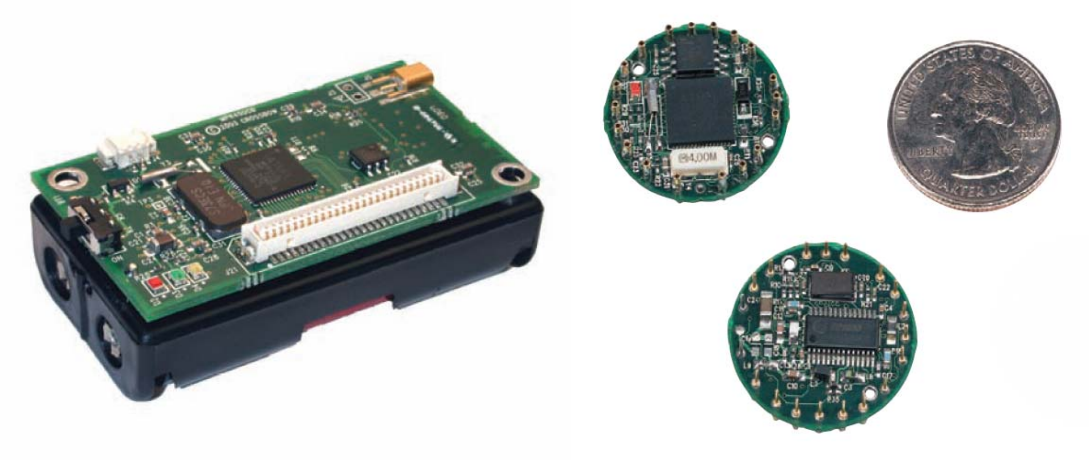

Fig. 1.3 Sensor nodes

1.4 Protocol Stack

The sensor networks follow the general rules of other networks concerning their protocol stack with the following layers: Application, Transport, Network, Data Link and Physical. Though, due to specific characteristics of these networks especially power constraints or specific application tasks, some of these layers could merge in one. Moreover, some of the functionalities of each layer, as it is explained below, can be very different than classical TCP/IP protocol stack. Following, we present some of the main functionalities of each layer in a typical sensor network.

a. Physical Layer

Physical Layer deals with frequency selection, carrier frequency generation, signal detection, modulation and encryption. Its main priority is energy minimization and secondary tasks, like other wireless networks.

b. Data Link Layer

Data Link Layer deals with the multiplexing of data streams, data frame detection, medium access and error control. Due to power constraints, WSNs implement specific MAC protocols that take in mind the power conservation and data- centric routing. The design of MAC protocols embodies the achievement of at least two targets. The first target is the creation of a specific network infrastructure that is able to self- organize and, also, to be able to establish connections among thousand of nodes. The second target is the fair recourse allocation between all nodes.

Unfortunately, existing MAC protocols fail to meet these targets due to the fact that power conservation is a secondary concern in their development. These MAC protocols are not designed to provide self- organization of the network during the initial deployment or in case of nodes failure, due to power limitations. Two efforts for MAC protocols that meet these requirements are the Sensor- MAC (SMAC) [22] and CSMA MAC [22].

c. Network Layer

Network layer design is based on the power considerations of the network. WSNs embody attributed based addressing and location awareness. Network layer, like in traditional networks, is responsible for deciding which node to talk to.

The simplest and mostly spread network protocol is flooding. With flooding, each node broadcast its data to all the other nodes in network until these data reach the destination.

The main advantage of flooding is simplicity. It requires no costly topology maintenance or complex route discovery. The shortcomings, however, are substantial. The first problem is that of implosion. Implosion occurs when two nodes (A and B) share multiple (n) neighbours. Node A will broadcast data to all of these neighbours. Node B will then receive a copy of the data from each of them. The second issue is that of overlapping, when two nodes share the same sensing region. If a stimulus occurs within this overlap, both nodes will report it. The last and most crucial problem is resource blindness. Flooding does not take into account available energy resources.

A step further from flooding is gossiping. In gossiping, when a node receives data, it randomly chooses a neighbour and sends the data to it. Gossiping avoids the problem of implosion, but does not address the other two concerns (overlap, recourse blindness) and contributes to the latency of the network.

d. Transport Layer

Transport layer is mainly used when there is need for communication with a network outside the WSN. This is usually needed from the sink to the user. This can become a problem due to the fact WSN addressing is not based on global addressing and attribute-based naming is used to indicate the destinations of DATA packets.

e. Application Layer

One example is Sensor Management Protocol, SMP [23] at the application layer, which is used to make the hardware and software of lower layers transparent to the Sensor Network Management Applications. The system administrators and programmers interact with the Sensor Network using SMP. Again, the lack of global identification and infrastructure-less nature of sensor networks must be taken into consideration. SMP provides the rules to enable interaction between applications and sensor networks for the followings:

- Data aggregation, attribute-based naming, and clustering
- Exchanging of data related to the location finding algorithms
- Time synchronization
- Moving sensor nodes
- Turning nodes on or off
- Querying WSN configuration status, reconfiguring the WSN
- Authentication, key distribution, and security

1.5 Thesis objectives

In this Thesis, congestion in a special category of WSNs, the event- based networks is investigated. The distinguishing feature of event- based networks is the production and transmission of a big number of data packets to the sinks when the network comes in a crisis

state. Crisis state is the network's state, when a specific event satisfies a pre- specified condition.

A typical example is the case of fire in a forest. Sensors are programmed to send data packets when the temperature in the forest exceeds a pre-specified value. In the case of a fire, sensor nodes which are neat to the event, sense the fire and begin transmitting a large amount of data packets simultaneously. The target of this network is the continuous provision to sinks (i.e. local fire stations) with data packets, in order to keep them updated, for the fire's frontline. If the high load of data packets left unattended (no congestion control) in this case the network becomes congested and the packets will not be forwarded to sinks. The result will be the failure of the network's mission.

Taking these input as granted, it is obvious that there is a need for the development of an algorithm which is able to forward almost all data packets to the sinks, safely and reliably. Due to the high importance of these packets and the strict time limitations, reduction of data rate is prohibited (i.e. in a forest which is getting burned, almost every data packet provides new information for the spreading of fire).

In order to achieve this target, we take advantage of a special characteristic of these networks, namely the plethora of unused nodes. Using these nodes we increase the recourse provisioning in the paths, using the power and buffers of nodes which in this specific moment are in dormant state. The use of these nodes creates alternative paths to the sinks, alleviating the highly congested areas.

Chapter 2

Related Work

2.1 Proposals and Implementations aiming to Solve Congestion Problem

In this chapter we present a number of publications which are focusing on Congestion Control in WSNs either directly or indirectly. Publication which are focusing directly in Congestion, are presented in more detail.

Publication Directly focusing on Congestion Control in WSNs

a. CODA (COngestion Detection and Avoidance) [24]

Comprises of three mechanisms:

- CODA congestion indicators
- Open-loop, hop-by-hop backpressure
- Closed-loop, multi-source regulation

CODA congestion indicators: Uses a combination of the present and past channel loading conditions and the current buffer occupancy, in order to accurately detect congestion at each receiver. CODA uses a technique called channel sampling (to obtain current channel

utilization) that activates local channel monitoring at the appropriate time to minimize cost while forming an accurate estimate.

The channel sampling algorithm is triggered when the node's packet queue becomes non empty. This is the time when nodes want to start packet transmission and when the question of whether the channel is congested or not, becomes important. The time after starting the sampling period is divided in sampling epochs, in which one of the channels is periodically sampled, say N times. When, in epoch n, a number of M out of N samples indicate a busy channel, this epoch's utilization = M/N. Then, the estimate of k consecutive epochs must be combined. Once congestion is detected, nodes notify their upstream neighbors via a backpressure mechanism.

Open-loop, hop-by-hop backpressure: In CODA, a node broadcasts backpressure messages for the period it detects congestion. Backpressure signals are propagated upstream to the source. In the case of impulse data events in dense networks, it is very likely that backpressure will propagate directly to the sources. Nodes that receive backpressure signals can reduce their sending rates or drop packets based on the local congestion policy (e.g., packet drop, AIMD, etc.). When an upstream node (toward the source) receives a backpressure message it decides whether or not to further propagate the backpressure upstream, based on its own local network conditions.

It can also serve as an on demand "clear to send" signal, so that all other neighbors except one sender (which could be chosen randomly, or a node can assign more chances to more desirable senders) can be silenced at least for a single packet transmission time.

Closed-loop, multi-source regulation: A sink node might receive data from multiple sources. This may create a persistent hot spot close to the sink or in other regions of the network. In this situation, hop – by- hop backpressure mechanism is not efficient.

In CODA, closed loop regulation operates over a slower time scale and is capable of asserting congestion control over multiple sources from a single sink in the event of persistent congestion. When the source event rate is less than some fraction of the maximum theoretical throughput of the channel, the source regulates itself. When this value (threshold) is exceeded, a source is more likely to contribute to congestion and, therefore, closed-loop congestion control is triggered.

The source enters sink regulation only if this threshold is exceeded. At this point a source requires constant, slow time-scale feedback (e.g., ACK) from the sink to maintain its rate. The reception of ACKs at sources serves as a self-clocking mechanism allowing sources to maintain their current event rates. In contrast, failure to receive ACKs forces a source to reduce its own rate. The sources stop to request further regulation as soon as their generation rate drops below the threshold value.

b. Congestion Control and Fairness for Many-to-One Routing [25]

This proposal for congestion control assumes a tree routing structure from all data sources to a sink. It is a distributed and scalable algorithm (each sensor mote requires state proportional to the number of its neighbors), implemented in sensor nodes that eliminates congestion within a sensor network, and ensures the fair delivery of packets to a central node (sink).

This algorithm exists in the transport layer of the traditional network stack model, and is designed to work with any MAC protocol in the data-link layer with minor modifications.

Since in general we have many sensors transmitting data to the sink, the scenario is about many-to-one multi-hop routing (can easily be extended to unicast or many-to-many routing). Each sensor receives and forwards packets from its upstream neighbors; each upstream neighbor is the root of an upstream sub-tree. The sensor learns the number of data sources in each of those upstream sub-trees, measures its own downstream forwarding rate, computes per-source fair rate, which is propagated upstream such that the data sources do not send packets beyond the rate.

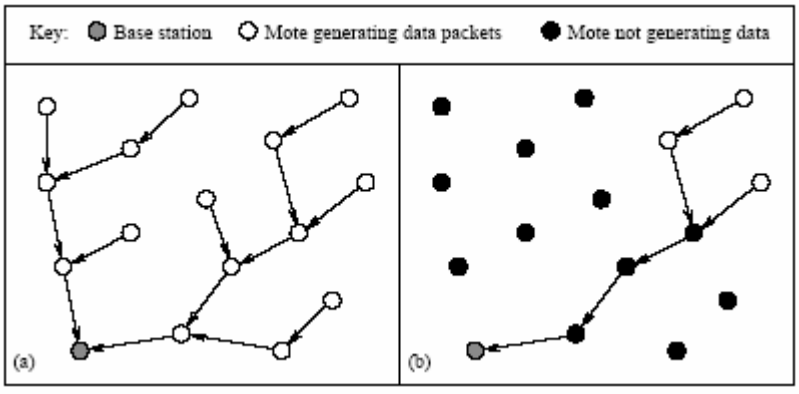

Fig. 2. 1 CODA routing examples

There are two alternative scenarios in which this algorithm can be applied to:

- All nodes are generating data and routing them to the sink
- Most nodes in the network are silent, only the nodes that detect an event generate data.

Such routing structures often result in the sensors closer to the base station experiencing congestion, which inevitably cause packets originating from sensors further away from the base station to have a higher probability of being dropped.

The algorithm deals with two types of congestion:

- Reduce Type A: a lot of nodes transmit simultaneously producing interference (reduce throughput of each other)] via phase shifting (including jitters in DLL or Application Layer).
- Reduce Type B: [buffer within a particular node overflows] via buffer monitoring.

The basic concept for controlling congestion consists of the following steps that repeatedly run at each sensor node:

- Measure the average rate r at which packets can be sent to from this mote
- Divide the rate r among the number of children motes downstream n, to give the per-node data packet generation rate $r_{data} = r/n$, adjust the rate if queues are overflowing or about to overflow,
- Compare the rate r_{data} with the rate $r_{data,parent}$ sent from the parent, use and propagate the smaller rate downstream.

The implementation of this solution requires:

- Obtaining the number of downstream nodes n,
- Measurement of the average rate r,
- Obtaining the per-node rate disseminated by the parent, and
- Propagation of rate downstream.

c. Mitigating Congestion Control in WSN [26]

In this paper, a technique called **fusion** is suggested. Fusion combines three congestion control techniques that operate at different layers. These techniques are working together in order to provide the congestion control in WSN.

The three techniques are the following:

- Hop-by-hop flow control
- Source rate limiting scheme
- Prioritized MAC

Hop-by-hop flow control: Each sensor sets a congestion bit in the header of every transmitted packet. Using the broadcast characteristic of the wireless medium, every packet provides congestion feedback to all nodes in a radio neighborhood with every transmission. So, there is no need for explicit control messages, which waste a large portion of the, already limited, bandwidth.

Hop-by-hop flow control consists of two components: congestion detection and congestion mitigation. Congestion detection is a way to detect congestion, based on monitoring sensor's queue size. If the sensor's queue space falls below a specified limit, a congestion bit is set. Otherwise the congestion bit is removed. Congestion mitigation is a way to control the nodes' transmissions in order to prevent queues at their next- hop node from overflowing.

When a sensor overhears to a packet with the congestion bit set, it stops forwarding data. Otherwise, the congestion would grow bigger, and eventually the whole network will collapse.

Rate Limiting: It is a way of limiting the sending rate of a sensor. Each sensor listens to the traffic its parent forwards to estimate N, the total number of unique sources routing through the parent. A token bucket scheme is used to regulate each sensor's send rate. A sensor accumulates each token every time it hears its parent forward N packets, up to a maximum number of tokens. The sensor is allowed to send, only when its token count is above zero and each send costs one token.

Prioritized MAC: Mac layer provides assistance to sensors in order to react to congestion fast enough. A standard CSMA Mac layer is used, with the modification that implements a prioritization scheme. According to this scheme, congested nodes have higher priority compared to the other nodes. Specifically, if a sensor is congested, its back-off window is one-fourth of the size of a non-congested sensor's back off window, allowing queues to drain and increasing the likelihood congestion control in- formation will propagate throughout a sensor's neighborhood.

Methods Indirectly referring to Congestion Control in WSNs

d. Event-To-Sink Reliable Transport (ESRT) Protocol [27]

Wireless sensor networks (WSN) are event based systems that rely on the collective effort of several sensor nodes.

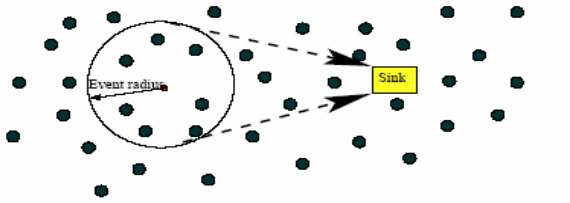

Fig. 2. 2 Typical sensor network topology with event and sink.

The event-to-sink reliable transport (ESRT) protocol is a transport solution. It works by carefully adjusting the sensor's reporting rate to achieve two goals:

- A sufficient number of packets is received
- Not many more packets than needed are received to avoid congestion and save energy.

The algorithms of ESRT mainly run on the sink, with minimal functionality required at resource constrained sensor nodes. ESRT protocol operation is determined by the current network state based on the reliability achieved and congestion condition in the network. If the event-to-sink reliability is lower than required, ESRT adjusts the reporting frequency of source nodes aggressively in order to reach the target reliability level as soon as possible.

If the reliability is higher than required, then ESRT reduces the reporting frequency conservatively in order to conserve energy while still maintaining reliability. This self-configuring nature of ESRT makes it robust to random, dynamic topology in WSN.

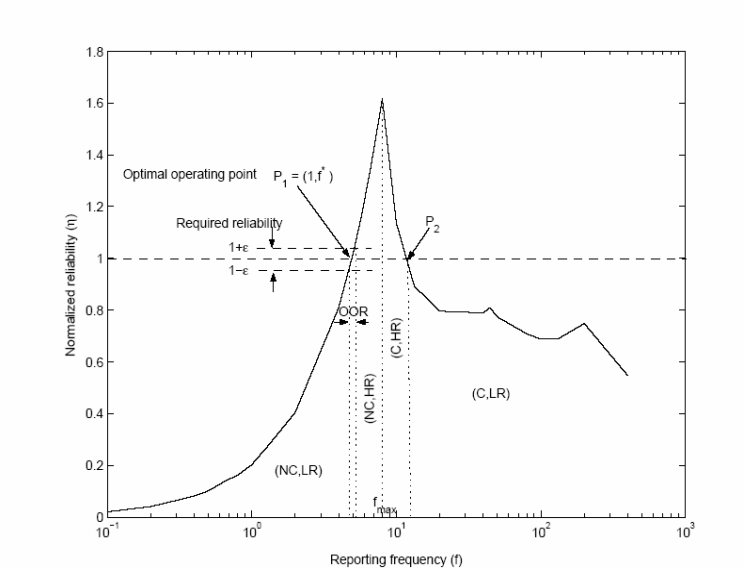

Fig. 2. 3 The five characteristics regions in the normalized reliability, *n*, vs reporting frequency, *f*, behavior

In a decision period, this protocol counts the number of packets r_i received during round i and this value is compared with the desired number R of packets that is needed for sufficient information fidelity. The ratio $n_i = r_i /R$ is a metric of reliability. The goal of the protocol is to keep ni in the interval [1-ε , 1+ε]. Protocol via sink controls the sensor's reporting frequency f. The reporting frequency f_{i+1} to be used for round i+1 is computed by the sink at the end of the round i and is broadcasted (high power sink) to all sensors, which immediately adopt the new frequency.

Protocol operation is divided in five separate states, where f reporting is accordingly computed. Summary of the ESRT protocol operation is presented on the table below.

Network State	Description	ESRT Action
(NC , LR)	No congestion, Low Reliability	Multiplicatively increase f Achieve required reliability as soon as possible
(NC , HR)	No congestion, High Reliability	Decrease f conservatively Cautiously reduce energy consumption so as not compromise on reliability

(C , HR)	Congestion , High Reliability	Decrease f aggressively to state(NC/HR)to relieve congestion Then follow action in (NC, HR)
(C , LR)	Congestion, Low/Equal Reliability	Decrease f exponentially Relieve congestion as soon as possible
OOR	Optimal Operating Region	f remains unchanged

Fig 2. 4 Summary of ESRT protocol operation

e. RAP: A Real-Time Communication Architecture for Large- Scale Wireless Sensor Network. [28]

RAP supports real-time communication in large-scale sensor networks. The network layer of RAP ensures fairness between nodes and improves the ratio of packets that make their latency deadlines. To accomplish this task, RAP provides support for deadline- and distance-aware packet scheduling.

Packets originating from deeper in the network sources, have higher priority than packets originating from close to the sink sources. While RAP focuses on the network's ability to meet deadlines, our work focuses on managing overload and congestion in a sensor network

The architecture of RAP is shown in Fig 2.5

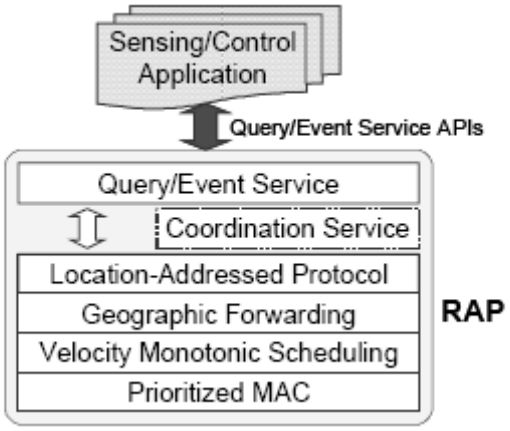

Fig. 2. 5 The RAP communication architecture

Sensing and control applications interact with RAP through a set of Query/Event Service APIs. A Query/Event Service layer submits the query or event registration to an area. The Query/Event Service at the sensors in that area then (periodically or a periodically) sends

query results back to the base station. If an event is registered, the query is started only if the registered event happens.

The sensor-based communication is supported by a network stack, including a transport-layer Location- Addressed Protocol (LAP), a Geographic Forwarding (GF) routing protocol, a Velocity Monotonic (packet) Scheduling (VMS) layer, and a prioritized MAC. This network stack embodies a set of efficient and localized algorithms to reduce the end-to-end deadline miss ratio of sensor-base communication. This network stack is the focus of this paper.

The coordination service is responsible for dynamic group management and data aggregation among sensors (e.g., multiple sensors coordinate to determine the location of a target through triangulation).

f. Directed Diffusion [14]

Directed diffusion is data centric (design philosophy than a protocol) because all communication is for named data. In a directed diffusion-based network all nodes are application-aware. This enables diffusion to achieve energy savings by empirically selecting good paths (small delay) by caching and processing data in-network (e.g., data aggregation).

Directed diffusion has four basic elements: interests, data messages, gradients, and reinforcements. An interest message is a query from a sink node to the network, which indicates what a user wants. It carries a description of a sensing task that is supported by a sensor network.

In sensor networks data is the collected or processed information of an event (e.g. physical phenomenon), is named (addressed) using attribute-value pairs and a sensing task is diffused throughout the sensor network as an interest for named data. This dissemination sets up gradients within the network designed to "draw" events (i.e., data matching the interest).

Specifically, a gradient is direction state created in each node that receives an interest. This direction is set toward the neighboring node from which the interest was received. Events start flowing towards the sinks of interests along multiple gradient paths. Finally, the sensor network reinforces (via the sink) one, or a small number of these paths.

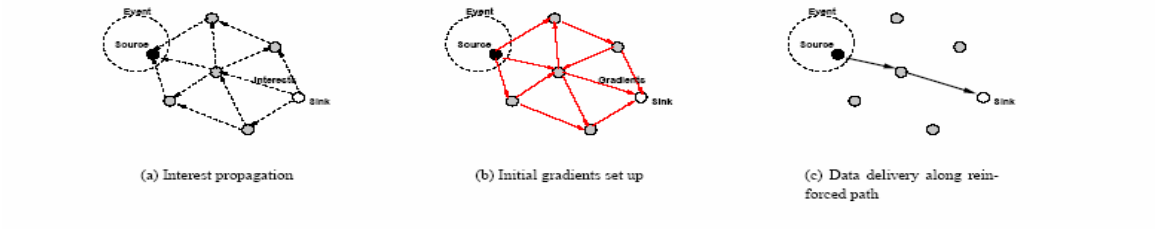

Fig. 2. 6 A simplified schematic for directed diffusion

Fig.2.6 shows interests propagation, initial gradients set –up and, finally, data delivery from source to destination.

Directed Diffusion could be responsive to congestion. Although it is not specifically designed for congestion control it may adapt for this task (to some degree):

- By reinforcing paths with small delays
- It can use in-network data reduction techniques such as aggressive aggregation when congestion is detected.

2.2 Comments on Related Work

The methods listed below are directly focusing on Congestion Handling problems

- CODA (COngestion Detection and Avoidance)
- Congestion Control and Fairness for Many-to-One Routing
- Mitigating Congestion Control In WSN

The following methods, although their main purpose is not Congestion Handling they deal with it in some extend

- Directed Diffusion
- RAP: A Real-Time Communication Architecture for Large- Scale Wireless Sensor Network
- Event-To-Sink Reliable Transport (ESRT) Protocol

[24] CODA: seems to be a very well- designed protocol for congestion detection and handling. It consists of two congestion control mechanisms that, in isolation, seem to be

insufficient. Nevertheless, when cooperating, they complement each other with very interesting results. Channel sampling used for congestion detection maintains low or no cost operations during no congestion conditions and it also seems to work well enough when congestion occurs. Rate adjustment is done for controlling congestion.

This feature (rate control) renders CODA inadequate for the nature of networks that we examine in our Thesis. Rate Control in not acceptable for the reasons that are explained in section 1.5 and Chapter 4.

[25] Congestion Control and Fairness for Many-to-One Routing: it is a distributed algorithm that is scalable and robust enough in nodes failures. and is also seems to be fair to all data sources on the network. On the other hand, it uses sub-tree size computations and in order to work well, it needs some modifications on the MAC Layer Protocols to be able to accept packets from the parent nodes. It is also strictly depended on the sub-tree size information and network topology (childs - parent).

This approach has many similarities with the algorithm that we suggest in this Thesis. Nodes are "placed" in a hierarchical tree from the source to the sink and they report data only when a specific event is taking place. The main contrast with our algorithm is that in the context of this paper congestion is also attempted to alleviated, with data rate reduction techniques.

[26] The Fussion Technique, which is suggested, seems to be effective but in no applicable to the nature of networks that we deal with in this Thesis.

Although, the Hop- by- Hop flow control for congestion detection and Prioritized MAC can easily become part of our algorithm, Rate limiting, which is used for congestion alleviation, is not accepted.

[27] In ESRT, most of the protocol complexity is into the sink, and in addition the sink doesn't need to know the global network properties like the number of nodes. The sensor nodes just need to be able to receive the sink's commands to set new reporting frequencies. The protocol seems also to have some disadvantages: it only takes into account the reporting rate, and, on the other hand it is assumed that when sink broadcasts it is listened everywhere, which is often no true. Furthermore it uses a common regulation rate (for worst case hotspot area) for all sources in the network and this seems to be very conservative for sources with no congestion. Finally it doesn't quickly response when congestion is transient, because feedback latency is large.

ESRT can in no way apply to the specific event networks that we deal with in our Thesis. Its whole philosophy is very different. The main reason is the fact that the whole complexity is in the sink. According to our study (Chapter 4) we believe that congestion problem in WSN is easier to and can be more efficiently solved locally and not globally.

[28] RAP is a suggested Communication Architecture for Sensor Networks. The congestion control is achieved by minimizing a packet deadline miss ratio by providing general service APIs to an application service, and Velocity Monotonic Scheduling (VMS) to priorities packet urgency based on both deadline and distance-aware. VMS also solves a fairness problem, as the more distant packets tend to have higher priority than the closer ones. Each packet priority is based on a static or dynamic requested velocity. It in not clear yet whether this technique is scalable for all networks.

This paper does not focus on the Congestion Control Problem and, of course, it is infeasible to apply it in our case. Nevertheless as a future work, the adoption of its fairness technique in our algorithm can be studied.

[14] Directed Diffusion is not specifically designed for congestion control, but it may adapt to this to some degree, using reinforced paths with a small delay (therefore choosing paths with least traffic). However this strategy will penalize distant data sources from the sink. Furthermore, because it uses sink –initiated congestion control, it reacts much slower than the other localized methods

Directed Diffusion can not apply to event- based networks due to the sink- initiated data requests.

Chapter 3

Congestion

3.1 Congestion in Wired Networks

Network congestion occurs when offered traffic load exceeds the available capacity at any point in a network. Basically, in wired networks, the main issue is the effective and fair allocation of resources among the competing nodes. The resources are the bandwidth of the links and the buffers on the routers where the packets are forwarded.

The packets are placed in a queue in the routers, waiting their turn in order to get transmitted. When too many packets are expecting to use the same link, the routers queue overflows and packets have to be dropped. When such drops are frequently happening, the network is **congested.**

In wired networks congestion can be detected with the following ways:

a. By monitoring Queue length: If the routers' buffers begins to grow up and the transmission rate is lower that the packets' arrival rate, this is an indication that congestion is going to happen

b. By monitoring packet losses: If the routers begin to drop packets due to overflow, this is also an indication of congestion

c. Indirectly, by monitoring the trend of throughput or response time (Fig 3.1):

As the load in the network increases, the throughput of the network increases linearly up to a “knee” point. Then, as the load increases more than the knee point, the throughput reduces.

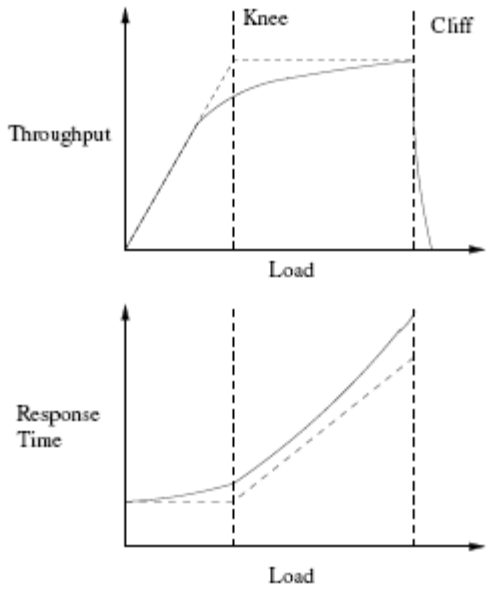

Fig. 3. 1 Network Performance versus load

Congestion control in wired networks is usually achieved using end-to-end and network-layer mechanisms acting in concert. However, this approach does not solve the problem in wireless networks, because concurrent radio transmissions on different ``links'' interact with and affect each other, and, because radio channel quality shows high variability over multiple time-scales

3.2 Congestion in WSN

As stated in Chapter 1 a wireless sensor network is constrained by computation ability, memory space and energy supply. Congestion is an event which adds to energy waste in these networks, causes even, network collapse, when it takes place on a large scale.

Specifically when a packet is dropped, the energy spent by upstream sensors on the packet is wasted. The further the packet has traveled, the more the waste. When the buffer at a sensor x is already full and if the upstream neighbors attempt to send data to x, their efforts

(and energy) are deemed to be wasted and, worse yet, counter-productive. For instance, their RTS packets may collide with nearby transmissions, causing throughput reduction of other sensors.

Thus measures must be taken in order to avoid or control congestion when this happens in a network.

Generally, congestion may appear in two patterns (types):

Pattern A: In a particular area, densely deployed nodes attempt to transmit simultaneously (triggered from a specific event). This causes interference at the listening node which is in range of the transmitting node.

Pattern B: In a particular node, the buffer that holds packets to be transmitted overflows due to continuous packet transmission from other nodes.

a. Mechanisms for Congestion Detection and handling

There are three mechanisms for congestion handling

- Congestion detection, which judges if congestion occurs, using two indicators: buffer occupancy and channel utilization.
- Congestion avoidance, which studies how to prevent congestion (usually controlling rate either locally or end-to-end) from happening
- Congestion alleviation, which reacts to congestion in order to eliminate it

Generally, congestion control mechanisms are based on three “techniques”

- End- to- End
- Path based
- Hop-by-Hop

End -to- End: In this approach, a congestion control mechanism is embedded in the transport layer protocol [29]. The receiver can figure out how many packets and which have arrived, by checking their sequence number, as indicated by ACK packets. Additionally, mechanisms like window- size or feedback information can be used to inform the sender for the possibility of congestion in a certain path.

The End- to- End approach is difficult to apply in WSN due to the fact that the path from the sender to the receiver is usually very long, and mostly many to one (many sensors to a sink). Furthermore the numbers of data (event) packets that are generated when an event is taking place are huge. This fact renders impossible the transmission of acknowledgement packets for all the data packets. To cope with this problem new efforts are taking place for end- to- end reliable event transfer.

Path based: This approach tries to improve some of the drawbacks of the end- to end solution. The main difference is the faster recognition of congestion. When congestion is detected, the source is informed backwards by hop- by- hop signals.

This solution can work, if congestion is detected near the source. If congestion is detected near the sink, which is the most usual event, then the same problems as in the end- to-end approach appliy.

Hop- By- Hop: This approach employs hop- by- hop detection and prevention**.** The difference, compared to the other approaches, is that there is no need for a long backward path to sender. The problem of congestion is solved locally, through communication between the neighbor nodes. This hop-by-hop mechanism does not focus on a particular transmission. It focuses on the detection of congestion and the transport of corresponding information to all neighbors. There, appropriate countermeasures must be taken in order to prevent any further congestion. Actions can vary from the random discard of new packets to further signaling of the congestion situation towards the next neighboring nodes.

3.3 Study of Congestion in sensor networks through simulation in OPNET

In order to investigate the behaviour of a sensor network during congestion we created specific scenarios using the OPNET simulation tool. For our work, we used MANET Station Advanced (Fig 3.2), accordingly customizing it, to emulate a real sensor node.

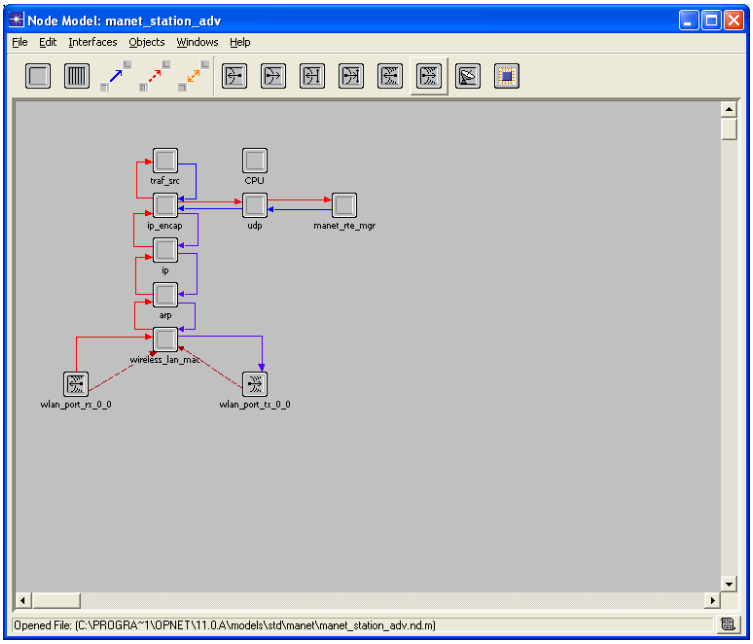

Fig. 3. 2 Manet station advanced in OPNET

Specifically the followings parameters were used:

I. Network consists of sensor nodes in a square area 500 x 500m having the attributes listed below:

- Buffer size 1024 B
- Channel capacity : 2Mbps
- Packet size : 512 B
- Max distance 50m (Rx Group)
- Tx power : 1mw
- Event generation (120 s - end of simulation) using bursty (geometric distribution 0.8)

In Fig 3.3, Rx Group is customized for all transceivers to have maximum distance radio coverage 50m.

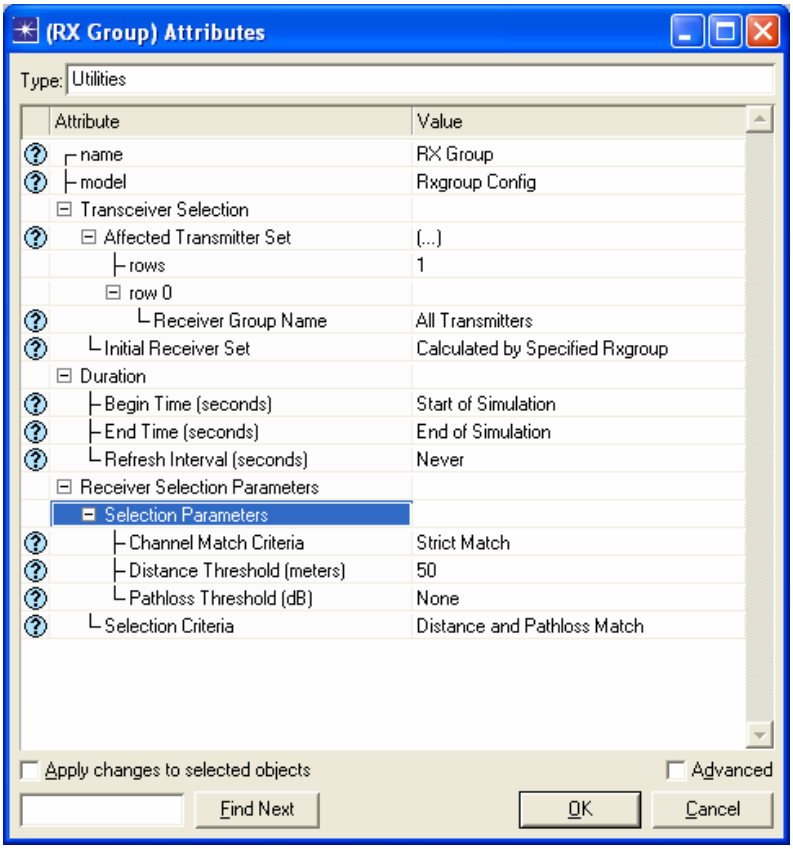

Fig. 3. 3 Rx Group attributes

II. DSR routing protocol

The Dynamic Source Routing (DSR) [30] protocol has been used for our simulations, because it is used in the domains of sensor networks.

DSR is a simple, but efficient, routing protocol specifically designed for use in multi-hop wireless ad hoc networks of mobile nodes. By using DSR, the network is completely self-organizing and self-configuring, and doesn't require any existing network infrastructure or administration. Network nodes cooperate to forward packets to each other to allow communication over multiple "hops" between nodes that are not directly within wireless transmission range of one another. As nodes in the network move about or join or leave the network, and as wireless transmission conditions (such as sources of interference change), routing is automatically configured and maintained by the DSR routing protocol. Since the number or sequence of intermediate hops needed to reach any destination may change at any time (due to battery, interference, etc), the resulting network topology may be quite rich and rapidly changing.

The DSR protocol allows nodes to dynamically discover a *source route* across multiple network hops to any destination in the ad hoc network. Each data packet in the network carries the complete in its header, ordered list of nodes through which the packet must pass, allowing packet routing to be trivially loop-free.

a. Scenarios

In this section, through simulations that we conduct, we will demonstrate the ways that congestion appears in WSN

The approach used in the scenarios consists of one sink (node 66) and several sensors.

Scenario 1 : The topology presented in Fig 3.4 has been created.

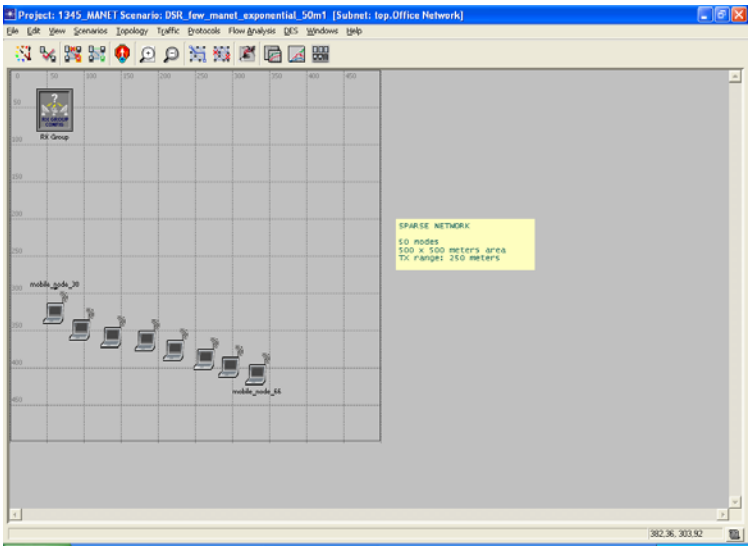

Fig. 3. 4 Topology for Scenario 1

On the above scenario, we tried to simulate a real Manet network, in order to investigate its behaviour (specifically packets drops) with the selected topology and to compare it with a sensor network (scenario 2). We deliberately create this topology, because each node is able to transmit data only to the node next to it. So, we will be able to understand exactly the point and reason of congestion (if appears) in the network

For this scenario, the default node's buffer size is 16 MB.

Scenario 2: The same topology, as is scenario 1, has been used, but in this scenario Manet nodes have been customized accordingly to emulate sensor nodes according to the changes stated in the beginning of section 3.3.

In this topology, every node is forced to route its packets through its neighbour node. This happens because every node in its radio range has only one node on the direction to the sink. (Fig 3.5)

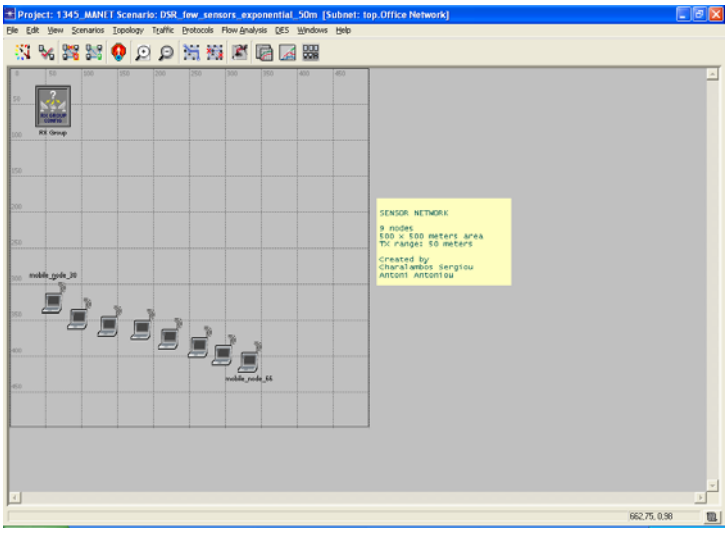

Fig. 3. 5 Topology for Scenario 2

Results

Comparing the results of Scenario 1 and 2 concerning packet drops the graph in Fig 3.6 is created.

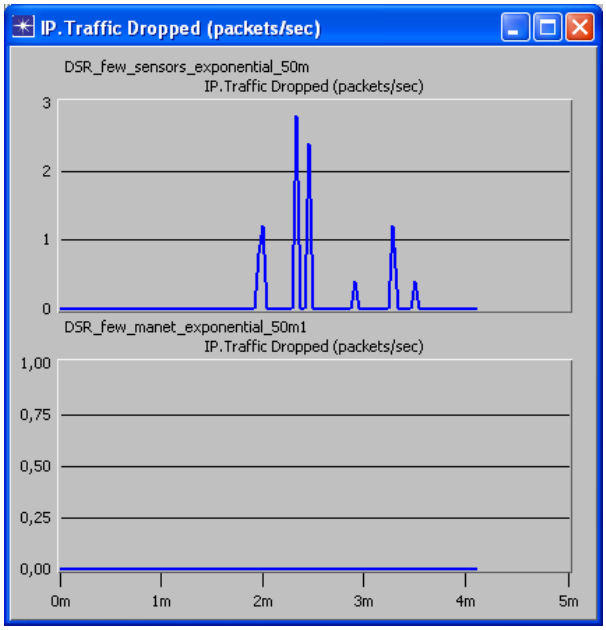

Fig. 3. 6 Packets/ sec dropped for Scenario 1 and 2.

It is clear that in scenario 2 (sensors network) congestion occurs and results in packets drop. The change on the behaviour between scenarios 1, 2 happens mainly due to the large difference on nodes buffer size (Manet 16 MB, Sensors 1KB).

Scenario 2 - Step 2: In this step we concentrated on congestion Pattern B.
In this topology five successive sensors (left to right) generate traffic to the sink and the other two nodes are sleep nodes.

In Figure 3.7 the specific nodes which drop packets, are presented.

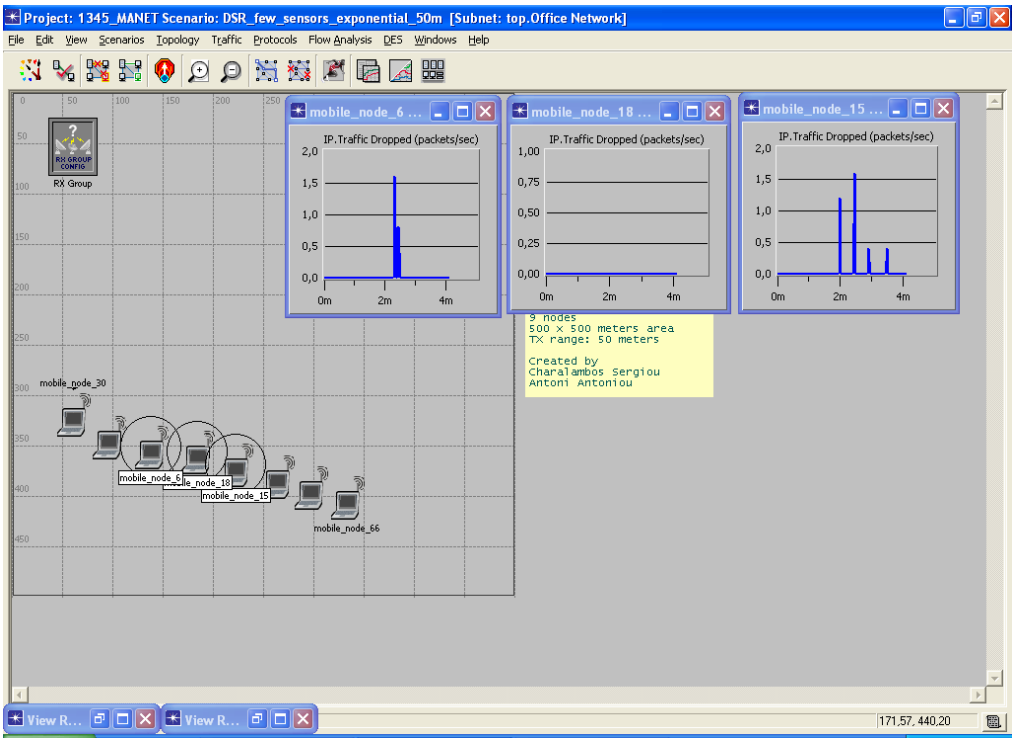

Fig. 3. 7 Results for Scenario 2- Step 2

As it is shown in Fig 3.7, node 6 and node 15 (in the middle of the network) are receiving packets in a higher rate that they can transmit and store. Eventually the excess packets are dropped; as it is shown in sub graphs. **This exactly situation is the one that we are going to handle with our proposed algorithm in Chapter 4.** Nodes are becoming congested due to the fact that their buffer overflows.

Scenario 3: In this scenario, we tried to represent Pattern A congestion, that is when a hotspot occurs due to interference that derives from simultaneous transmissions.

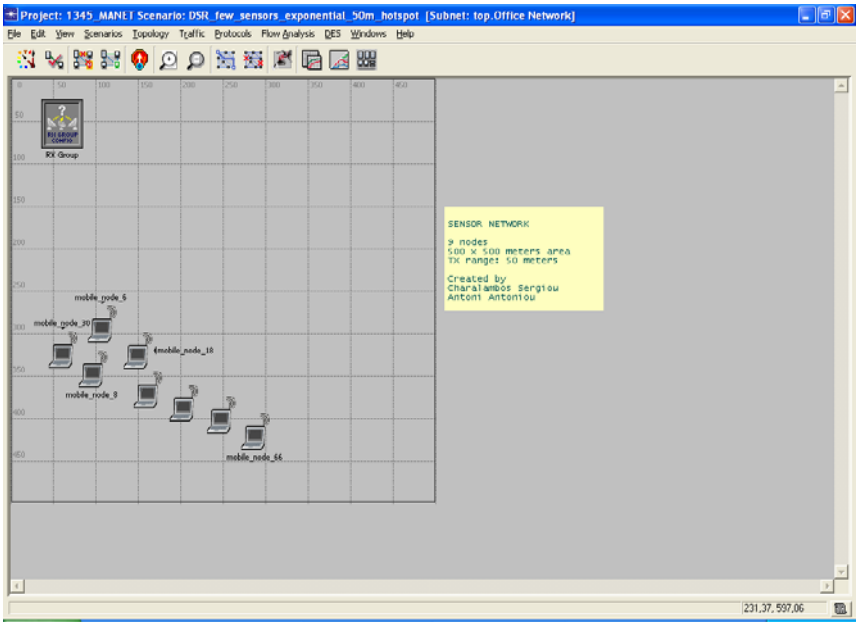

Fig. 3. 8 Topology for Scenario 3

In the topology of Fig 3.9, nodes 30, 6, 8, 18, 15 have packets drops due to this kind of congestion. In this situation, the above mentioned nodes generate traffic with the remaining nodes being in sleep mode.

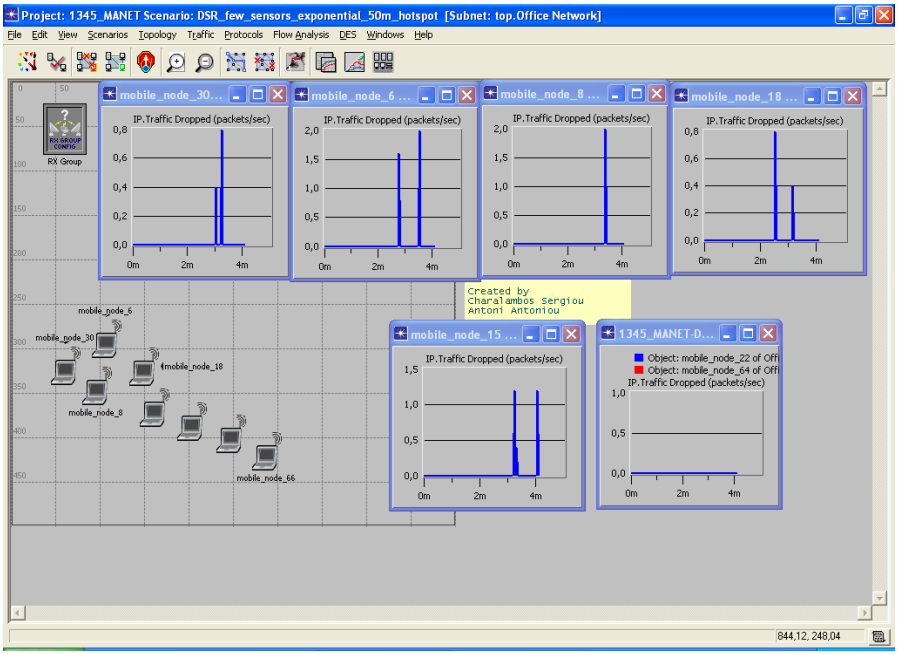

Fig. 3. 9 Packets/ sec dropped for Scenario 3

Scenario 3- Step 2: We created 4 different instances (one node generate traffic, then two nodes, three nodes, four nodes) of the scenario.

The results are depicted in Fig 3.10

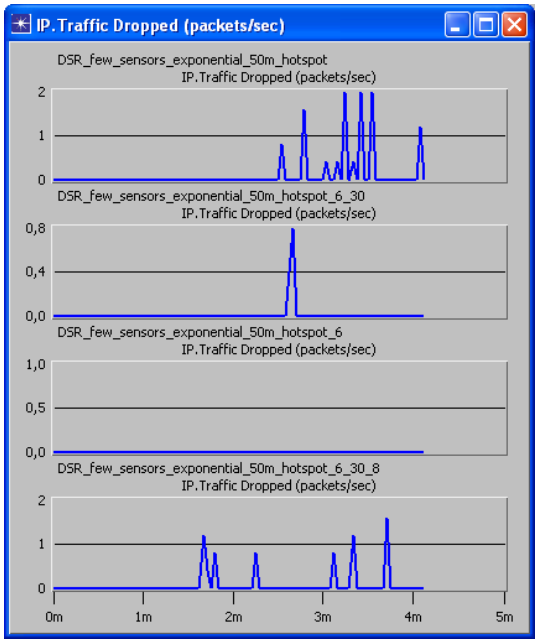

Fig. 3. 10 Results for Scenario 3-Step 2

It is clear that each time a new traffic generator node is added, packets drops happen more frequently. This results from the fact that interference increases.

By exporting data from Fig. 3.10 on a MS Excel Spreadsheet we present how the hotspot interference contributes in congestion occurrence (packets drops).

In Fig. 3.11 it is shown that when node 6 is the only sensor node that transmits there are no packet drops. When nodes 2 and 30 transmit simultaneously, then less than 2 packets are dropped during the simulation, and when nodes 6, 30, 8 are simultaneously transmitting, more than 6 packets are dropped. The situation gets worse when all four nodes are transmitting.

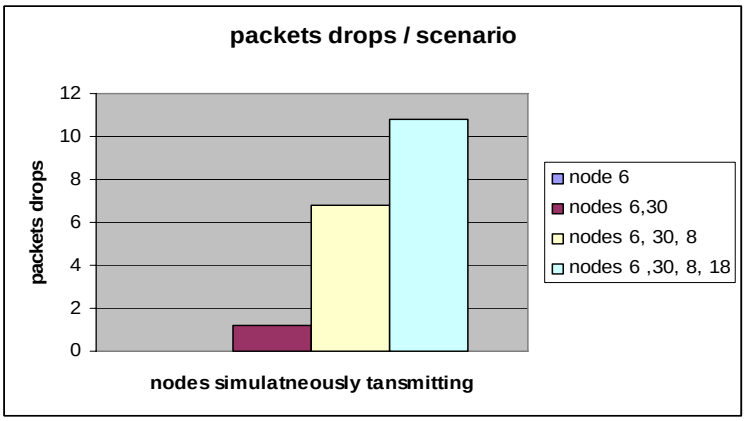

Fig. 3. 11 Packets Drops' per Scenario

It's obvious that the more simultaneous transmissions occur, the bigger the congestion problem –packet drops and, inevitably, the more energy is consumed.

Scenario 4: In this scenario we tried to simulate a situation where the traffic to the sink is forwarded through one node (bottleneck).

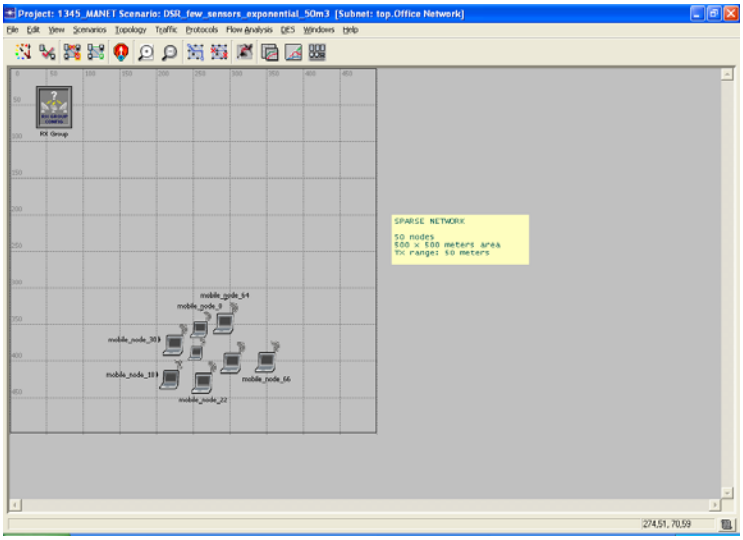

Fig. 3. 12 Topology for Scenario 4

Results are depicted in Fig. 3.13

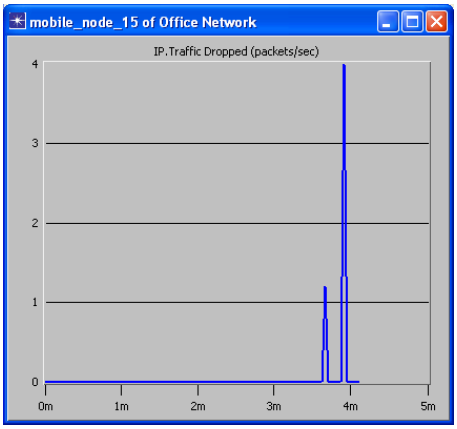

Fig. 3. 13 Packets/ sec dropped for node 15

Most packet drops occur in node 15, which is the bottlenecked node. Some drops also occur on node 30 due to interference.

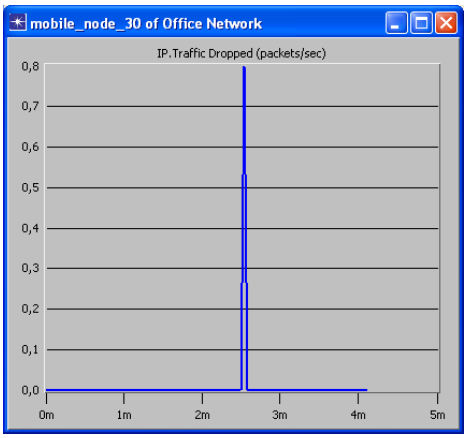

Fig. 3. 14 Packets/ sec dropped for node 30

Chapter 4

Proposed Congestion Control Scheme

In this chapter we present a scalable and distributed framework for minimizing congestion and assuring reliable data transmissions in ***event based networks***. Event based networks is a particular category of sensor networks. In these networks, reports are produced only upon the observation of a specific event. This event should satisfy a pre-specified condition.

A typical example of these networks, as we also stated in section 1.5, is the increase of temperature in a forest. Sensor nodes should immediately begin the transmission of data packets to the sinks (sinks could be the area's neighbor fire stations).

Especially in this crisis state, the sudden traffic increase may lead to congestion in the network. When congestion occurs, the network will enter into an unstable state and packets will be randomly dropped. This is particularly undesirable, because the data generated during the crisis state are of great importance, often critical, to the applications [31].

In Chapter 4 we propose an algorithm which is able to control a congestion situation and which is efficient enough to safely transmit almost all these valuable data to the sinks. It becomes clear that this algorithm must be able to transmit almost all the data packets, without throttling the source nodes' data rate. Throttling the data rate could be

proved fatal for these networks, due to the fact that each data packet provides the network with updated information concerning the monitored event (i.e spreading of fire in a forest).

4.1 General

In event-based environments there is a need for controlling the sudden traffic increase. Due to the nature of these environments, sudden traffic increase occurs when the monitoring event is happening. This high generation of data packets is usually uncontrolled and often leads to congestion.

When congestion occurs, the network may enter into an unstable state. In this state the networks' behavior is unpredictable. If there is no congestion control mechanism the network's reaction to congestion is the random drop of data packets [31]. Beside the obvious energy consumption, the major drawback in this method is that the packets which are produced during this state are of great importance. So, the need for early congestion prediction and alleviation is obligatory.

In different studies [27, 32] it is observed that the number of nodes with occupied queues grows if congestion gets worse. When congestion is detected, the sources should be notified in order to take action to face congestion. The most popular approach for this notification is the transmission of a control packet to the source, from the sink.

Nevertheless this technique, from the sink to the source, may prove fatal for the network, especially if congestion arises late in the path from the source to the sink. The number of nodes with occupied queues will increase and probably the network will collapse.

To prove this let's consider Fig 4.1 [33]

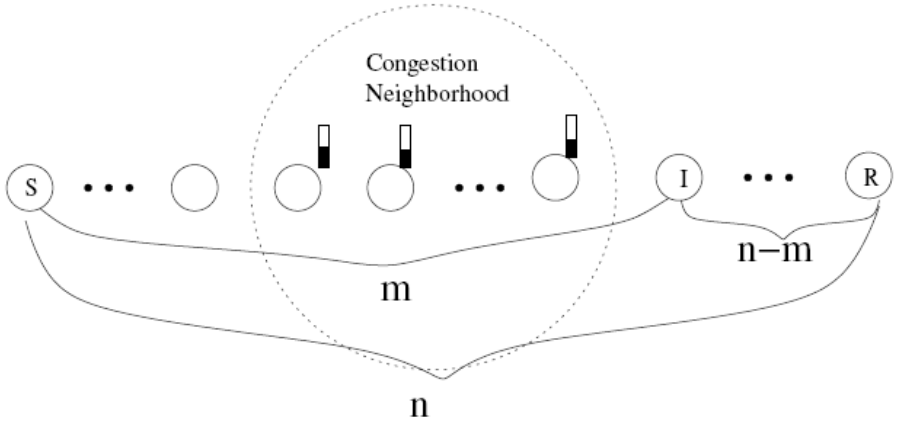

Fig. 4. 1 A single connection in WSN

In this figure a source and a sink is used. Node S, R, and I denote the source, sink and intermediate node respectively. The total number of hops along the path is **n**. The source is **m** hops away from the intermediate node I, which detects the congestion.

Let's check the signaling cost. It is assumed that energy dissipation in transmission of one packet per hop is a constant, regardless of the configuration of MAC and link layer. It is denoted by **є**.

In Figure 4.1, it is easy to deduce that the signaling energy expenses per cycle **Tcn** is

$$\mathbf{E = m.\ є}$$

and the propagation delay is

$$\mathbf{T = 2m.\ t_h}$$

where $\mathbf{t_h}$ is the propagation delay per hope.

It is clear that when **m** approaches **n** the signaling energy that is needed to inform the source for the congestion is increasing. Furthermore the propagation delay is increasing, which, in its turn leads to time lag concerning the information of the source for the congestion. Fatally the source will continue to send data packets in the same rate until the reception of the control message and eventually the number of nodes with occupied queues will increase.

Bearing in mind these inputs, a congestion control scheme is needed which is able to solve the problem locally and immediately without loosing any data packets and of course without wasting so much energy. Other works that have been published like CODA [24] and fussion [26] which also try to face the problem locally by using backpressure messages**; they manage to alleviate congestion by throttling the data rate**. In our effort, due to the mission of the event-based networks, reduction of data rate is prohibited. Sources must continue to send their data with the rate that they sense the events.

4.2 Scheme Design

Bearing in mind the related work and the unique features of WSN, especially the limited power and storage resources, a new algorithm is proposed attempting to solve the congestion problem in these networks.

For the development of this algorithm we involve a major input, which exists in sensor networks. This is the plethora of unused recourses.

Based on this major input, our target is to develop an algorithm which is feasible and can be embodied in the sensor motes that are used in real networks (mica- motes etc). The effort to alleviate congestion situation will be enhanced by unused nodes.

The algorithm consists of two parts, the **Alternative Path Creation** (APC) and **Hierarchical Tree Creation** (HTC). The philosophy of these two algorithms is similar. Both of them are based on the creation of alternative paths from the source to the sink, when congestion is going to take place. For the creation of these paths the nodes which are in dormant state are used. APC uses these nodes in a generally random way, compared to HTC where these nodes are placed in a Hierarchical tree from the source to the sink. The final algorithm is called HTAP (Hierarchical Tree Alternative Path) and is a combination of these two algorithms.

Following we introduce these algorithms explaining why we choose their combination.

a. Alternative Path Creation

The initial idea for the creation of this algorithm derived from a particular concept of the theory of Dynamic Alternative Routing (DAR) of R.J. Gibbens, F.P. Kelly, and P.B. Key used in public telephony[34]. In this concept it is stated that if you have a good route without problems, stick to it, until something goes wrong.

With some major changes in the implementation, this concept is adopted by us in the case of congestion control in WSN.

As reporting above, according to publications [31], it is stated that in wireless sensor networks there are many nodes, which, in many cases, when a specific event is detected they are not taking part in the path from the source to the sink (in the routing process). This is due to the fact that these nodes are far away from the event.

The main target of this algorithm is to take advantage of these nodes and use them for the creation of alternative paths from the source to the sink. The creation of these paths will unload the highly dense parts of the network and will lead the data packets safely to a sink though other routes.

The basic theory of this algorithm is that a source node keeps transmitting data packets to a specific node in a level higher than itself, until it receives a control message from this node that is not able to handle any more packets. This is either because is going to become congested or due to the fact that it will soon run out of power. In such a case, with a procedure which is described in the algorithm the source node will search in its neighbor table and find the most appropriate node to transmit the further data.

To explain the concept let's consider the figure below.

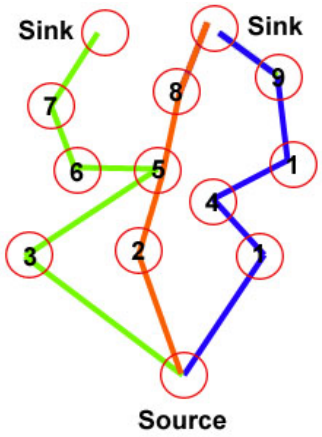

Fig. 4. 2 Transmission from Source to Sinks

In this figure, we consider that there is a source which is sending data packets. Initially, it is sending data to the node numbered 1. The data packets follow the blue path to a sink. As soon as node 1 calculates, that is going to be congested is sending a control packet to the source in order to inform it about the fact. When source node receives the control packet it is then searching in its neighbor table, which includes nodes 1, 2 and 3. to find the proper node to keep sending data (how the neighbor node is created and which is the proper node is explained in the algorithm). The sending node is now transmitting to node 2.

It must be stated that this procedure is taking place in all the levels and not only in the source. For example, if node 8 becomes congested, it will send a control message to the transmitting node (in this example is 5) to inform it about the fact and node 5 will apply the same procedure as the source node in Fig 4.3 (it will forward data through node 6 instead of 8)

I. Implementation of APC

The implementation philosophy of the algorithm follows the steps below:

- A simple hierarchical flooding protocol is used for the formation of the network's topology; the purpose is for each node to discover its neighbor nodes. Through this procedure, each node updates its neighbor table. In addition through this protocol, sensor nodes are theoretically placed in levels from the source to the sink.

- At each packet transmission each node piggybacks its congestion state (buffer occupancy). The neighbor nodes overhear the packet transmission [26] and update their neighbor tables with this information.

- During the triggering of an event, the source node begins transmitting data packets creating flows to the sink. If the sending data rate is higher than the sending rate that the receiving node can transmit, the receiving nodes will soon face a buffer congestion situation and the results would probably be the random drop of data packets. In order to avoid this situation each candidate congested receiver is sending a backpressure packet to the sender to inform it, that if it continues to transmit packets with the same rate it will soon be congested. So the sender stops the transmission of packets to the candidate congested receiver and searches in its neighbor table to find the least congested receiver in order to continue the transmission of data.

- The transmitting node begins transmitting the data to the alter node. The same phenomenon can happen in each level (between the neighbor nodes). The alternation of receivers leads to the creation of alternative paths.

II. Alternative path creation (APC) issues

We aim to evaluate the effectiveness of the Alternative Path Creation (APC) algorithm as to its effectiveness as a technique to alleviate the congestion problem in WSNs for event triggered networks.

The major **advantage we anticipate** is the fact that it utilizes nodes which in other ways would be in a dormant state. Basically, it increases the resource provisioning in the network with the energy and buffer capacity of these nodes.

This solution seems to create good results concerning congestion alleviation (Appendix B). But, this is not the only fact in an algorithm.

The reason we haven't used this algorithm solely is due to a problem it encounters. The problem is that it scales poorly with dense networks where there are many nodes. More specifically the problem has to do with the mean time of the transmission of packets from the sources to the sink, which in some cases is highly increasing. This means that a packet could be received "late" causing problems in the accomplishment of the mission. The reason is explained below

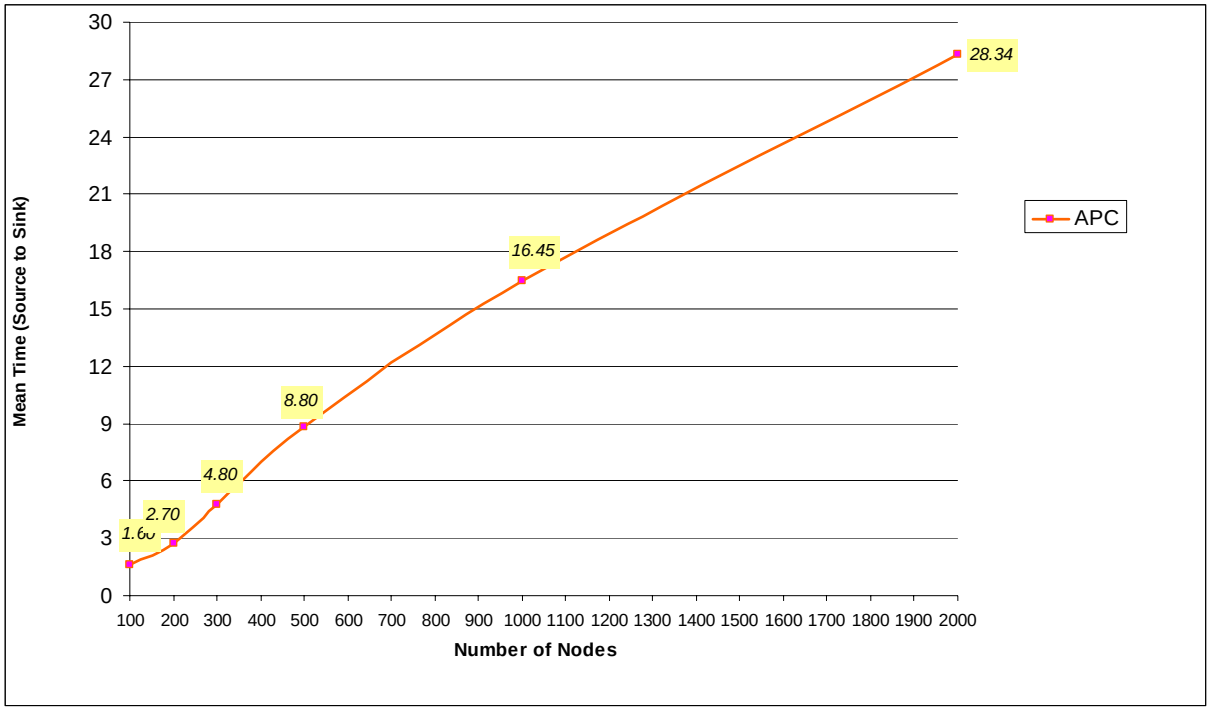

Fig. 4. 3 APC's Mean Time (Source to Sink) vs Number of Nodes

Each node is transmitting these packets like "hot potatoes" meaning that nodes do not have perception about the congestion situation in other fields on the network. This algorithm has the perception of only the level above. The lack of knowledge of the rest network may lead the packets to a worst situation than the one that they are in first place meaning that they may traverse through many other nodes in order to reach the sink (certainly in any case they avoid hotspots due to the fact that they always choose the least congested node). Fig 4.4 explains this situation.

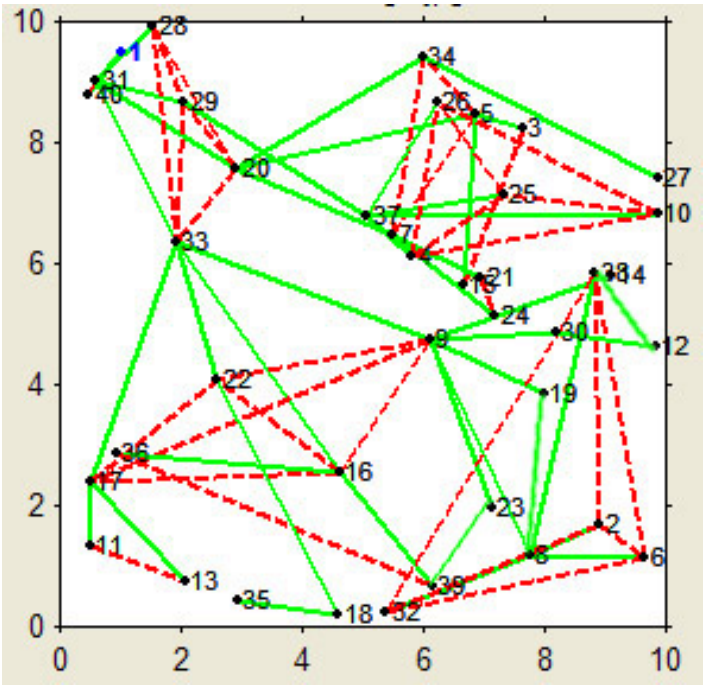

Fig. 4. 4 APC snapshot

In Fig 4.4 node 1 is the source and node 6 is the sink. Node 38 is just one hop away from the sink. If, for example node 38 becomes congested, packets from node 30 can be forwarded to sink either through node 32 (the best case, one hop) or following the path 30, 9, 22, 36, 39, 32 following 6 hops. If exactly after the forward of packet to node 30, node 38 reliefs from congestion, the next packet which carries the latest information will be received by the sink in one hop, and the previous packets will become a "stale" packet.

b. Hierarchical Tree Creation (HTC)

In order to correct the main disadvantage of APC algorithm a modification is proposed. The Hierarchical Tree Creation.

I. Implementation of HTC

This algorithm consists of two main steps:

- **Route Creation:** In this step a hierarchical tree is created beginning at the source node. Each node is assigned a level according to the hierarchical tree. The source node is assigned a level 0 and broadcasts a *level_discovery* packet. Sensors that receive this packet are handed as children to the transmitter and are set as level 1 (they will ignore subsequent *level_discovery* packets). Each of these nodes broadcasts a *level_discovery* packet, and the pattern continues with the level 2 nodes etc. The source when it receives the *level_discovery* packet updates its neighbor table.

- **Flow Creation:** Connection is established between each transmitter and receiver using a 2-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. This connection is performed using a 2-way handshake**.** Having a source node A and a receiver B, node A sends a first packet to B. When node B receives this packet, it sends an **ack** packet back to A. In this **ack** packet the node B piggybacks the congestion state at the moment. In this way, the source node is aware of the congestion state of all the children and is also able to forward them data packets. When the congestion state of children changes to a pre-specified limit (in our work is set to 90% of the buffer of each node) this node updates its congestion state by sending a packet to the source node.

II. Hierarchical Tree issues

According to the simulations that we conducted, this algorithm also reacts effectively to congestion (Appendix B) and the mean time for transmitting packets from the source to sink, reduces compared to APC algorithm (Fig 4.5). This mean that HTC algorithm is able to overcome the disadvantage of APC concerning the transmission time in dense networks.

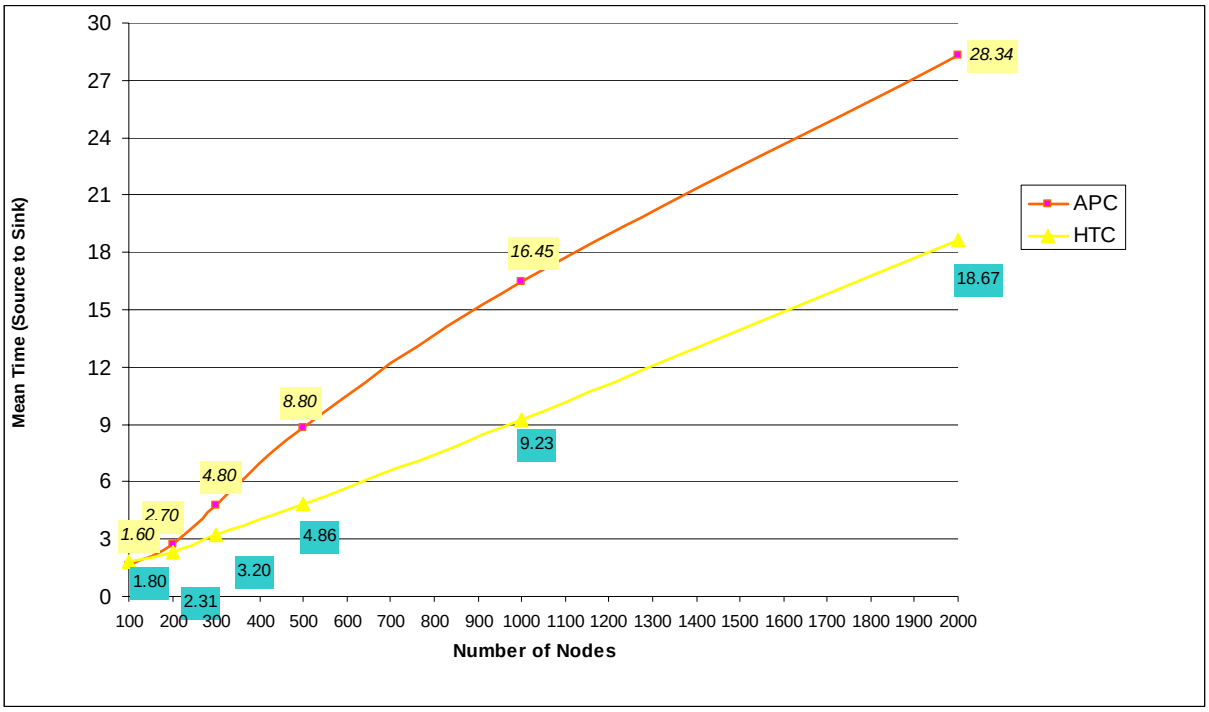

Fig. 4. 5 HTC's vs APC's Mean Time (Source to Sink)

In this case, the packets follow the hierarchical tree which is created at the network deployment. Through the nodes above it each node has perception of the whole network through the routes that have been created in the route discovery phase. Each packet follows specific flows in order to reach the sink.

When congestion is going to happen in a specific receiver node, this node sends a control packet to the transmitter node to inform it to change destination node. The transmitter node searches in its neighbor table and finds the most appropriate node, towards which it begins the transmission of data. The procedure is exactly the same as in APC algorithm. The difference is that according to this algorithm route from the source to the sink, pre- exist and followed by the packets.

However as in APC so in this algorithm, a main **disadvantage** exist. This is the energy consumption compared with APC algorithm (Fig 4.6). The 2-way handshake requires each node to receive a packet and send one in response. For a source node A with children B and C, node A broadcasts level discovery packet and then connection packet. Nodes B and C receive both packets and then transmit an **ack** packet to node A, piggybacking also their congestion state. This example uses 4 transmissions and 4 receptions.

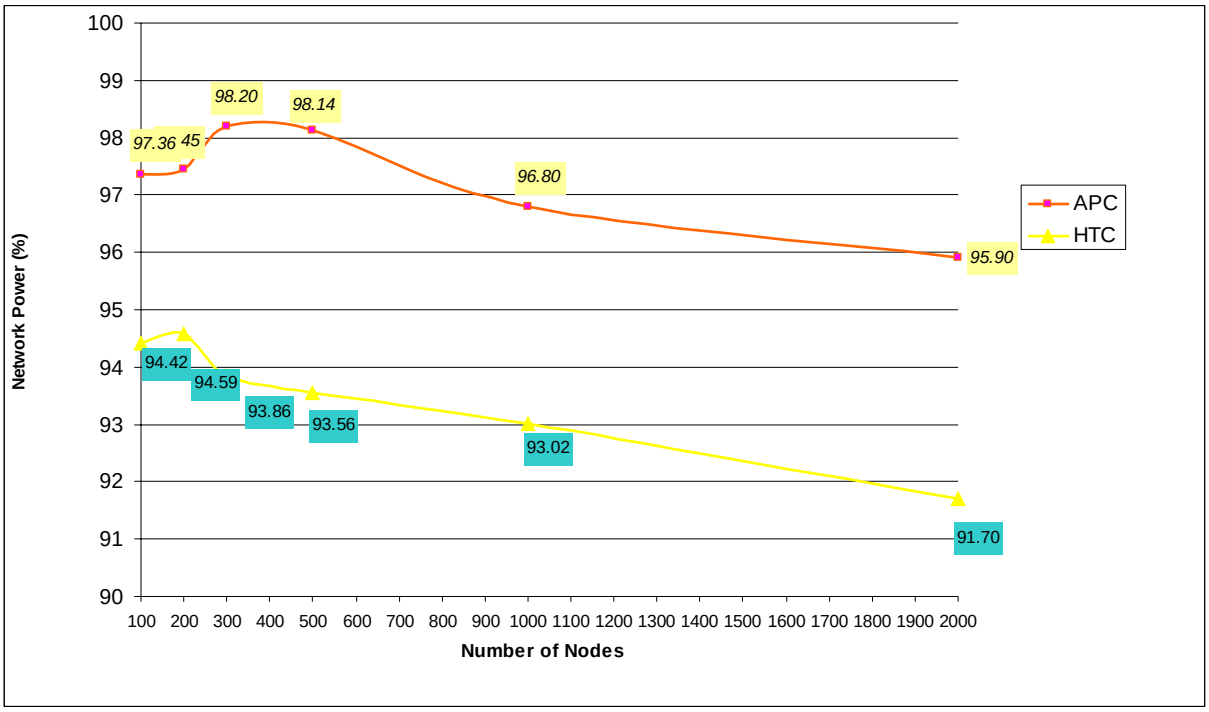

Fig. 4. 6 APC's vs HTC's Network Power

c. The HTAP Algorithm

The HTAP algorithm is a combination of the previous mentioned algorithms. Due to the fact that the advantage of the first algorithm is the disadvantage of the second algorithm and vice versa, this algorithm embodies the advantages of both algorithms in order to eliminate the disadvantages.

This means that in this algorithm the **APC** is going to be applied in combination with the **HTC** when the network is densely deployed.

The density or the sparseness of a network is not an abstract term. A specific threshold needs to be specified by which each node should be able to decide when to apply APC by itself or in combination with HTC. In most of the cases, due to the mission, sensor nodes are dropped by airplanes or fill a place in a field etc. This means that some nodes could not become part of the network, due the fact that they fell too far away from other nodes and there is no other node in their transmission range. Some other nodes could be very densely deployed and other could be very sparse.

This threshold results out of the number of nodes that a specific node has in its neighbor table. If the number is below the pre-specified (for the specific network) threshold

the nodes apply the **APC** algorithm by itself. Otherwise, they apply it with the **HTC** algorithm. Threshold is extracted through simulations in Appendix B.

4.3 Algorithms Description

This section describes the three previous mentioned algorithms. These algorithms are activated when a congestion situation is about to appear in the network.

Moreover in this section the flooding algorithm is described. The flooding algorithm is used at first, at the first deployment of the network; in order for each node to discover its neighbor node and to update their network tables. In order to "assist" the Hierarchical Tree Algorithm a ***level_discovery*** functionality is also added to this algorithm

Flooding Algorithm with Level Discovery

set *neighbor_nodes* to 0
if *current_node* is source node
    Set *level* to 0
    Broadcast *flood_packets* with *level*
else if *current_node* receives *flood_packets* and is accepting them
    set *current_node* to *level+1*
    send *ack_packet* with *current_node_id*
    broadcast *flood_packet* with *current_node_id* and *level*
    ignore subsequent *flood_packets*
else if *current_node* receives *ack_packet*
    neighbor nodes+1

The APC algorithm is described next. After the application of the flooding algorithm, each node is aware of its neighbor nodes. As it was mentioned in the analysis of APC algorithm, the nodes are also aware of the congestion state of their neighbor nodes and are

update their neighbor tables, by overhearing the transmitted packets of the other nodes, in which their congestion state is piggybacked.

APC algorithm is described below

APC: Alternative Path Creation Algorithm

if *current_node* receives **ack** with *congestion_level* full

update *neighbor_table*

search *neighbor_table*

find *node_id* with min (*congestion_level*)

send data packet

if *current_node* receives *congestion_update_message*

update *neighbor_table*

else if *current_node* receives data packet and accepting them

Set *buffer* to *buffer+1*

if *buffer+3*=full *//Buffer is going to be congested*

send **ack** packet with *congestion_level* full

Figure 4.7 shows an execution of APC algorithm

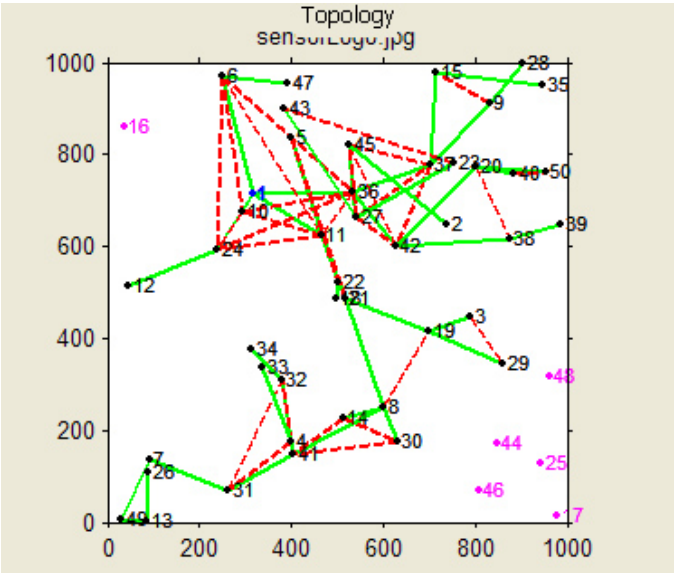

Fig. 4. 7 APC snapshot

In Fig 4.7 data are produced from a source node next to node 1 and are forwarded to sinks 6 and 49. In our simulations we use more than one sink.

The same situation, concerning the neighbor table, applies for the Hierarchical Tree algorithm. Here, the differences are related with level discovery and the connection- oriented situation, in order to form the hierarchical tree. Flooding algorithm assists in level creation as it was described before.

HTC: Hierarchical Tree Creation Algorithm

Create routes from the source to sink
using two way handshake

In case of node congestion
find *node_id* with min (*congestion_level*)
    send data packet
if *current_node* receives *congestion_update_message*
    update *neighbor_table*
else if *current_node* receives data packet and accepting them
    set *buffer* to *buffer+1*
    if *buffer+3*=full *//Buffer is going to be congested*
    send **ack** packet with *congestion_level* full

The combination of the two algorithms to create the Hierarchical Tree Alternative Path (HTAP) algorithm is described below. As it was described before, when the neighbor nodes of a specific node is below a specified threshold the APC algorithm applies, the HT applies otherwise.

In this case, we will consider that the network is facing a congestion situation, all the time.

HTAP: Hierarchical Tree Alternative Path Algorithm

Set *neighbor_nodes_threshold* to [prespecified value]

if *neighbor_nodes*< *neighbor_nodes_threshold*

apply ***APC***

else

apply ***HTC***

Figure 4.8 shows an execution of HTAP algorithm

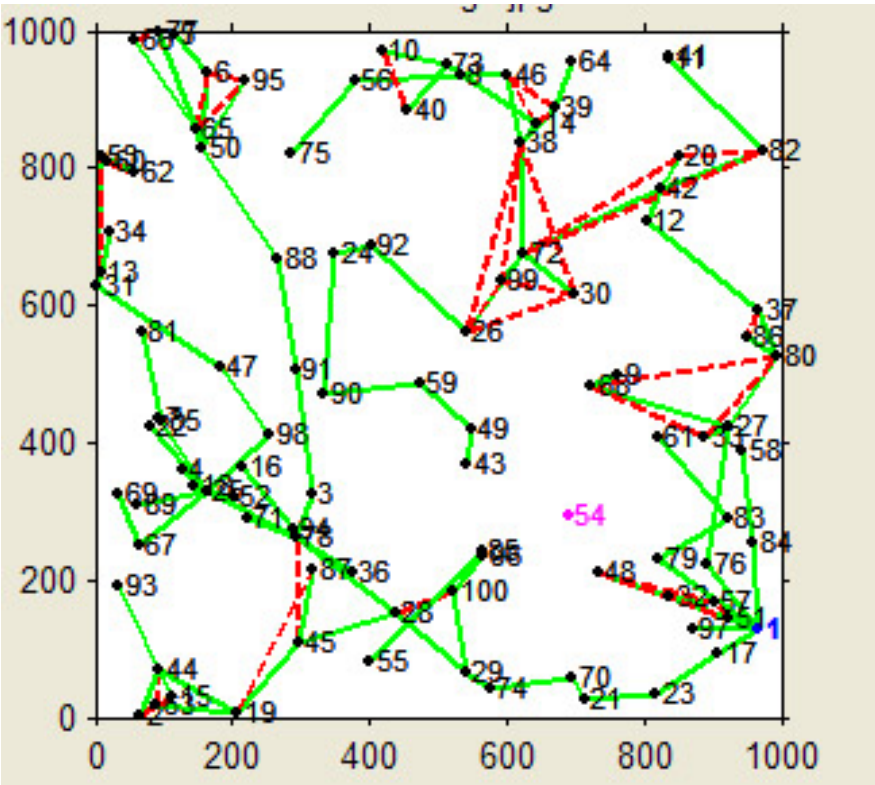

Fig. 4. 8 HTAP snapshot

4.5 Powerless (dead) nodes

Special care should be taken in the HTAC algorithm concerning the nodes which their battery is exhausted. These nodes are causing major problems to the network, especially when they are source nodes.

Thus, when a node is going to lose its power, it should immediately extracted from the network and the tables of their 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.

This algorithm deals with two cases. The first case is when the "dead" node is the source node and the other case is when the "dead" node is a child node.

In the first case, when the remaining source's node power is diminished, the source node broadcast an *elect_packet* to its neighbor nodes. The neighbor nodes communicate their power levels with each other and the one with the most remaining energy is elected as the new node. Power is diminished, and the other nodes remove it from their neighbor tables.

Powerless Nodes Handle Algorithm

```
if current_node is source node (current_level=1) and node_power allows 2 more Tx
// Power is going to get exhausted
broadcast elect_packet with current_node_id
    if current node receives elect_packet
        broadcast elect_packet_power with current_node_id
            if elect_packet_power>=current_node_power
            set current_node to root node
else if current_node is child node (current_level!=1) and node_power allows 2 more Tx
    broadcast remove_packet with current_node_id
```

Chapter 5

Simulation Results and Analysis

To evaluate the proposed algorithm, scenarios were created to compare various network and nodes parameters. The proposed algorithm has been implemented on Matlab, as a part of a specific simulator for Congestion Control in WSN. The source code for the Matlab simulation is described in Appendix A

5.1 Simulation Environment

In all simulation environments and scenarios we choose randomly to deploy nodes in a rectangular grid. We select two grid sizes 1000x 1000m and 500x 500m. This is a common choice for size grid in literature. We use the two different sizes to illustrate a denser node placement (for the same total number of nodes in the grid).

Furthermore in order to trigger off a congestion situation, all the nodes in the network are almost congested, meaning that their buffer occupancy in near to 90%.

On the other hand we introduce **more than one sink**. Each sink is able to handle a big amount of data.

Each time the number of nodes in the network is increasing the sources produces proportional number of data packets.

5.2 Scenarios Analysis and results

The first series of simulations that we conducted was in a grid of 1000m x 1000m. In each run, the parameters below were kept stable we altering the number of nodes in the grid.

X distance (m)	1000
Y distance (m)	1000
Transmit Power (dBm)	-2
Sensitivity Threshold (dBm)	-81
Path Loss Coefficient	3.5
Node CPU (MHz)	4
Radio Freq. (MHz)	433
Buffer Occupancy (%)	85
Data packet	1 packet
Control Packet	¼ packet

In order to get some reference results, the first simulation series run has been conducted with **no congestion control** algorithm. Basically, in this case the nodes were placed in a hierarchical tree (flooding algorithm) but no congestion prevention measures were taken when congestion happened (e.g no retransmissions). The only "extra" provision was the handling of "dead" (powerless) nodes.

The first parameter that has been investigated is the networks' energy consumption during a crisis state. As it was stated in Chapter 3, during a crisis state there is a sudden traffic increase which will certainly lead to congestion if no countermeasures are taken.

This specific metric, simply adds the remaining power of all nodes in the network and divides it by the number of nodes.

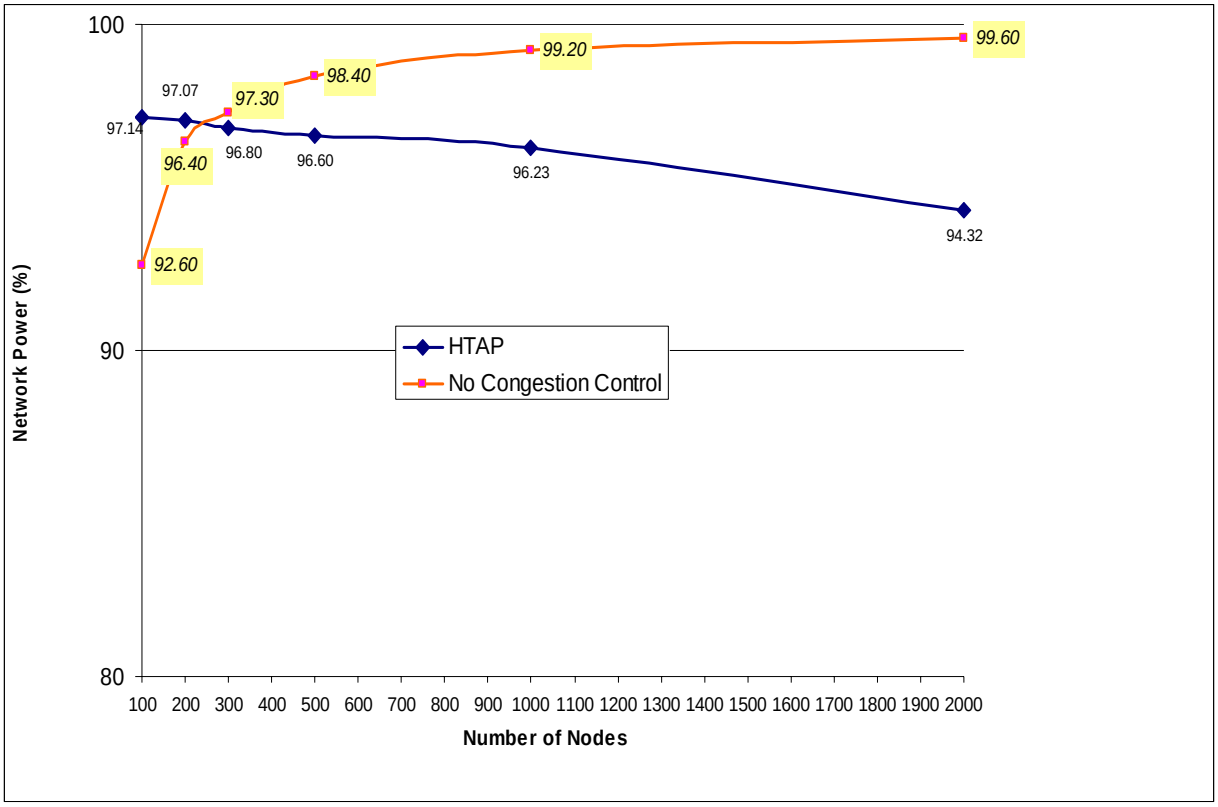

Fig. 5. 1 Network power versus Number of Nodes

At first glance this graph seems to give some "unexpected" results. As it is depicted here, as long as the number of nodes in the network is increasing, "no congestion control" algorithm appears to save more nodes' power. Of course, this is not true. The reason can easily be explained.

HTAP's main target is the involvement in congestion alleviation phase, a number of nodes which are in dormant state. These nodes will create alternative paths to the sinks. Bearing in mind that the network's dimensions are constant and the numbers of nodes are increasing, the more nodes are in the network, the more of them are involved in the congestion alleviation phase. This means that packets traverse to the sink through a higher number of hops, leading to energy consumption from many nodes.

On the other hand, when the "no congestion control" algorithm applies, the situation is clear. In this procedure is involved a specific number of nodes (only a path from the source to the sink). These nodes are, like all the nodes in the network, are almost congested. As long as they receive packets that are not able to handle, they drop them. Furthermore, the control

packets that they exchange are limited. This situation will lead to extinction of a specific number of nodes ,while the rest will have their power in full.

This leads to a subjective energy saving. Taking into account that the metric we examine, derives from the division of the remaining nodes' power by the number of nodes, it is easy to understand that as the number of nodes in the network is increasing, more nodes are in dormant state and of course more nodes have their power untouched, leading to increment of the overall power.

The next parameter that was studied was the number of packet drops during a crisis state. Keeping the parameters same as above, the following results were noted:

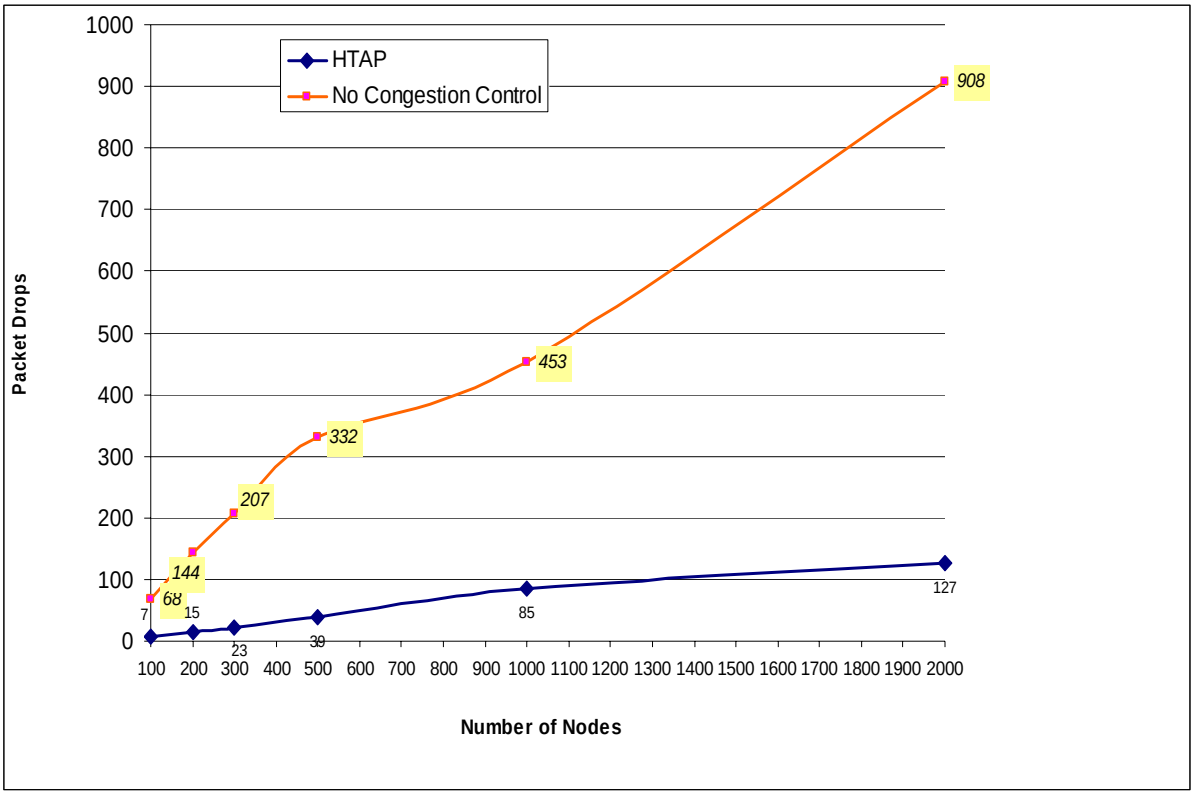

Fig. 5. 2 Packet Drops versus Number of Nodes

As it is presented in the graph, the number of packets drops with HTAP, increase slightly as the number of nodes is rising. Yet, in all cases packet losses, are much fewer compared with "no congestion control" algorithm. This fact is a good indication that HTAP algorithm is able to minimize, with good results, the congestion problem and is able to "save" a big number of data packets, from drop.

The problem concerning the "no congestion control" algorithm can become even worse, bearing in mind that in this simulation series, retransmissions have been disabled, and the "dead" nodes provision is active.

The next metric we study is the percentage of dropped packets, compared to transmitted packets.

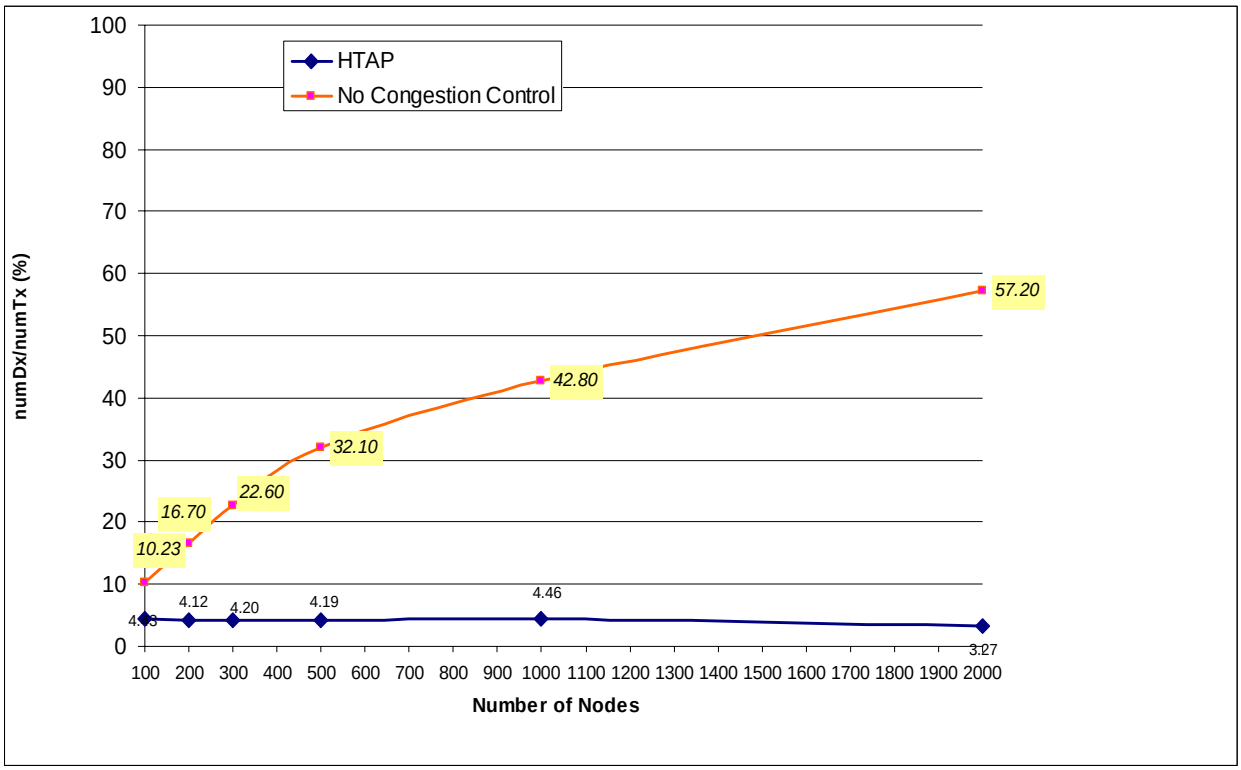

Fig. 5. 3 numDx/numTx (%) versus Number of Nodes

As it is depicted here the percentage of dropped packets to transmitted packets is generally stable meaning that HTAP alleviates congestion even under a big number of nodes. This algorithm is able to react efficiently even when the numbers of nodes and data packets are increasing in the network. On the other hand, if "no congestion control" algorithm is applied the percentage of lost packets is increasing. The situation is getting worse, especially when the number of nodes in the network is increasing leading the simulation to the creation of more data packets

The next parameter that has been simulated, was the mean travel time of packets from the source to the sink.

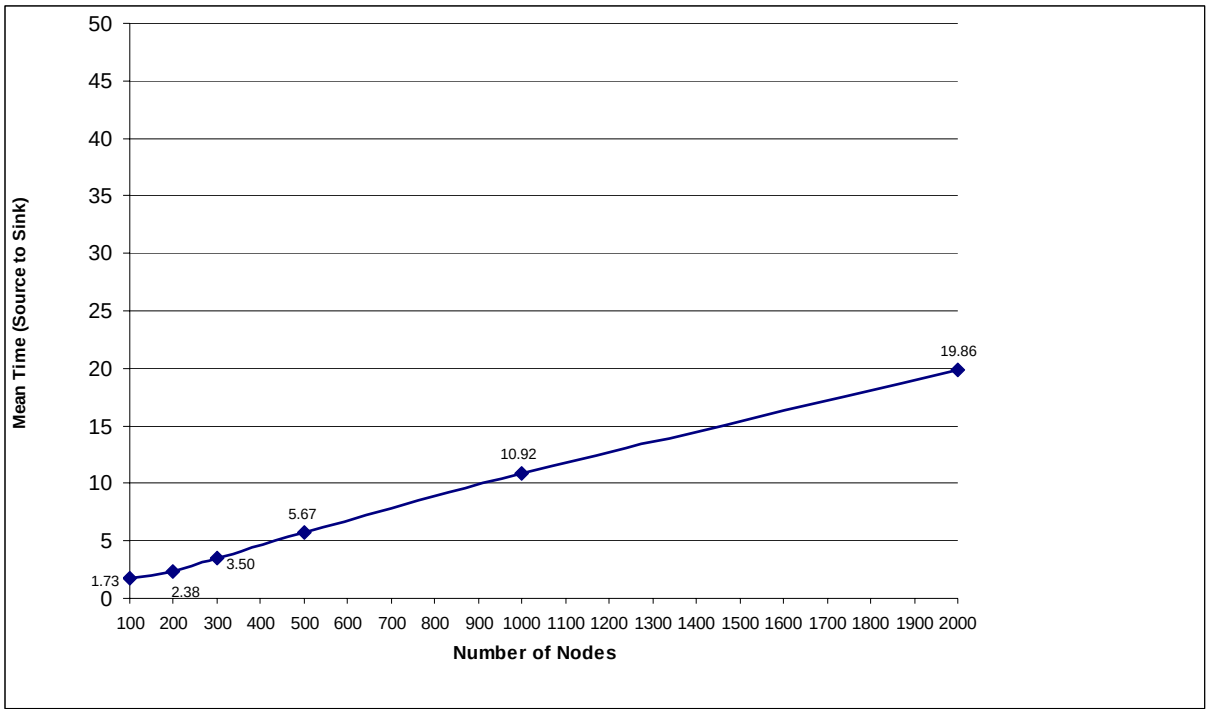

Fig. 5. 4 Mean Time (Source to Sink) versus Number of Nodes

This parameter has not been compared with "no congestion algorithm", due to the fact that the results of "no congestion control" algorithm are not reliable for this parameter. The reason is that this metric measures the time of all received packets to the sink and calculated the mean time. Bearing in mind that a big number of packets with "no congestion control" algorithm are dropped, the results could not be true.

Basically, this metric shows the trend of mean time while the number of nodes in the network is increasing.

As it is depicted here, the time is slightly increasing as the number of nodes in the network is increasing. This is normal having in mind that the creation of alternative paths creates many hops between the source and the sink. **This can be considered as a disadvantage of the proposed algorithm** since the time is sometimes critical factor for the success of the application.

Second series of simulations:

Keeping all the previous parameters the same, we narrowed the grid to 500m x 500m. Our target was to check how efficient is the algorithm when the grid is relatively small, compared to the amount of nodes that exist in the network.

The first parameter that was checked, is the packet drops.

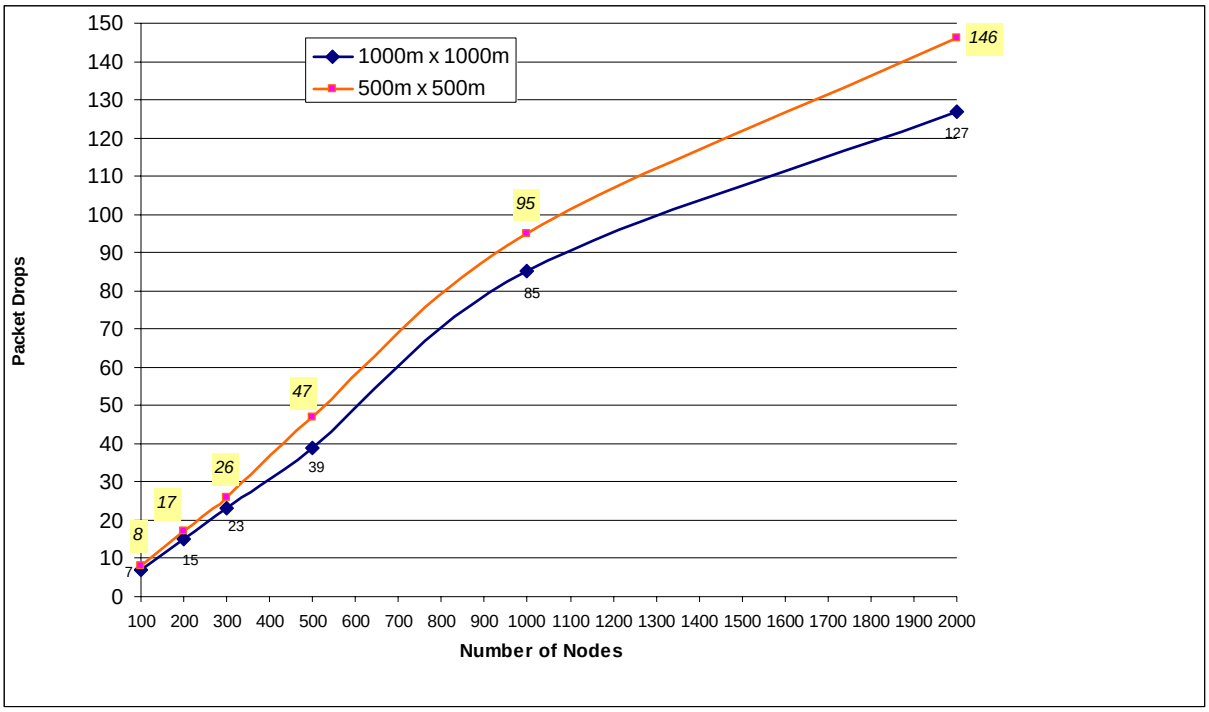

Fig. 5. 5 Packet Drops versus Number of Nodes (500m x 500m Grid)

The packet drops are slightly increasing compared to 1000x1000m grid. This fact is normal, taking into account that keeping all the parameters the same and especially in this case the transmition power, the neighbor table of each transmitting node is growing. This fact leads to bigger number of control packets, and normally, to more packet drops.

Moreover, although the path from the source to the sink is now smaller, the fact that more nodes are slotted in this path, leads to more packet transmissions which can lead to small increment in packet drops.

This conclusion is reinforced from the following graph.

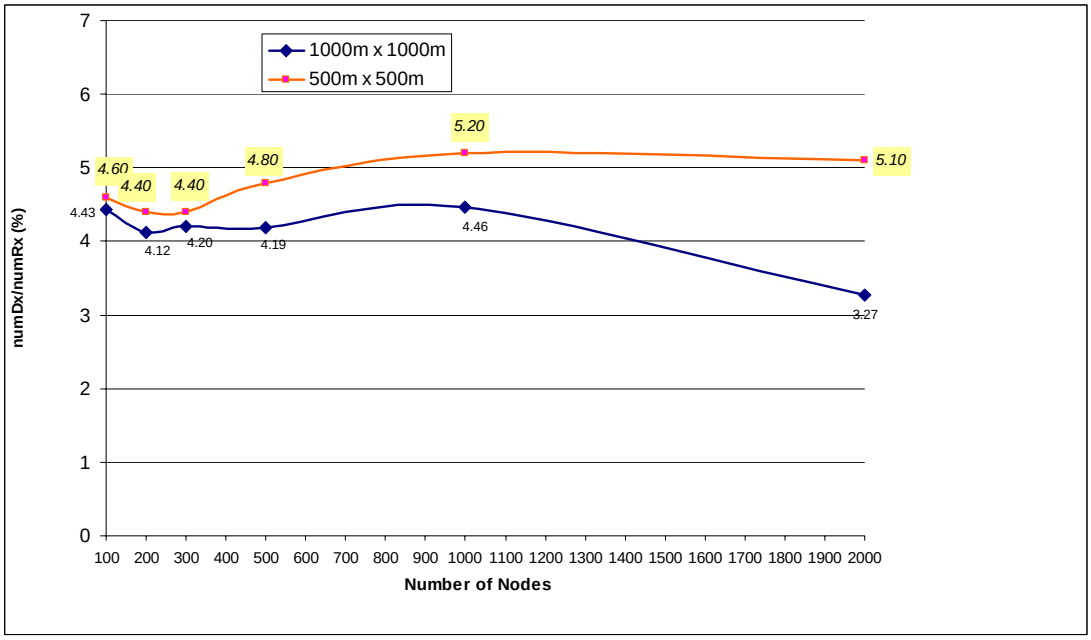

Fig. 5. 6 numDx/numRx (%) versus Number of Nodes (500m x 500m Grid)

As it is shown here, the ratio of transmitted to received packets is generally stable, but a bit above the 1000x1000m grid. This fact is trivial, bearing in mind that the number of control packets is a bit more when the network is densely deployed.

Finally we check the mean time for the transmission of packets between the source and the sink. The following graph depicts the results:

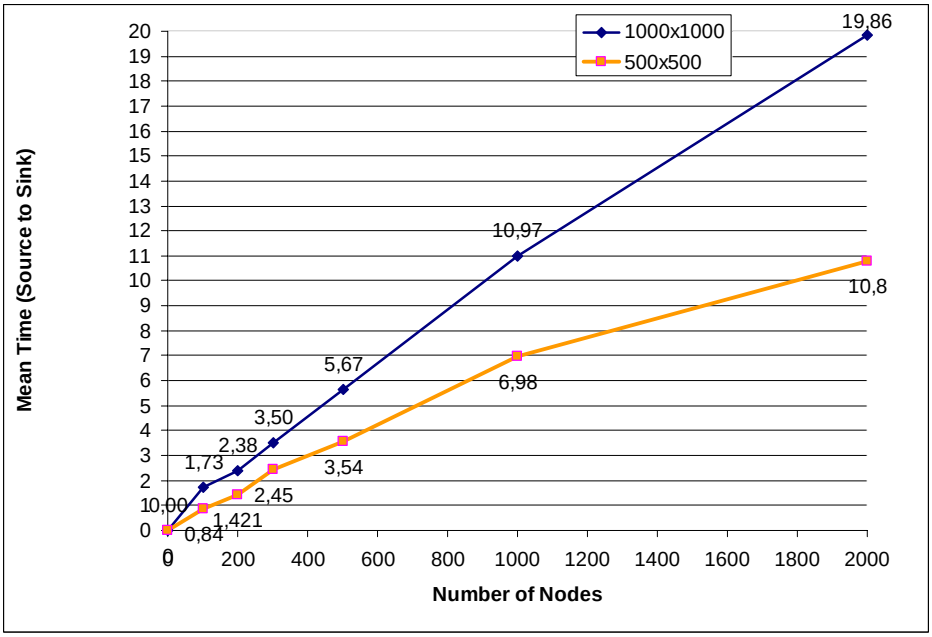

Fig. 5. 7 Mean Time (Source to Sink) versus Number of Nodes (500m x 500m Grid)

As it is presented in Fig 5.9 the mean time for the packets to travel between the source and the sink is much less, compared to the time that is needed when the grid's dimensions are 1000x1000m. This fact is expected, taking into account the fact that the grid is smaller and the propagation delay is not so much, as in 1000x1000m grid. Although the nodes that are slotted in the path from the source to sink are more compared to 1000 x 1000m grid, the whole propagation time for the transmission of data packets from the source to the sink, is still less.

Third series of simulations:

Finally, the following simulations run, in order to check the reaction of HTAP algorithm when the nodes' transmission power is changing. To achieve this task everything was kept stable except of the nodes' power transmission. More specifically, the simulations run in a 500x500m grid, having 300 nodes.

The first parameter that has been examined, is the mean time for the transmission of packets from the source to the sink. As the transmission power of the nodes is increasing, more nodes are getting to its neighbor table and the source node in getting closer to the sink.

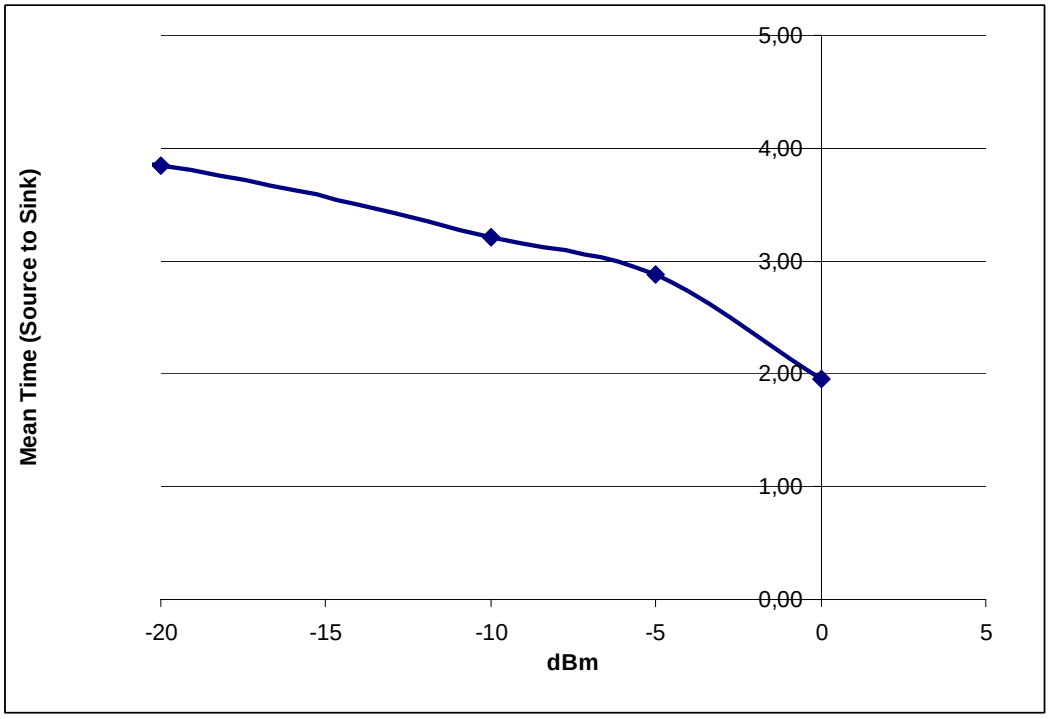

Fig. 5. 8 Mean Time (Source to Sink) versus Transmit Power

It is clear that HTAP functions normally, being able to transmit the packets faster to

the sink, as long as the nodes' power is increasing. The more the transmission power is, the less is the mean time for the traverse of data packets from the source to the sink.

The next parameter that has been investigated was the actual packets drops. The following graph states the results.

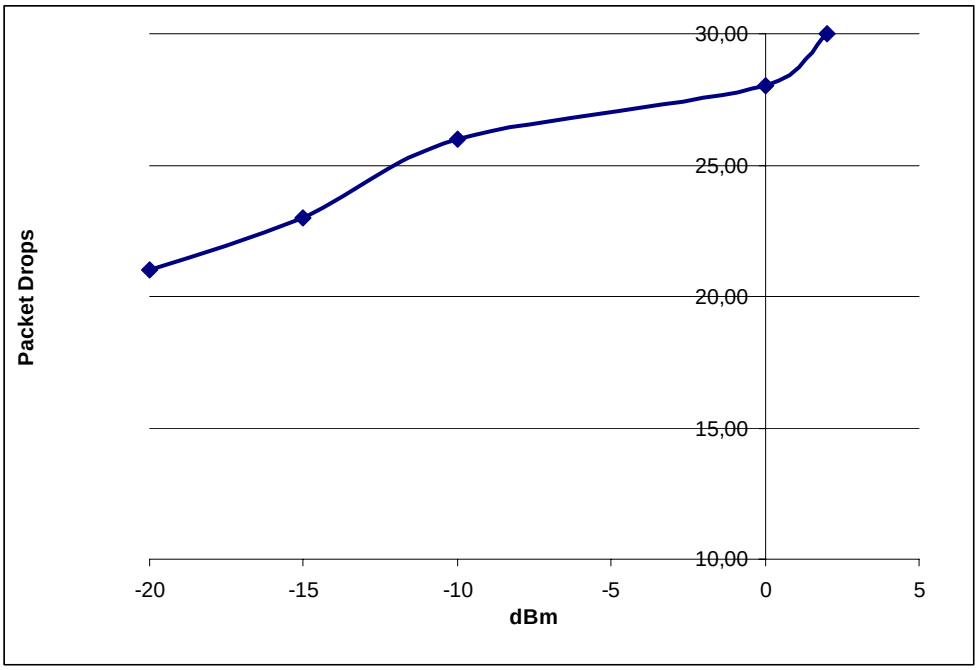

Fig. 5. 9 Packet Drops versus Transmit Power

As it is represented in this graph, there is a small increment in packet's drops when the networks power is increasing. This small increment results out of the fact that the neighbor table of the node is increasing and, normally, the control packets are increasing, too. This leads to more packets in the network and, normally, to a very slight increase in packets drops.

Normally in situations where the node's power is increasing, packets drops are expected due to interference also. In this specific simulator, interference has been disabled due to the fact that we focus on "buffer based" congestion. So packet drops due to interference are not added to graphs.

Chapter 6

Summary & Future Work

6.1 Summary

In this thesis we investigated the congestion problem that takes place in WSN. In our work we dealt with a specific category of WSNs, the event- based networks. As we discussed in our Thesis, these networks are producing data only when the monitored event exceeds a pre-specified threshold.

A typical example of event- based WSNs is the case of fire in a forest. Sensors are programmed to send data packets when the temperature in the forest exceeds a pre-specified value. In case of fire, sensor nodes which sense the event (fire) begin to transmit simultaneously a large amount of data packets. The target of this network is the continuous provision to sinks (i.e. local fire stations) with data packets, in order to keep them updated, for the fire's frontline. In this case, if the high load of data packets left unattended (no congestion control) the network becomes congested and the packets will not be forwarded to sinks. The result will be the failure of the network's mission.

The special features of event- based Wireless Sensor Networks have driven us to the need of development of a new algorithm. These features beyond the common (in WSNs) "power limitation" problem are the random placement and the need for transmission of all the data packets to the sinks. This last feature prohibits the use of data rate reduction mechanisms for the alleviation of congestion.

Studying the related works, which are presented in Chapter 2 we assert than none of them fully satisfies the conditions that are demanded in order for these networks to succeed in their mission. The most demanding condition as we also stated in the previous paragraph is the safe transmission of an event if a lot of data packets are produced during the crisis state. Crisis state is the network's state when the monitored event is taking place and leads to a great production of data packets. Almost all related works approach the congestion state by reducing the data rate which may lead in event loss.

In our approach we consider a special characteristic of WSNs, which according to our opinion has not attracted the interest that deserves in this specific problem. This is the plethora of sensor nodes which are in dormant state during even a congestion situation (i.e icrease resource provisioning).

The algorithm that we propose, HTAP (Hierarchical Tree Alternative Tree), takes advantage of these nodes, involving in the congestion alleviation phase their resources, power and storage (buffer space).

This algorithm consists of two similar algorithms, APC (Alternative Path Creation) and HTC (Hierarchical Tree Creaton). In the case of congestion these algorithms are using their neighbor tables to find alternative nodes in order to transmit their data. Their difference is basically the path they use. APC uses a completely random path compared to HTC which, as it name implies, uses a Hierarchical Tree path from the source to the sink. HTAP is a combination of these two algorithms which adopts the advantages of both algorithms and minimizes their disadvantages, as it is stated in Chapter 4 and Appendix B.

HTAP algorithm has been simulated in Matlab environment and has been compared to a situation without congestion control mechanism. The results showed that HTAP algorithm is able to reliably lead a large percentage of data packets to the sinks. In contrast the same network without congestion control mechanism, losses almost half of its data packets.

Moreover, simulations depicted that HTAP can also contribute to the increment of network's life since it uses power from a big number of nodes. This means that networks are dying out, uniformly. In other cases, due to congestion, it is possible that a specific area of the networks gets exhausted, while other areas are "untouched". Moreover a special provision has been taken in our algorithm in order that the entire exhausted nodes to get removed immediately from the network's topology, because they may cause major problems.

Finally, a problem that has also been addressed from our simulations is the fact that mean time for the transmission of packets form the source to the sink, is increasing while the

number of nodes in the network is increasing. This could become a “headache”, if the network is very densely deployed.

6.2 Future Work

Congestion Control in WSNs is a task which, in any case, is energy consuming. Having in mind the limited energy of these nodes, the task of even higher limitation of control packets remains.

Furthermore, this algorithm has been tested only in a simulation situation for limited scenarios. An extensive simulative evaluation and optimization of the algorithm and a subsequent test in real nodes is needed in order to prove its utility

Furthermore as future work the integration of HTAP as a congestion alleviation method to already proposed solutions like CODA [23] and “Congestion Control and Fairness for Many-to-One Routing” [25] to replace the data rate reductions that both papers are using can be considered.

Moreover, we believe that HTAP could become very efficient if the nodes are grouped in clusters and compare their data, before these are forwarded. In this case, there could be a significant limitation in the transmission of control packets, which is certainly leading in severe energy conservation.

Finally, we believe that the “recourse provisioning” technique is possible to significantly contribute in the solution of many open issues in WSNs. The plethora of cheap recourses is an advantage of these networks. These nodes is possible to work collaboratively, combining in a way there recourse and there data.

References

[1] Römer, Kay, Friedemann Mattern, "The Design Space of Wireless Sensor Networks". *IEEE Wireless Communications* **11** (6): 54-61, December 2004

[2] Thomas Haenselmann, "*Sensornetworks*". GFDL Wireless Sensor Network textbook, 2006

[3] Alberto Cerpa, Jeremy Elson, Michael Hamilton, Jerry Zhao, Deborah Estrin and Lewis Girod, "Habitat monitoring: application driver for wireless communications technology", *Workshop on Data communication in Latin America and the Caribbean, p.20-41, San Jose, Costa Rica,* April 2001

[4] E. Biagioni and K. Bridges, "The Applications of Remote Sensor Technology to Assist the Recovery of Rare and Endangered Species," *International Journal of High Performance Computing Applications, Special Issue on Distributed Sensor Networks,* April 2003

[5] L. Schwiebert, S. Gupta, and J. Weinmann, "*Research Challenges in Wireless Networks of Biomedical Sensors*," Mobile Computing and Networking, pp. 151–165, 2001.

[6] H. T. Kung and D. Vlah, "Efficient Location Tracking Using Sensor Networks," *in proc. of IEEE Wireless Communications and Networking Conference (WCNC)*, March 2003

[7] R. Brooks, P. Ramanathan, and A. Sayeed, "Distributed target classification and tracking in sensor networks," *Proceedings of the IEEE, vol. 91, no. 8, pp. 1163–1171,* 2003

[8] C. Chien, I. Elgorriaga, and C. McConaghy, "Low-power direct-sequence spread-spectrum modem architecture for distributed wireless sensor networks". *In Proc. Intl. Symp. on Low Power Electronics and Design (ISLPED), Huntington Beach, CA,* August 2001

[9] Cramer, R.J., Win, M. Z., and Scholtz, R. A., "Impulse radio multipath characteristics and diversity reception," *IEEE International Conference on Communications '98, vol. 3* pp. 1650-1654, 1998.

[10] Eugene Shih , Seong-Hwan Cho , Nathan Ickes , Rex Min , Amit Sinha , Alice Wang and Anantha Chandrakasan, "Physical layer driven protocol and algorithm design for energy-efficient wireless sensor networks" *Proceedings of the 7th annual international conference on Mobile computing and networking,* July 2001, *p.272-287, Rome, Italy*

[11] Alec Woo and David E. Culler, “A transmission control scheme for media access in sensor networks”, *Proceedings of the 7th annual international conference on Mobile computing and networking,* July 2001, *p.221-235, Rome, Italy*

[12] K. Sohrabi, J. Gao, V. Ailawadhi, and G. Pottie. “Protocols for self-organization of a wireless sensor network” *IEEE Personal Communications, pages 16--27,* October 2000

[13] T. He, J. A.Stankovic, C. Lu, and T. F. Abdelzaher, “SPEED: A Stateless Protocol for Real-Time Communication in Sensor Networks,” *in proc. of International Conference on Distributed Computing Systems (ICDCS’03)*, May 2003.

[14] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva, “Directed Diffusion for Wireless Sensor Networking,” *ACM/IEEE Transactions on Networking*, vol. 11, no. 1, pp. 2–16, February 2002

[15] W. R. Heinzelman, J. Kulik and H. Balakrishnan, “Adaptive Protocols for Information Dissemination in Wireless Sensor Networks”, *Proc. 5th ACM/IEEE Mobicom Conference, Seattle, WA,* August 1999

[16] S.M. Hedetniemi, S.T. Hedetniemi and A.L. Liestman, *A survey of gossiping and broadcasting in communication networks, Networks 18(4):319-359, 1988*.

[17] Shigang Chen and Na Yang, "Congestion Avoidance Based on Lightweight Buffer Management in Sensor Networks," *IEEE Trans. Parallel and Distributed Systems*, vol. 17, no. 9, 2006, pp. 934-946

[18] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, "A Survey on Sensor Networks", *IEEE Communications Magazine, Vol. 40, No. 8, pp. 102-114, August 2002*

[19] Jason Lester Hill, “System Architecture for Wireless Sensor Network,” PhD Dissertation, University of California, Berkeley, 2003.

[20] M. A. M. Vieira, C. N. Coelho, Jr., D. C. da Silva, Jr., and J. M. da Mata, “Survey on Wireless Sensor Network Devices,” Proc. IEEE Intl. Conf. Emerging Technologies and Factory Automation (ETFA ’03), pp. 537-544, 2003.

[21] J. Feng, *F. Koushanfar*, and M. Potkonjak, “System-Architectures for Sensor Networks: Issues, Alternatives, and Directions.” *ICCD, Invited to special session on Sensor Networks, September 2002.*

[22] Wei Ye and John Heidemann “Medium Access Control in Wireless Sensor Networks”, *Usc/Isi Technical Report Isi-Tr-580, October 2003.*

[23] J. Zhao, R. Govindan, and D. Estrin. “Computing Aggregates for Monitoring Wireless Sensor Networks”, *Proc. of First IEEE International Workshop on Sensor Network Protocols and Applications, May 2003*.

[24] C-Y. Wan, S. B. Eisenman, and A. T. Campbell. “CODA: Congestion detection and avoidance in sensor networks". *In Proc. ACM SenSys 2003, pages 266-279. Los Angeles, November 5-7 2003.*

[25] Cheng Tien Ee , Ruzena Bajcsy, Congestion control and fairness for many-to-one routing in sensor networks, *Proceedings of the 2nd international conference on Embedded networked sensor systems, November 03-05, 2004, Baltimore, MD, USA.*

[26] Bret Hull , Kyle Jamieson , Hari Balakrishnan, Mitigating congestion in wireless sensor networks, *Proceedings of the 2nd international conference on Embedded networked sensor systems, November 03-05, 2004, Baltimore, MD, USA*

[27] Y. Sankarasubramaniam, O. B. Akan, and I. F. Akyildiz, “ESRT: Event-to-Sink Reliable Transport in Wireless Sensor Networks” *Proceedings of ACM MobiHoc`03, Annapolis, Maryland, USA, June 2003, pp. 177-188.*

[28] Chenyang Lu , Brian M. Blum , Tarek F. Abdelzaher , John A. Stankovic , Tian He, RAP: A Real-Time Communication Architecture for Large-Scale Wireless Sensor Networks, *Proceedings of the Eighth IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS'02), p.55,* September 25-27, 2002.

[29] S. Kunniyur and R. Srikant “End-to-end congestion control schemes: Utility functions, random losses and ECN marks”. *IEEE/ACM Trans. on Networking, 11(5):689{702, October 2003*

[30] David B. Johnson “Routing in Ad Hoc Networks of Mobile Hosts” Proceedings of the *Workshop on Mobile Computing Systems and Applications, pp. 158-163, IEEE Computer Society*, Santa Cruz, CA, December 1994

[31] [J. Kang, B. Nath, Y. Zhang, S. Yu “Adaptive Resource Control Scheme to Alleviate Congestion in Sensor Networks” *Workshop on Broadband Advanced Sensor Networks, October 25-29, San Jose, California, USA*

[32]Alec Woo, David E. Culler, "A Transmission Control Scheme for Media Access in Sensor Networks", *Seventh Annual International Conference on Mobile Computing and Networking, pp 221-235, July 2001*.

[33] Yunli Yong, "A Congestion Control Scheme for Wireless Sensor Networks", *Master's Thesis, University of Texas*

[34] R.J. Gibbens, F.P. Kelly and P.B. Key "Dynamic Alternative Routing - modelling and behaviour", *Proc 12th International Teletraffic Congress, Turin. North-Holland, 1988.*

Appendix A

A.1 Matlab Source Code

In this Appendix we append the Matlab Source that we programmed in order to create this simulation environment and to evaluate our results. This simulation is specially constructed for congestion control, meaning that all nodes are almost congested.

Appendix B

B.1 Issues on proposed Algorithms

B.1 Issues on proposed Algorithms

In this Appendix we state the advantages and disadvantages of APC and HTC proving the reasons of our choice, to use these algorithms in combination and not solely.

The first parameter (and most important) that we check is packet drops. As it was reported in Chapter 4, both algorithms provide similar results concerning congestion control.

Fig. B.1 proves this statement.

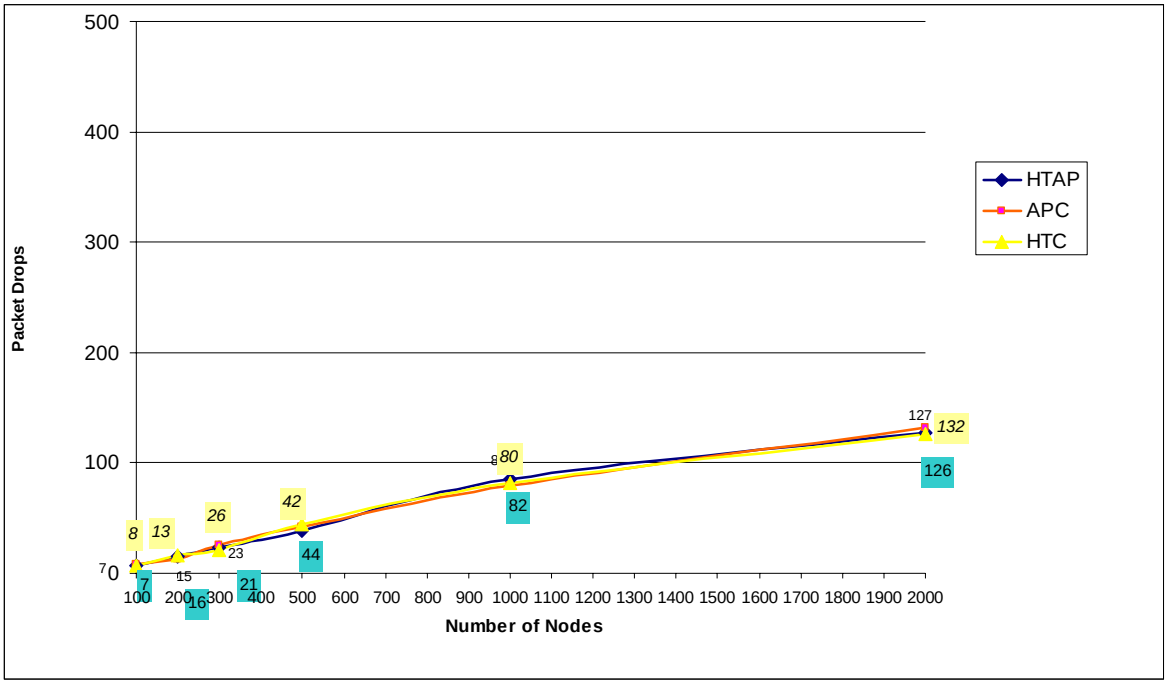

Fig B. 1

The reason we haven't used these algorithms solely is some performance problems that they present. These problems are examined below.

Let's start with APC. As we stated in Chapter 4, APC reacts with very good results in a congestion situation. But its major problem is that the mean time for the traverse of packets

from the source to the sink is highly increasing when there is increment to the number of nodes in the network. To prove this fact let's check fig B.2

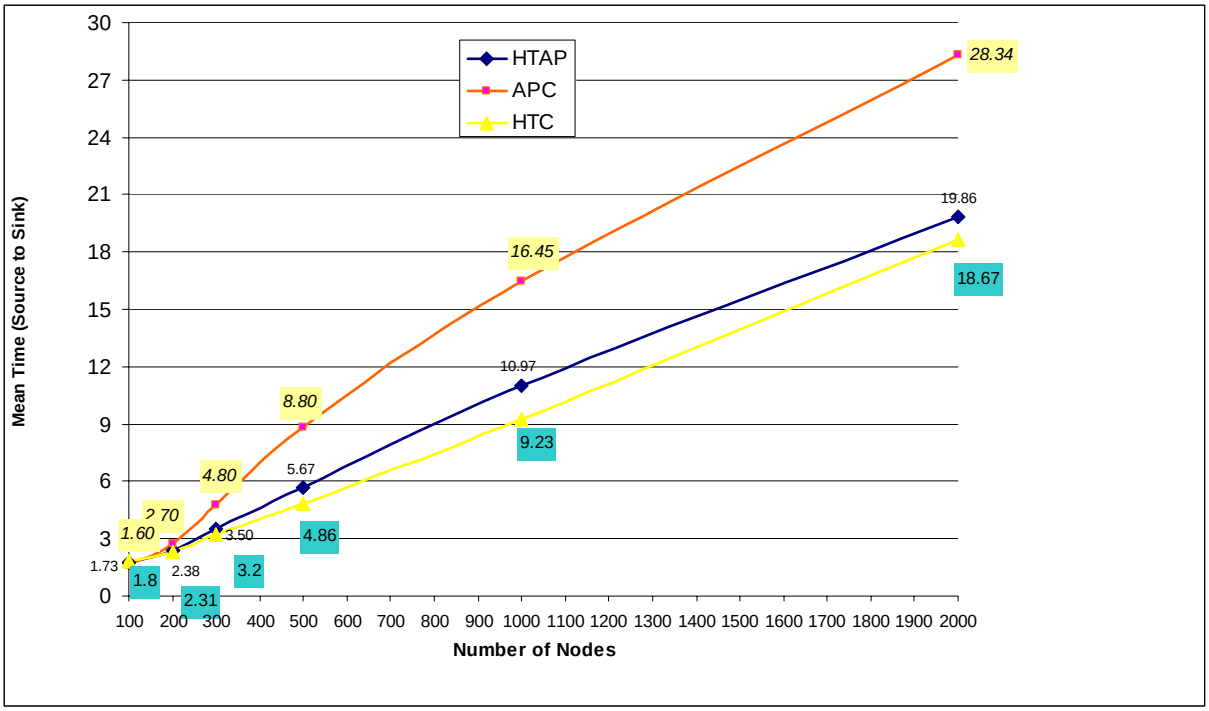

Fig B. 2

As it is depicted in Fig. B.2, the mean time from the source to the sink using APC is dramatically increasing when the number of nodes in the networks is increasing. This situation is not accepted in event- based networks, where time limits are strict.

If we compare the mean time in APC algorithm with the mean time in HTC we watch that HTC seems to be much more efficient compared with APC. This is true concerning the mean time.

This situation was accepted until we checked the remaining network power when HTC algorithm applies. The results are presented in fig B.3

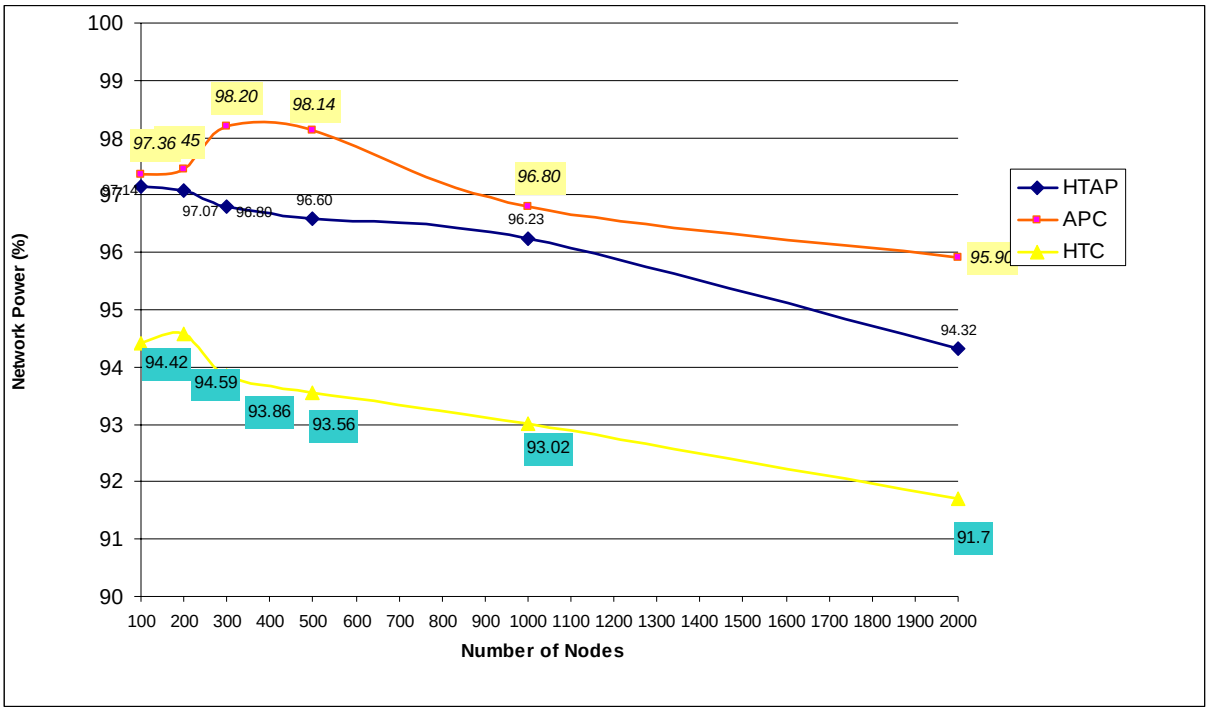

Fig B. 3

As it is shown in Fig. B.3 the networks power is decreasing in a relative high rate when the number of nodes in the network is increasing. Especially if we compare these results with APC algorithm the result is that energy consumption is much more with HTC. This happens due to fact the two way handshake that is taking place between the networks nodes, in order to form the paths from the source to the sink, leads to the transmission and reception of, comparatively to APC, big number of control packets.

Summing the "energy tax" of these control packets it leads to the conclusion that is much more, comparing to the possible "extra" transmissions using the APC.

So, taking into the account these inputs, we concluded that a compromise between these two algorithms could give the better results. This how HTAP came from.

The final problem that we face was **the threshold** in HTAP. The value of this threshold declares the maximum number of neighbor nodes, that a transmitting node (source) can have in its table in order to apply only APC algorithm. If the number is above this value then it applies HTC. According to the results that we extracted from our simulations, out of similar graphs like in fig. B.2 and B.3 **the threshold in our simulations is set to number 4**. With this number we get the best "compromise" results between the two algorithms.