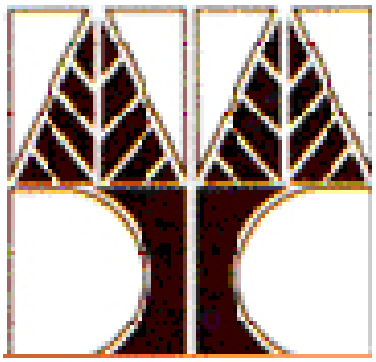
Master's THESIS

CONGESTION CONTROL IN WIRELESS SENSOR NETWORKS

Antonis Antoniou

UNIVERSITY OF CYPRUS



DEPARTMENT OF COMPUTER SCIENCE

May 2007

UNIVERSITY OF CYPRUS
DEPARTMENT OF COMPUTER SCIENCE

CONGESTION PROBLEM AND CONGESTION CONTROL IN WIRELESS SENSOR NETWORKS

Antonis Antoniou

Supervising Professor
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

May 2007

Abstract

In wireless sensor networks (WSN), 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 WSN.

In this thesis, we identify several network aspects, some of which are unique to sensor networks, that affect congestion, as for example: sensor reporting rate; number of MAC retransmissions; RTS/CTS mechanism; buffer size; average path length; ad-hoc routing protocols; and Minimum Contention Window. We also study congestion symptoms, such as delay and packet loss, for different Wireless Sensor Network Congestion Types. Moreover, since congestion affects on energy consumption and fairness, these are also examined.

A brief analysis of the existing congestion control schemes shows that most of them are effective on mitigating congestion mainly through rate control and packet drop mechanisms, but do so at the cost of significantly reducing application fidelity measured at the sinks. Also very few techniques appear to adopt the approach of resource provisioning (e.g waking up nodes and 'careful' routing). Although these may just use extra resources and reroute traffic or use multiple paths they may result in high energy consumption.

Based on the above research a hybrid theoretical framework of a congestion control scheme is proposed which is composed of two parts: (i) Throttle the traffic when applications' fidelity requirements are met and / or energy consumed is more than the available budget, and (ii) Increase the network resources that appear in redundancy in sensor networks using multi-path routing, or inserting more sinks, or controlling the number of sources sensing an event.

The resource control part of the scheme implements solutions that was previously tested with simulations in NS-2, where the results showed that such congestion control schemes can mitigate congestion and extend networks' lifetime when using network topology knowledge and turning off the extra resources as soon as the source traffic decreases, to save energy.

Acknowledgements

Firstly I wish to thank Computer Science Department of University of Cyprus for the chance given me to follow this MASTER. For the during of my stay I had the opportunity to associate with a number of truly remarkable people.

I would like to express my gratitude to my advisor Dr. Andreas Pitsillides for his support and encouragement throughout the duration of the project. I would like also to thank Dr. Vasos Vassiliou and again Dr Andreas Pitsillides for their support and help during the graduate courses Computer Networks and Internet , EPL 606 and Wireless Networks , EPL 657 respectively. These courses have encouraged me to follow this thesis.

I also want to thank my wife for her patience and support, particularly during the less exciting periods of my studies.

Mere words are insufficient to express my gratitude to my parents, for their love and incredible support, without which this would not have been possible.

Table of Contents

List of Figures

List of Tables

List of Acronyms

ACK	Acknowledgment
ADC	Analog to Digital Converter
AGT	Agent Trace Level
AIMD	Additive Increase multiplicative Decrease
AODV	Ad-Hoc On –demand Distance Vector
AQM	Active Queue Management
ARQ	Automatic Repeat Request
AWK	Interpreted Language written by Alfred Aho, Peter Weinberger, and Brian Kernighan in 1978
CBR	Constant Bit Rate
CODA	Congestion Detection and Avoidance
CPU	Central Processing Unit
CSMA	Carrier Sense Multiple Access
CSMA/CA	Carrier Sense Multiple Access With Collision Avoidance
CTS	Clear to Sent
CW	Contention Window
DCF	Distributed Coordination Function
DIFS	DCF Inter Frame Spacing
DLAR	Dynamic Load Aware Routing
DSDV	Destination-Sequenced Distance Vector
ESRT	Event –to-Sink Reliable Transport
IEEE	Institute of Electrical and Electronics Engineers
IFQ	Interface Queue
IP	Internet Protocol
LAN	Local Area Network
MAC	Medium Access Control
NACK	Negative ACK
NAM	Network Simulator Animator
NS	Network Simulator
PHY	Physical Layer
QoS	Quality of Service
RREP	Route Reply
RREQ	Route Request
RSSI	Received Signal Strength Indicator
RTS	Request to Send
SIFS	Short Inter Frame Spacing
SIR	Signal to Interference Ratio
SMAC	Sensor MAC
TCL	Tool Command Language
TCP	Transmission Control Protocol
TTL	Time to Live
TORA	Temporally Ordered Routing Algorithm
UDP	User Datagram Protocol
WSN	Wireless Sensor Network

Chapter 1

INTRODUCTION

1.1 Thesis target and motivation

1.1. Thesis target and motivation

In recent years, due to advances in low-power circuit and radio technologies, wireless sensor networks emerged and have received a lot of attention. A typical sensor network is formed by a large amount of nodes. Usually there is no pre-determined topology for a sensor network. Instead, these sensor nodes construct and dynamically maintain the structure of the network through wireless communication.

Sensor nodes have restricted power. They are usually equipped with batteries. In many cases, replenishment of the power resource is impossible. This nature imposes the requirement of energy-efficiency on all layers of protocols. Besides, sensor nodes are also constrained by relatively weaker processors and limited memory. Wireless sensor networks have a wide range of applications in habitat observation health monitoring, object tracking, battlefield sensing, etc. Intense study has been carried out in recent years on physical layer, MAC layer and network layer.

For applications where a sensor node reports sensed conditions of a region to one or a couple of sink nodes, sensor networks work with a light load most of the time. But, when an interesting event occur, such as enemy intrusion, the network will generate and need to transmit a sudden huge amount of data. In such cases, congestion control is of great importance. It can reduce the delay and save precious energy by regulating the transmitting rates, or by provisioning additional resources (e.g wake up nodes).

WSNs are different from traditional wireless networks and ad-hoc networks. In addition , WSN will explicit its own phenomena when congested. Existing protocols for wired networks like TCP are not suitable for WSN.

The problem of congestion (control and avoidance) in sensor networks remains largely open and not clearly described yet. When a sensor receives more data than it can forward, the “redundant” data has to be buffered. Then Congestion occurs because the limited buffer space becomes full and as a result the extra data (new or old) have to be dropped. This leads to both waste of communication and energy resources of the sensor nodes and also hampers the event detection reliability because of packet losses.

Hence, it is mandatory to address the congestion in the sensor field to prolong the network lifetime, and to provide the required quality of service (QoS) that WSN applications demand.

This M.Sc. Thesis aims to study the problem of congestion in WSNs and identify causes and symptoms strictly related with the philosophy and design of WSNs. The main objective is to examine the behavior of several network parameters and their impact to congestion in WSNs and address a theoretical framework for congestion control and avoidance that is different from existing traditional schemes that are based on rate control to alleviate congestion. Some other issues are also examined, such as energy consumption and fairness problems and their relation to congestion. The simulation scenarios were implemented and tested with the use of the NS-2 simulator.

The thesis is organized as follows. In Chapter 2 we give background information of congestion in computer Networks, and also describe pertinent WSNs features. In Chapter 3 we analyze congestion in WSNs and its design philosophy impact on congestion problem. In Chapter 4 we review the congestion control component mechanisms of existing schemes, introducing the problem statement and in Chapter 5 we present all the tests and evaluations with simulations in NS-2 .In section A cases of study, we examine congestion problem, causes and symptoms, and how several network

parameters affect congestion, according to Chapter 3 analysis. In section B cases of study, we evaluate solutions proposed for congestion alleviation of specific congestion types on wireless sensor networks. In Chapter 6 we propose a theoretical Framework for all the three WSNs congestion types, that is using either traffic control or resource control techniques to alleviate congestion and meet the application's fidelity requirements. This is based on previous tests and evaluations of Chapter 5.

Finally in Chapter 7 we present the contributions of the thesis and conclusions that evolve from the Thesis Chapters and some suggestions are introduced for future work.

Chapter 2

BRIEF REVIEW OF CONGESTION PROBLEM IN COMPUTER NETWORKS and WSNs

---

2.1. Congestion in Computer Networks.

2.1.1. What is the Congestion Problem

Congestion can be realized in many ways, but in simple terms one may say that, if, for any time interval, the total sum of demands on a source is more than its available capacity (Equation 2-1), the source is said to be congested for that interval. Mathematically speaking:

$$\sum Demand > Available\,\mathrm{Re}\,sources$$

Equation 2-1 Congestion Problem

In computer networks, there are a large number of resources such as buffers, link bandwidths, processor times, servers and so forth. If for a short interval, the buffer space available at the destination is less than that required for the arriving traffic, packet loss occurs. Similarly, if the total traffic wanting to enter the link is more than its bandwidth, the link is said to be congested.

When a number of packets dumped into the subnet by the hosts is within its carrying capacity, they are all delivered and the number delivered is proportional to the number sent. However as traffic increases beyond the network capacity, the routers are no longer able to cope and they begin losing packets. At very high traffic , performance collapses completely and almost no packets are delivered, due to dropped packets and retransmissions.

For wired networks like the INTERNET, there are mixed links with different bandwidths. The node with the lowest bandwidth along a path from the source to the destination is called the bottleneck. Usually, congestion occurs in the bottleneck since it receives more data than it is capable of sending out. In this situation, packets will be queued and sometimes get dropped. As a consequence, response time will increase and throughput will also degrade.

2.1.2. Congestion Costs

- Large queuing delays are experienced as the packet arrival rate nears the link capacity.
- The sender must perform retransmissions in order to compensate for dropped packets due to buffer overflow.
- Unneeded retransmissions by the sender in the face of large delays may cause a router to use its link bandwidth to forward unneeded copies of a packet.
- When a packet is dropped along the path to destination, the transmission capacity that was used at each of the upstream links to forward that packet to the point that it is dropped end up having been wasted.

2.1.3. What is Congestion Control?

Figures 2-1, 2-2 illustrate network performance as a function of the load. When the load is light, packets delivered is linearly proportional to the packets sent and delay is almost unchanged. After the load reaches the network capacity (the knee point), packets delivered won't increase much with the load. Instead, packets will be queued and the delay will become longer in this period. The throughput may suddenly drop (packet delivered are highly decreased) if packets get discarded due to buffer overflow, which is called the cliff point as shown in Figure 2-1.

Congestion control is necessary in avoiding congestion and/or improving performance after congestion. It aims to make the network operate around the knee point in Figure 2-1. Congestion control schemes are usually composed of three phases: **congestion detection, congestion feedback and congestion control**.

The criteria for congestion vary with protocols. Congestion can be determined

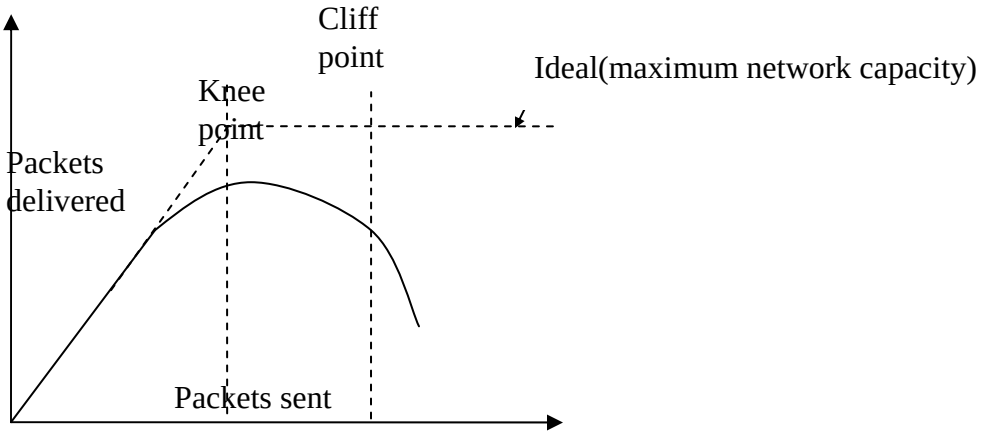


Figure 2-1 Packets Delivered vs Packets Sent

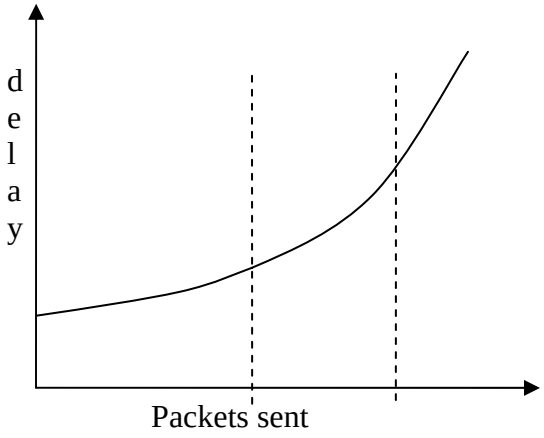


Figure 2-2 Delay vs Packets Sent

by checking queues length. It can also be indirectly detected by monitoring the trend of

packets delivered in time or delay, as depicted in Figure 2-2. In addition, packet loss can also be a criterion of congestion in wired networks. Practically, congestion detection can be processed in intermediate nodes or receivers.

Congestion feedback mechanisms can be categorized in several ways. In one way, they are classified into explicit or implicit feedback. Explicit feedback means that feedback is sent to the sender in an explicit form, like a dedicated bit. In implicit feedback, feedback information doesn't occupy any dedicated bits. The well-known example of implicit feedback is TCP, where 3-Duplicate-Acknowledgments implies congestion. From the aspect of information carried by feedback, they can be categorized into binary or non-binary feedback. Binary feedback can only tell if there is congestion or not. In contrast, non-binary feedback carries more information, which can indicate the congestion level. TCP is a binary feedback mechanism.

The rate control function (used in TCP) is usually viewed as a distributed decision-making problem. Additive increase and multiplicative decrease (AIMD) is proved to be a feasible linear control algorithm by **[1] ,** according to the criteria of efficiency, fairness and convergence. Many other methods are used for congestion control (e.g. **[2])**.

2.1.4. Congestion Prevention Policies:

Any architectural or implementation decision of computer networks, that affects either side of Equation 2-1, affects the design of a congestion control scheme. Thus any design decision affecting the load (demand) or resource allocation can be considered as part of overall congestion control strategy of the network. These decisions are also called policies.

In Table 2-1 we present different data link, network and transport policies that can affect congestion.

LAYER	POLICIES
Transport	• Retransmission Policy • Out of order caching policy • Acknowledgement policy • Flow control policy • Timeout determination
Network	• Packet queuing and service policy • Packet discard policy • Routing algorithm • Packet lifetime management
Data Link	• Retransmission Policy • Out of order caching policy • Acknowledgement policy • Flow control policy

Table 2-1 Network Architecture Design Policies affecting Congestion

Next, we will briefly analyze each layer policies, starting with the data link layer and going upward:

- The retransmissions policy is concerned with how fast a sender times out and what it transmits upon timeout. A sender that times out quickly and retransmits all outstanding packets using Go-Back-N will put a heavier load on the system than a slower sender that uses a Selective-Repeat. Closely related to this is the buffering policy. If receivers discard all out of order packets these packets will have to be transmitted again later, creating extra load. With respect to congestion control, Selective-Repeat is clearly better than Go-Back-N.

- Acknowledgement policy also affects congestion. If each packet is acknowledged immediately the acknowledgment packets generate extra traffic. However if acknowledgments are saved and piggybacked onto reverse traffic, extra timeouts and retransmissions may be reduced, however at the expense of slower response in case of congestion. Also a tight flow control scheme (e.g small window) reduces the data rate

and thus helps fight congestion, but at the expense of throughput. It is clear from above that various tradeoffs exist in the selection of the various parameters affecting the policies.

- At network layer a discard policy is the rule telling which packet to be dropped when there is no space in the queue. A good policy can help alleviate congestion control and a bad can make it worse.

- A good routing algorithm can help avoid congestion by spreading the traffic over all the lines. Finally a packet lifetime management deals with how long a packet may live before being discarded. If it is too long, lost packets may travel in the network for a lot of time and if is too short time out may occur more frequently before reaching their destination, thus causing retransmissions.

- In the transport layer, the same issues as in the data link layer occur but in addition, determining the timeout interval is harder because the transit time across the network is less predictable than the transit time over a wire between the two routers. If a time out interval is too short, extra packets will be resend unnecessarily. If is too long, congestion will be reduced but the response time will suffer whenever a packet is lost.

2.2. WSNs background information

In this section we will briefly review some important concepts in WSNs, which will allow us to better understand congestion in WSNs and are mainly used in simulations.

2.2.1. Carrier Sensing Multiple Access (CSMA)

In wireless networks, a community of nodes share a single transmission medium. To avoid collision and better utilize the bandwidth, some kind of medium access control (MAC) protocol is needed. Carrier sensing multiple access (CSMA) is a random access protocol, which allows users to transmit data in a none predetermined way.

CSMA schemes require a user to be sure the medium is idle before the

transmission. This is called carrier sensing. If the medium is busy, the user has to back-off for a random period and then re-sense. The random period is to minimize collision since other users may also want to take the medium at the same time. Once the channel is idle, the user can start transmission.

In sensor networks, CSMA schemes are practically used, for example, IEEE 802.11 **[3]** and SMAC **[4]**. We will discuss a little about IEEE 802.11 in the following content.

The distributed coordination function (DCF) of IEEE 802.11 is essentially a carrier sensing multiple access with collision avoidance (CSMA/CA) scheme. In addition to physical sensing, it also employs a technique called virtual carrier-sensing. Virtual sensing is realized by a pair of control frames request-to-send (RTS) and clear-to-send (CTS).

Figure 2-3 illustrates the mechanism of RTS/CTS. Node S 1 sends a RTS frame to node S 2 before the real data transmission. Node S 0 also receives the RTS and is blocked by it. Upon receiving the RTS, node S 2 broadcasts the CTS frame to its neighbors. Thus, node S 3 is also blocked. Node S 1 starts transmitting data once receiving the CTS frame from node S 2.

The RTS/CTS mechanism is to deal with hidden terminal problems. In Figure 2-3, node S 5 is a hidden node of the transmission from S 1 to S 2, since S 5 is beyond the

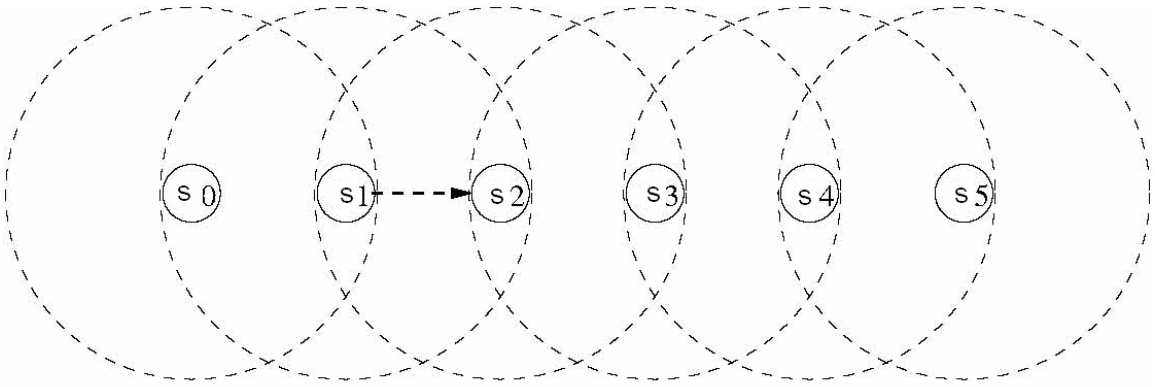


Figure 2-3 RTS/CTS Mechanism

interference range of S 2 (two hops). Node S 5 cannot sense the data flow from S 1 to S 2 and will think the medium is idle. If there is no RTS/CTS, node S 5 will directly start

sending data packets to S 4. In this case, the ACK frames from node S 4 will be very likely to collide with the data received by S 2. With the use of RTS/CTS, node S 5 won't get the CTS from S4 and cause interference to S 1 and S 2 since S 4 can detect the flow between S 1 and S 2.

The mechanism RTS/CTS introduces a lot of overhead especially when the data load is relatively low. Thus, sometimes, RTS/CTS is suggested to be disabled when IEEE 802.11 or its variant is used in sensor networks.

2.2.2. Multi-hop Wireless (CSMA Type) Network Capacity

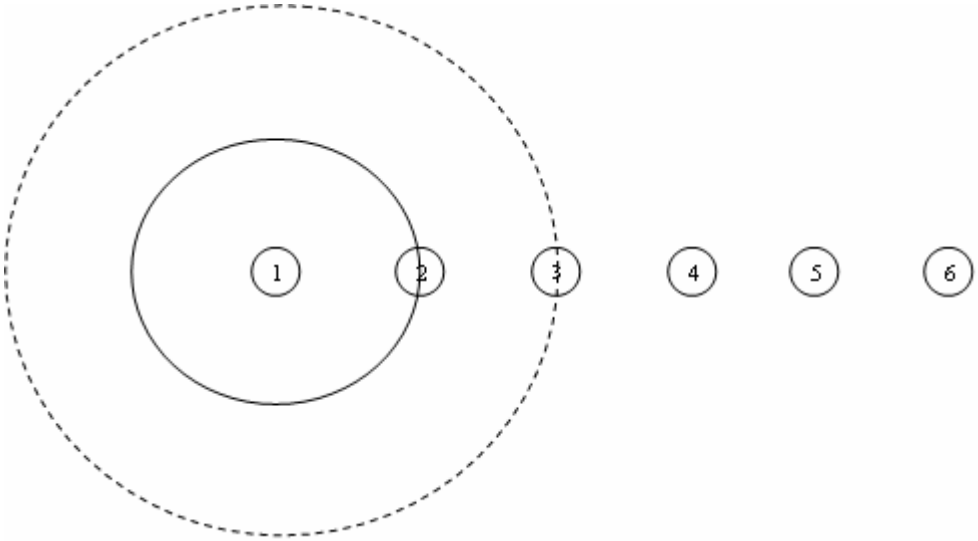


Figure 2-4 Multi-hop Wireless Network Capacity

Multi-hop wireless capacity is relevant for sensor networks as these networks show multi-hop behavior. Multi-hop capacity alone can create congestion. For example in Figure 2-4, suppose node1 is the source and node 6 is the sink of such a network. Assuming that the nodes that are node neighbors do not interfere with each other, when node 1 transmits, only node 4 and beyond can transmit simultaneously, because it is clear that when node 1 transmits, node 2 cannot transmit. Also node3 is blocked from node's 2 CTS, and if node 4 is transmitting then nodes 5, 6 are blocked by node 4. So an ideal MAC protocol could give a chain utilization of 1/3 **[5]** . This is giving a:

$$C=1/3x\ (packetsize)/(packetsize+RTSsize+CTSsize+ACKsize)xChannelCapacity$$

Equation 2-2 Chain Network Capacity

Which is about 0.425Mbps for a Channel Capacity of 2Mbps,and 1500,40,39,47 packet sizes for data packet, RTS,CTS and ACK packets respectively.

In case that interference range is up to 2 hops away, the capacity is even worst. In such a case in a 7 node chain node 3 experiences interference from 5 other nodes, while node 1 from only 3 nodes. This means that node 1 has better service rate than node 3 and can inject more traffic to the network that node 3 can handle. In such a case congestion occurs.

2.2.3. Ad-Hoc Routing Protocols AODV, DSDV

Ad–hoc Routing Protocols AODV and DSDV are mainly used for ad-hoc networks but also in wireless sensor networks. There philosophy and design could also contribute to congestion.

DSDV [[6]] is a hop-by-hop distance vector routing protocol. It is proactive; each network node maintains a routing table that contains the next-hop for, and number of hops to, all reachable destinations. Periodical broadcasts of routing updates attempt to keep the routing table completely updated at all times.

To guarantee loop-freedom DSDV uses a concept of sequence numbers to indicate the freshness of a route. A route *R* is considered more favorable than *R'* if *R* has a greater sequence number or, if the routes have the same sequence number, *R* has lower hop-count. The sequence number for a route is set by the destination node and increased by one for every new originating route advertisement. When a node along a path detects a broken route to a destination *D*, it advertises its route to D with an infinite hop-count and a sequence number increased by one.

Route loops can occur when incorrect routing information is present in the network after a change in the network topology, e.g., a broken link. In this context the use of sequence numbers adapts DSDV to a dynamic network topology such as in an ad-hoc network.

DSDV uses triggered route updates when the topology changes. The transmission of updates is delayed to introduce a damping effect when the topology is changing rapidly. This gives an additional adaptation of DSDV to ad-hoc networks.

AODV [7] **A**d-hoc **O**n-Demand **D**istance **V**ector routing is another popular routing algorithm for wireless ad hoc networks that operates using distance-vector routing mechanisms. It uses the concepts of Path Discovery and Maintenance. However, AODV builds routes between nodes on-demand i.e. only as needed. AODV does not depend on network-wide periodic advertisements of identification messages to other nodes in the network. It periodically sends "HELLO" messages in the local context of the system, to build up a set of neighbors. It then uses these neighbors in routing. Whenever any node needs to send a message to some node that is not its neighbor, the source node initiates a *Path Discovery*, by sending a *Route Request* (RREQ) message to its neighbors. Nodes receiving the RREQ update their information about the source. They also set up a backward link to the source in their routing tables. Each RREQ contains the source node's address (IP address) and a Broadcast ID that uniquely identifies it. It also has a current sequence number that determines the freshness of the message. Thus, a message number with a higher sequence number is considered to be *fresher* or more recent than that with a lower sequence number. The RREQ also contains a hop count variable that keeps track of the number of hops from the source. On receipt of the RREQ, the node checks whether it has already received the same RREQ earlier. If it has received the same RREQ earlier, it drops the RREQ. Otherwise, if it is an intermediate node without any record of a route to the final destination, the node increases the hop count and rebroadcasts the RREQ to its neighbors. If the node is the final destination, or an intermediate node that knows the route to the final destination, it sends back the *Route Reply* (RREP). This RREP is sent back via the same route traversing which the node had received the message from the source. As the RREP propagates back to the source node, the intermediate nodes setup forward pointers to the actual destination. When the source node receives the RREP, it checks whether it has an entry for the route. If it did not have any entry in its routing table, the node creates a new entry in the routing table. Otherwise it checks the sequence number of the RREP. If the RREP arrives with the same sequence number as in its tables but with a smaller hop count, or a greater sequence number (indicating fresher route), it updates its routing table and starts using this better route. Once an entry for the new route has been created in the table, the node can start

communication with the destination. Every time a node receives subsequent RREPs, it updates its routing table information, and only forwards those that are fresher or contain a smaller hop count. Each routing table entry contains information for the destination, the next node, number of hops to the destination, sequence number for that destination, active neighbors for the route and expiration time of the table entry. The expiration time frame is reset every time the source routes a packet to the destination.

2.2.4. Packet Losses Types I,II,III,IV In Wireless Multi-hop Networks.

In this section we list the various ways in which packets can be lost in a multi-hop wireless network. According to literature **[8]** are classified into four types:

Type I : If the transmitting mote is far from the receiving mote, the signal will attenuate significantly by the time it reaches the receiver. The signal attenuation is difficult to model since the radio signal strength is not uniform at the same distance from the mote in all directions.

Type II : If more than one sensor mote in the sensor network is transmitting simultaneously, interference will occur at the listening mote that is within range of the transmitting motes. In general, motes can be too far away to be considered neighbors, but still be close enough to interfere with reception. This type of interference is difficult to model also due to the same reason given for Type I loss.

Type III : The third cause of loss is due to self-interference, that is, a mote's transmission interferes with itself at the receiver, due to multi-path-effects, Rayleigh fading etc.

Type IV : The type of loss occurs when a packet is successfully received by a mote but has to be dropped due to queue overflow.

Of the causes listed, types I and III are dependent on the exact location and environment in which the motes are deployed, as well as the radio technology implemented in the motes. For instance, sensor motes placed less than 3 feet apart on a wall may not be able to hear each other due to reactions of the wall. We therefore cannot assume to have any control over these losses. Type IV losses are due to congestion within the network. Clearly, the correct implementation of congestion control will minimize this type of losses. In this thesis we focus on type II and IV losses.

2.2.5. WSNs Singularity Features

Wireless Sensor Networks **[[9] , [10]]** differ from conventional network systems in many aspects. WSNs usually involve a large number of spatially distributed, energy-constrained, self –configuring and self –aware nodes. Furthermore they tend to be autonomous and require a high degree of cooperation and adaptation to perform the desired coordinated tasks and networking functionalities.

Sensors technical characteristics since the time of writing this project are presented in Table 2-2. Also WSNs have many differences from other wireless ad hoc networks. Specifically number of sensor nodes are several orders of magnitude higher than nodes in ad hoc networks, sensor nodes are densely deployed, limited in power , computational capacities and memory, they have no global ID , they mainly use broadcast communication, they are prone to failures and network's topology may change very frequently.

Mote	WeC	rene	dot	mica	mica2	mica2 dot	iMote [26]	btNode[25]
Released	1999	2000	2001	2002	2003	2003	2003	2003
Processor	4 MHz				7 MHz	4 MHz	12 Mhz	7 Mhz
Flash (code, kB)	8	8	16	128	128	128	512	128
RAM (kB)	0.5	0.5	1	4	4	4	64	4
Radio (kBaud)	10	10	10	40	40	40	460	460
Radio Type	RFM				ChipCon	ChipCon	Zeevo BT	Ericson BT
μcontroller	Atmel						ARM	Atmel
Expandable	no	yes	no	yes	yes	yes	yes	yes

Table 2-2 Sensor Mote Technical Characteristics

Sensor Network Components

The main components of sensors consist of a sensing unit, a processing unit, a transceiver, and a power unit as shown in Figure 2-5, adopted from **[[9]]**. Each component is described below.

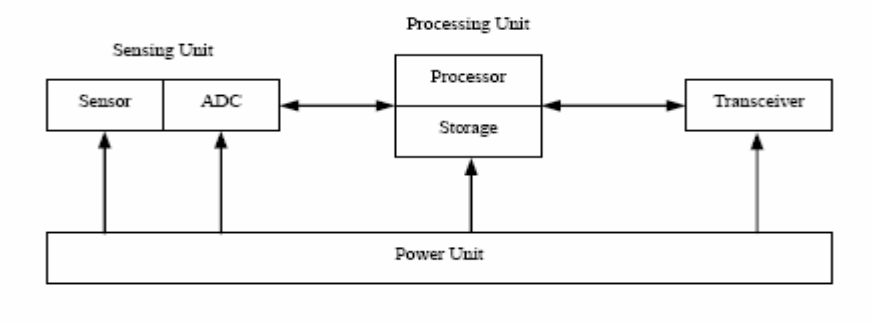


Figure 2-5 Sensor Main Components [[9]]

2.2.5.1. 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

2.2.5.2. Processing Unit

The processing unit plays 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 **[[11] , [12]]**. The processing unit needs storage for tasking and to minimize the size of transmitted messages by local processing and data aggregation **[[13]]**. Flash memory is widely used due to its cost and storage capacity.

2.2.5.3. Transceiver

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

2.2.5.4. 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.

2.2.5.5. Transport Protocol in Sensor Networks

Depending on the type of application, each sensor node may be required to perform some local (in - network) computations and data aggregation. Applying TCP to wireless sensor networks is expensive because of its three-way handshake mechanisms and packet header size. UDP is considered to be more suitable for sensors although it was designed to provide unreliable data transport.

Chapter 3

CONGESTION IN WIRELESS SENSOR NETWORKS

3.1. What is Congestion In WSNs.

WSN congestion occurs when offered traffic load exceeds available capacity at any point in the network.

In **[14 , 15]** the following analysis explain the details. Specifically suppose a phenomenon driven reporting model where a sensor reports if it is in range of the phenomenon. Assume that we have N sensors out of which M sensors are in range of the phenomenon at a given time T. Assume that the M sensors are in interference range with each other (e.g., the transmission range is greater than or equal to the sensing range). Of the M reporting sensors, each sensor S_i will transmit data toward the observer with bit rate $b(S_i)$. The total data in transit from time T to T +δ where δ is the average latency can be expressed as:

$$\text{Data} = \sum_{i=1}^{M} b(s_i)$$

Equation 3-1 Total Sensing Data in Transit

If this value reaches a certain fraction of the channel capacity, congestion will occur. If the Ctotal is the total channel capacity then :

$$\sum_{i=1}^{M} b(Si) <= aCtotal$$

Equation 3-2 Upper Bound of Reporting Rate

where α is a fraction of the capacity dictated by the self -interference that arises in multi-hop connections (α is typically around 0.25). **Thus, the upper bound on the reporting rate is dictated by the channel capacity**

The lower bound is application specific so application requirements are met.

$$Capplication <= \sum_{i=1}^{M} b(Si) <= aCtotal$$

Equation 3-3 Reporting Rate Bounds

Equation 3-3 is very important, and is often discussed during this Thesis.

3.2. Differences From Wired Networks Congestion

Congestion in wireless networks is different from that of wired networks. Due to the memory restrictions of the sensor nodes and limited capacity of shared wireless medium, network congestion may be experienced during the network operation.

Provisioning a wireless sensor network so that congestion is a rare event is extremely difficult. Sensor networks deliver myriad types of traffic, from simple periodic reports to unpredictable bursts of messages triggered by external events that are being sensed. Even under a known, periodic traffic pattern and a simple network topology, congestion occurs in wireless sensor networks because radio channels vary in time (often dramatically) and concurrent data transmissions over different radio links interact with each other, causing channel quality to depend not just on noise but also on traffic densities.

Moreover, the addition or removal of sensors, or a change in the report rate can cause previously un-congested parts of the network to become under-provisioned and congested. Last but not least, when sensed events cause bursts of messages, congestion becomes even more likely.

Furthermore the shared media access amongst sensors brings a new congestion scenario that is not present in a wired network. More specifically, CSMA's "fair" media

access directly contributes to buffer overflow. Refer to Figure 3-1, where data sources y, z, and w send packets to the sink via x. If y, z, w, and x each obtain a fair share of channel capacity, x will receive three packets for every packet it sends out, so packets will be dropped due to buffer overflows.

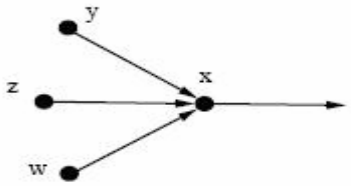


Figure 3-1 Fair Media Access –Congestion Contribution

Because radio links are not shielded from each other in the same way that wires or provisioned cellular wireless links are, traffic traversing any given part of the network has a deleterious impact on channel quality and loss rates in other parts of the network. Poor and time-varying channel quality, asymmetric communication channels, and hidden terminals all make even well-regulated traffic hard to deliver. In traditional wired networks and cellular wireless networks, buffer drops and increased delays are the symptoms of congestion. Over the past many years, researchers have developed a combination of end-to-end rate (window) adaptation and network-layer dropping or signaling techniques to ensure that such networks can operate without collapsing from congestion.

On the Internet, congestion control is done end-to-end at the transport LAYER, in transport protocols like TCP.

End-to-end flow control schemes like TCP are not well-suited to the domain of WSNs for the following reasons:

(1) Mismatch with applications that send at constant periodicity, with occasional bursts. As previously mentioned many sensor network applications involve low-rate CBR flows that might experience a sudden increase in transmission rate when an interesting event occurs. With TCP, every incoming ACK causes an increase in the transmission window size. The problem is that if the stream wasn't actually saturating the network (as in a low-rate CBR), this window inflation is artificial and does not signify that the

capacity indicated by the window is actually available! Now, when an event occurs that causes a sequence of packets to be sent in quick succession, TCP would assume that the large window was usable, but the result would be packet loss because the network isn't actually capable of sustaining this large window (rate).

(2) End-to-end acknowledgment overhead. Many end-to-end congestion control schemes require ACKs to be sent from the receiver, to allow the sender to obtain an accurate idea of the state of the network. Many sensor streams don't require the reliability semantics of TCP, making the cumulative ACKs unnecessary. Furthermore, since most sensor data packets are small, end-to-end ACKs would consume a substantial fraction of the overall network bandwidth.

(3) Bad performance when windows are small. Protocols like TCP are notorious for poor performance when windows are small. Although some recent solutions have been proposed for this problem, a fundamental problem is that small-window paths often tend to cause high packet loss rates in the way TCP adapts while probing for more bandwidth.

3.3. Sensors Networks General Design Philosophy and Impact on Congestion & Congestion Control Algorithms.

3.3.1. Clustering

A way of building hierarchy in sensor networks is by creating clusters. In this way a sensor node is having a special role in controlling others and in addition is a natural place of compressing traffic converging from many sensors. In that way traffic to the sink is reduced and congestion may be avoided**[[16]]**. On the other hand clustering may cause congestion due to interference of simultaneous transmissions.

3.3.2. In-network Processing (Aggregation)

When organizing a network in a distributed fashion, the nodes in the network are not only passing on packets but they are also actively involved in taking decisions about how to operate the network. Such a technique of in-network processing is aggregation. When the application needs to measure just the max, min, averages etc of a measurement

of the event then such aggregation function happens in sensor nodes and reduces the way the information is forwarded to the network.

3.3.3. Distributed Organization - Distributed protocols

Scalability and also robustness goal of sensor networks design make it imperative to organize network in a distributed fashion. That means that there should be no centralized entity in charge. The WSNs should cooperatively organize the network , using distributed algorithms and protocols. According to this design philosophy a congestion control algorithm should be designed in distributed way.

3.3.4. Reliability

Packet loss due to congestion can impair event detection at the sink even when enough information is sent out by the sources. Hence, congestion control is an important component for reliable event detection in WSN. Correlated data flows are loss tolerant to the extent that event features are reliably communicated to the sink. Due to this unique characteristic of WSN, required event detection accuracy may be attained even in the presence of packet loss due to network congestion. At any case the desired accuracy levels at the sink must be met. This is done by conservatively reducing the reporting rate so that both achieve accuracy requirements and avoid congestion.

3.3.5. Link Layer Reliability

If Automatic Repeat Request (ARQ) is implemented, the occurrence of congestion results in packets experiencing huge delays before reaching the base station. On the other hand congestion without ARQ implementation causes packets to be dropped along the route to the base station. Thus, a packet that originated a few hops away from the base station will have a smaller chance of reaching the base station than one that is generated by a node just a hop away. As a result, data may either experience too much latency to be of relevance, or the amount of data delivered will be insufficient for the application since too many packets may have been dropped along the way. To solve this problem an end to end congestion control similar to TCP is needed.

3.3.6. Energy

Congestion collapse has dire consequences for energy efficiency in sensor networks. When offered load increases past the point of congestion, fewer(useful) bits can be sent with the same amount of energy. The network wastes energy transmitting bits from the edge towards the sink, only to be dropped. This phenomenon is called livelock.

Furthermore any action taken to avoid or eliminate congestion must be taken considering energy limitations of sensor nodes.

3.3.7. Control Information dissemination

In general, the transmission and reception of additional control packets not only cause sensor motes to expend more energy, but are also likely to contribute to congestion. A technique used in some protocols, is the piggy-backing of control information on data packets, thus eliminating the overhead of additional packets. This implicitly assumes that there is no need for control information when no data is being sent, an assumption that is valid in our case since no data packets being sent implies that congestion is not present.

3.3.8. Application Awareness

Depending on the type of sensing application the rate of event impulses may be occasional or more frequent. Some applications may only generate light traffic from small regions of the sensor network (e.g., target detection) while others (e.g., fires, earthquakes) may generate large waves of impulses potentially across the whole sensing area which causes high loss.

From an application perspective, the value of information sensed by the sensor needs to be considered as well. If a sensor is providing some unique information about some feature of the phenomenon, then the application might require that sensor to report irrespective of the location of that sensor.

Thus, application level information must be used in determining what sensors to report and when to meet the application performance metrics. Congestion Problem may or may not occurs depending of the application specifics. Also congestion control mechanisms must take these application requirements into account.

3.3.9. Observer (Sink) distance from the phenomenon

The longer the average path from the sensors to sink the more data must be transmitted throughout the network increasing network load (decreasing network capacity)and causing congestion.

3.4. Types Of Congestion

We group them into two types according the reasoning of the problem appearance:

3.4.1. Type A

In a particular area, many motes within range of one another attempt to transmit simultaneously, resulting in type II losses and thereby reducing throughput of all motes in the area. This definition of congestion has been used in **[[8]]** as well. We note that explicit local synchronization among neighbouring motes can reduce type II loss in this regard, but cannot eliminate it completely because non-neighbouring motes can still interfere with transmission.

3.4.2. Type B:

Within a particular mote, the queue, or buffer used to hold packets to be transmitted, overflows. This is the conventional definition of congestion, widely used in wired networks. This is also the cause of type IV losses.

It is possible to have both types of congestion occurring at the same time.

Also there is another –different approach for naming congestion types. According to the place in the network the problem happens **[[17] , [18]]** and the kind of sensor reporting traffic, **three** of this type of congestion are known from literature.

3.4.3. HOTSPOT near the source(Transient) – Source Congestion

First, densely deployed sensors generating data events during crisis state will create persistent HOTSPOTS very close to the sources (e.g., within one or two hops) **[[17] , [19]]**. In this scenario, localized, fast time scale mechanisms capable of providing

backpressure from the points of congestion back to the sources would be effective. Also local de –synchronization of sources would be effective too.

3.4.4. HOTSPOT near the sink (Persistent) –Sink Congestion

Second, even sparsely deployed sensors that generates data even at low data rates create transient hotspots potentially anywhere in the sensor field but likely farther from the sources, near the sink **[17 , 19]**. Fast time scale resolution of localized hotspots using a combination of localized back-pressure and packet dropping techniques would be more effective, in this case. Source nodes may not be involved in the backpressure because of the transient nature of the problem in this situation. Also an effective way of alleviating sink congestion is to deploy multiple sinks that are uniformly scattered across the sensor field, and then balance the traffic between these sinks.

3.4.5. Forwarder Congestion -Sparsely deployed –High data rates

Third, a sensor network will have more than one flow (sink-source pair), and these flows will intersect with one another **[17 , 19]**. The area around the intersection will likely become a hot spot. In a tree-like communication paradigm, every intermediate node in the tree can suffer from forwarder congestion. Compared to the other two scenarios, Forwarder congestions are far more challenging because it is very difficult to predict the intersection points due to the network dynamics.

In this case even sparsely deployed sensors generating data will create both transient and persistent hotspots distributed throughout the sensor field. A combination of fast time scale actions to resolve localized transient hotspots, and closed loop rate regulation of all sources that contribute toward creating persistent hotspots seems to be effective. Resource provisioning techniques could be used when rate control methods cannot meet application's requirements.

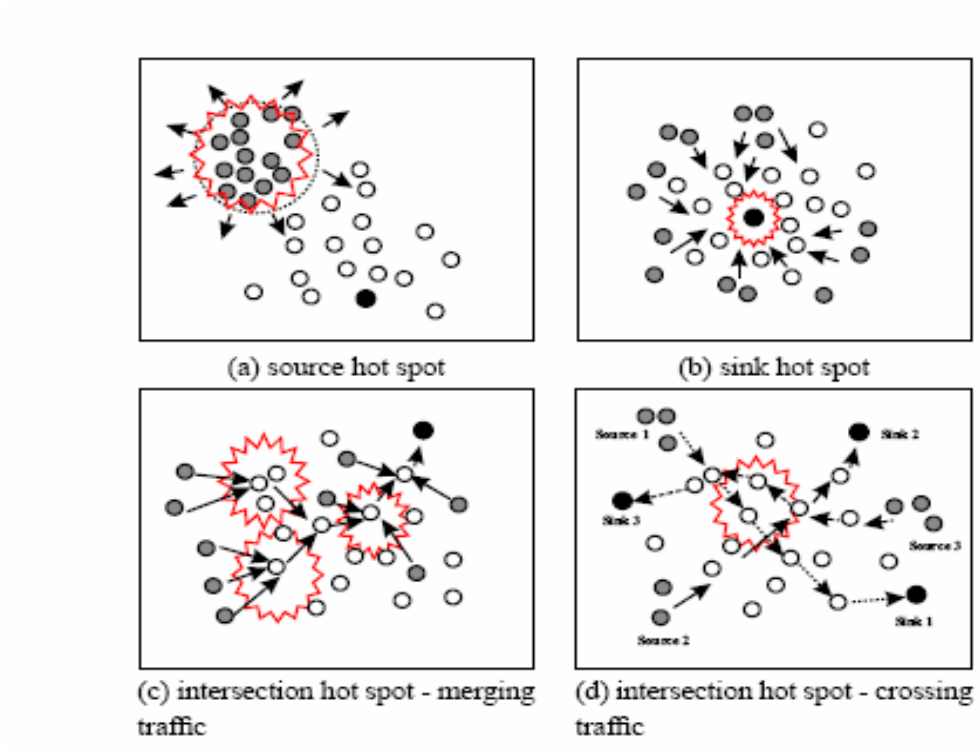


Figure 3-2 Sensor Network Congestion Types

3.5. Congestion Regions

Congestion period could be divided in 3 regions according to **[[20]]**. The region below *rate low threshold* , where the event reliability is relatively constant, is referred to as the *non-congested region*, since the buffer size of the nodes is enough to accommodate the traffic load. Beyond *rate low threshold* , a sharp transition phase is observed, which is referred to as *transition region*. This phase is where the network congestion builds up due to both traffic load increase and local contentions. Beyond a second threshold, *rate high threshold* , the reliability saturates which is referred to as *highly congested region*.

3.6. Main Causes Of Congestion Problem

3.6.1. Sensor Node Architecture Characteristics

3.6.1.1. Insufficient memory

In the occurrence of sudden traffic output queue will build up. If there is not sufficient memory to hold all of them, packets will be lost. Adding more memory may help at a point. In any case sensor nodes have very limited memory capabilities.

3.6.1.2. SLOW processors

Slow processors can also cause congestion. Sensors also behave as routers, and their CPU is slow at performing the bookkeeping tasks required from them (queuing buffers, updating table etc). This way queues can be build up, even when there is excess network capacity

3.6.1.3. Energy

Energy concerns can limit the available bandwidth, for instance, nodes periodically sleep to reduce energy consumption. Additionally congestion control capabilities [traffic redirection, more complex-effective algorithms] are limited from these energy concerns.

3.6.1.4. Wireless channel capacity

The larger the channel capacity the more traffic the sensor network is able to carry. Limited channel capacity is affected by interference, noise, and low power and modulation capabilities of the sensor. This is carried out from the well known SHANNON LAW:

$$C = B\log_2(1 + SIR)$$

Equation 3-4 Shannon Law

Because interference and noise are time – variable so channel capacity is variable too, and cannot be predicted. Better SIR can give more capacity when using strongest modulation schemes and higher transmission power. Spreading bandwidth using either Direct Sequence or Frequency Hopping techniques can also give more capacity.

3.6.2. Sensor network architecture characteristics

3.6.2.1. Many to One Nature

Due to the collaborative nature of the WSNs, the packet transmission about an event from multiple sensors to a single sink may create a bottleneck, especially around the sink. Hence, this many-to-one nature can also create congestion in the network.

In the unique funneling effect of WSNs events (e.g., periodic, discrete, and impulse traffic) generated under varying work-loads (e.g., light, moderate, high loads) move

quickly toward one or more sink points, as illustrated in Figure 3-3. This traffic convergence and subsequent build-up to a number of sensors near the sink can cause major congestion. A major limitation in the design of existing sensor networks is that they are ill-equipped to deal with the funneling of events and increasing traffic demands. This leads to increased transit traffic intensity, and may lead to congestion, and large packet loss.

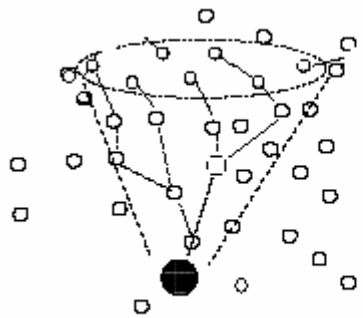


Figure 3-3 Funneling Effect on Congestion

3.6.2.2. Event- Driven Traffic

In sensor networks traffic patterns can be derived from the physical processes that they sense, which may result in extremely bursty, event-driven traffic.

Sensor networks typically operate under light load and then suddenly become active in response to a detected or monitored event. Depending on the application this can result in the generation of large, sudden, and correlated-synchronized impulses of data that must be delivered to a small number of sinks without significantly disrupting the performance (i.e., fidelity) of the sensing application.

3.6.2.3. Network Size-Density :Number of events and nodes density

Intuitively, for a WSN, increasing the number of sensors deployed in the field should result in a better performing network, otherwise, why pay the extra cost. In some cases large number of sensor nodes can be beneficial in several ways. Specifically, the accuracy of the sensing improve since there are more sensors in a position to report on the phenomena; the available energy within the network increases; and the additional

sensor density offers the potential for a better connected network with more efficient paths between the sensors and the observers. However, increasing the number of sensors in turn results in a higher number of sensors reporting their results per unit time. If this increased load exceeds the capacity of the network in terms of access to the shared wireless medium as well as congestion in intermediate nodes, increasing the number of active sensors may end up adversely affecting the performance of the network. The number of sensor networks comes with these trade offs, and must be appropriately designed in order not to causes congestion.

3.6.2.4. Packet Collisions

Increasing network contention causes an increase in packet collisions in the wireless medium. Based on the underlying medium access control (MAC) mechanism, after several unsuccessful retransmissions, these packets are dropped at the sender node. As a result, the decrease in buffer occupancy due to these drops may indicate lower congestion when only buffer level is used for congestion detection. Therefore, for accurate congestion detection in WSNs, a hybrid approach is required.

3.6.2.5. CSMA MAC Protocol-RTS/CTS use/blocking effect

From a network point of view, one of the primary reasons for using the *RTS*/*CTS* mechanism is to avoid network congestion resulting from frequent packet collisions.

The added cost of the RTS/CTS exchange is worthwhile when data packets are substantially larger than control packets. Under high-bursty traffic using RTS/CTS signaling can cause congestion .However, in sensor networks, data packets are usually small , and on some platforms the RTS/CTS exchange would incur a 40% overhead.

However, in the general setting of ad hoc-sensor networks, the current way of implementing the *RTS*/*CTS* mechanism gives rise to situations where a large number of nodes are unable to transmit any packet. These situations can lead to network-level congestion. Consequently, the throughput of the network goes to zero as the load increases. In IEEE 802.11 MAC, any node that receives an *RTS* packet is required to inhibit transmission. This rule is designed to ensure that the *ACK* packets can be received by the sender without any collision. However, due to this rule, a nearby node may get

falsely blocked, i.e. it may become prohibited from transmitting even if no other node is actually transmitting. Specifically, an *RTS* packet destined to a blocked node forces every other node that receives the *RTS* to inhibit transmission even though the blocked destination does not respond and thus no *DATA* packet transmission takes place. We call this problem the *false blocking problem*.

The simple blocking problem is localized to the neighbors of the blocked node and thus has a limited impact on the network performance. False blocking, however, may *propagate* through the network, i.e. one node may become false blocked due to a node that itself is false blocked. Therefore, false blocking may affect network performance severely. Due to false blocking, the throughput of the network goes to zero as the load of the network is increased beyond a certain value. Therefore, the *RTS*/*CTS* mechanism may congest a network instead of stabilizing it.

3.6.2.6. Reporting Rate

Mainly, WSN applications can be classified into two classes, i.e., event-driven and periodic . In event-driven applications, the reporting rate of sensor nodes may change during the lifetime of the network. Whereas, applications with periodic traffic, necessitate controlling the reporting rate for the proper operation of the network. In both cases, as a result of increased reporting rate, overall network congestion occurs even if local contention is minimized. Due to its collective and multi-hop nature, however, a collaborative approach is required in controlling flow rates in WSN.

3.6.2.7. Channel Contention and interference

In WSNs, the local channel contention in the shared communication medium causes overall network congestion. Channel contention may occur between different flows passing through the same vicinity and between different packets of the same flow. Consequently, due to channel contention, the outgoing channel capacity of a sensor node becomes time-variant. This time-variant nature makes the node's congestion level fluctuating and unpredictable even in case of constant incoming traffic rate. Moreover, high density of sensor nodes in the network topology exacerbates the impact of the channel contention.

3.6.2.8. Routing Protocols

A new technique designed for ad hoc networks is "on-demand," or reactive routing. Routing tables with full topological views are not maintained and only routes to nodes that a source needs to communicate with, are established on demand via source flooding. Existing on-demand routing protocols such as DSR (Dynamic Source Routing) **[[21]]** , AODV (Ad-hoc On-demand Distance Vector) **[[7]]** , and TORA (Temporally-Ordered Routing Algorithm) **[[22]]** use the shortest path as their routing criteria.

This route selection philosophy can lead to network congestion and long delays (because of congestion). Moreover, most on-demand protocols use caching mechanisms for intermediate nodes to "reply from cache," causing routing load to concentrate (Figure3-4) on certain nodes. Recent simulation studies **[[23]]** have shown that on-demand protocols that use shortest paths suffer from performance degradation as the network traffic increases. Also proactive ad-hoc routing protocol DSDV shows similar behavior.

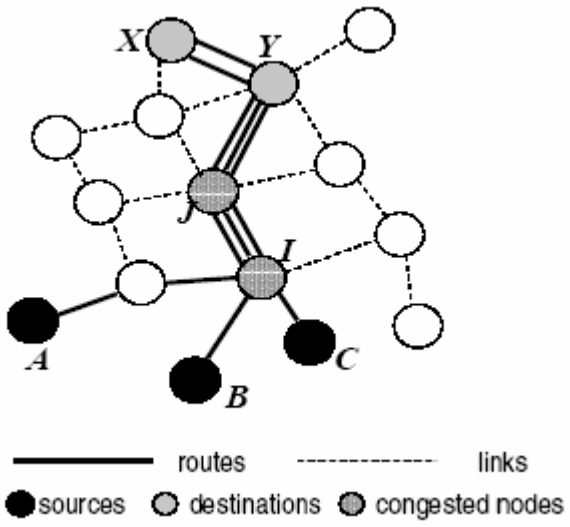


Figure 3-4 Routing Load Concentration

3.7. Congestion Key Symptoms

Some key symptoms of congestion collapse in wireless sensor networks are:

- Dramatic increase in loss rates.

This increase probably occurs due to MAC Layer Errors (Contention) and indeed due to buffer overflows.

- Starvation of most of the network due to traffic from nodes one hop away from the sink.

When the offered load increases, a decreasing number of nodes get a disproportionately large portion of bandwidth.

- Degradation on information fidelity measured on sink.

If information fidelity is below than expected and radio transmission errors are not many, the problem of this degradation mainly occurs due to congestion.

These are just observed symptoms, and a deeper investigation has to be done inside network parameters and characteristics to specify congestion detection indicators that will trigger congestion control. This is following in the next section.

Chapter 4

REVIEW OF THE STATE OF ART - CONGESTION CONTROL MECHANISMS

Congestion Control Component Mechanisms are composed from 3 sub-mechanisms. Congestion Detection which is the mechanism used to safely detect if the problem has occurred or is going to happen, the Feedback Mechanism, with which the sensor node or the sink gives a feedback to the network to take some action according to the problem and Control Scheme **,** which is the final action taken.

4.1. Congestion Detection

Accurate and efficient congestion detection plays an important role in congestion control of sensor networks.

To detect congestion, the level of congestion should be quantified to provide a fine-grained congestion control. We define two types of congestion levels in sensor networks as follows.

- per-node congestion level
- per-flow congestion level

The per-node congestion depicts the local congestion level each individual node perceives. To measure the per-node congestion level, each node can investigate the

statistics on several metrics such as queue length, packet drop rate, channel loading, etc. For example, **[24]** uses buffer utilization alone. However, if the wireless channel is unreliable either by the unreliable MAC protocol (e.g. when the maximum number of retransmissions is small) or by the unstable channel condition, the queue length or packet drop rate cannot be a metric for congestion detection. This is because even though a node's incoming traffic volume exceeds the outgoing channel capacity, the queue length remains small since packets quickly leave the queue due to collision and no subsequent retransmission after the collision. Therefore, in **[17]**, each node measures its congestion level based on the perceived channel loading as well as its queue length.

In general, the per-node congestion level measurement function f ( ) returns a positive real number indicating the congestion level based on the metrics of m1,m2, …,mn as follows.

Li = f(m1,m2, ….,mn)

where Li indicates the node i's congestion level.

A flow includes a source, a sink, and all the intermediate nodes between them. The data generated at a source traverse several intermediate nodes before they arrive at the sink. The delivered data volume to the sink is bottlenecked by the node whose per-node congestion level is the highest along the flow path. Therefore, the per-flow congestion level, denoted as Lflow, is defined to be the highest per-node congestion level of all the nodes in the flow and represented as follows.

Lflow = max(L1,L2, …Ln)

where Li indicates node is per-node congestion level. The per-flow congestion level can be calculated in a distributed manner by each node of a flow. The source records its current per-node congestion level in the header of a data packet destined for the sink. Each subsequent node compares its per-node congestion level with the per-flow congestion level recorded in the header of the data packet forwarded from the previous hop. If its per-node congestion level is greater than the per-flow congestion level, then the node updates the per-flow congestion level in the header of the data packet with its per-node congestion level and forwards it to the next hop. If its per-node congestion is less than the per-flow congestion level, the node simply forwards the packet without modifying the per-flow congestion level. Therefore, the highest per-node congestion level

is passed downstream towards the sink. The sink may enforce an end-to-end congestion control based on the per-flow congestion level. This approach could be used more efficiently in small wireless sensor networks. When sensor networks are composed of thousands of sensor nodes, the approach of per flow congestion is not well applicable as a lot of overhead would occur and frequent changes on congestion level would induce problems on accurate congestion detection.

Two main categories are figured in literature:

4.1.1. Sensor Node Initiating

It determines the local congestion levels and it performs better under high load traffic scenarios >=50% of channel load, but it performs as well in the lower traffic load region. In this category the following congestion indicators are used :

- Buffer Occupancy

Occupancy of Sensor Nodes Buffer: The simplest method is to compare the instantaneous buffer occupancy against some threshold value. If the threshold is exceeded the congestion state is diagnosed. This kind of detection is used from **[17, 25, 26,14 , 7, 28, 29, 30]** approaches. An improved method, like that used for **[24]**, takes the growth trend into account and eliminates the possibility of the threshold that make up for a large fraction of the buffer size, to detect congestion state too late. With this method the buffer size is regularly sampled and congestion is diagnosed when the instantaneous buffer level is above some threshold and additionally the buffer size has grown in the immediate past.

As shown from **[17]**buffer occupancy alone is not a reliable congestion indicator because packets can be lost in the channel due to collisions or hidden terminal situations and have no chance to reach a buffer. Only the situations of a full buffer and an empty buffer are reasonable indicators.

This is because without link ARQ, the clearing of the queue does not mean that congestion is alleviated since packets that leave the queue might fail to reach the next hop as a result of collisions. CSMA does not guarantee collision-free transmissions among neighboring nodes because of the detection delay.

However the buffer occupancy seems not to provide an accurate indication of congestion even when the link ARQ is enabled.

Following much of the prior work on congestion control, **[[31]]** uses an exponentially weighted moving average of the instantaneous queue length as a measure of congestion: *avgq* = (1_*wq*)*avgq* +*wq* _*instq*. The average queue length is updated whenever a packet is inserted into the queue. Thus, if *avgq* exceeds a certain upper threshold *U*, the node is said to be congested. The node remains in a congested state until *avgq* falls below a lower threshold *L*. In practice, a single threshold is too coarse-grained to effectively react to congestion.

- Channel Utilization .

The goal of channel sampling is to obtain an estimate of the current channel utilization U. For CSMA max utilization is Umax <1, beyond which the rate of collisions increases and goodput decreases. Congestion is diagnosed when the channel utilization is within some neighborhood Umax.

In CSMA networks, it is straightforward for sensors to listen to the channel, trace the channel busy time and calculate the local channel loading conditions.

Channel loading gives accurate information about how busy the surrounding network is, but it is inherently a local mitigation mechanism. It has limited effect, for example, in detecting large-scale congestion caused by data impulses from sparsely located sources that generate high-rate traffic. Listening to the channel consumes a significant portion of energy in a node. Therefore performing this operation all of the time is not practical in sensor networks. So sampling schemes activates local channel monitoring only [when queue is not empty] at the appropriate time to minimize the energy cost while forming an accurate estimate of conditions. This approach is used in **[[14]]**.

- A combination of methods a and b seems to work very well.

This kind of congestion detection is used in **[[17, 32, 33]]**.The above methods showed to work very well on detecting congestion. The following are also proposed but seem to have mixed results:

- Measuring the received signal strength **RSSI** on messages , to determine how busy the channel is from interference level. This is referenced as a possible congestion detection method in **[[33]]** .

- By counting the **continuous occupied queues** neighborhood queue) that may appear due to deadlock effects **[[34]]** because of CSMA MAC.

4.1.2. Post Facto Detection at the sink

According to this type of detection, congestion inference happens on the sink. This approach is mainly used in **[[32], [33]]** .

- Report rate/fidelity measurement on the sink

The sink as a point of data collection, can do smart monitoring of the event data quality and the measured application fidelity, and initiate control signals only when the measured application fidelity degrades below a certain threshold.

In **[[17]]** when a sink consistently receives a less than desired reporting rate, it can be inferred that packets are being dropped along the path, most probably due to congestion. In contrast, when in **[[24]]** receives less than the desired reporting rate it signals the sources to increase the source reporting rate unless a packet with the congestion notification bit set is received at the sink.

Such fidelity measurement approaches need to operate on a much longer time scale compared to the packet transmission time scale, and consider:

- End-to-end delay between sources and sink nodes since only the sink recognizes its own requirements on the sampling rate.

- Processing delay -a sink typically collects data from multiple sources regarding the same phenomena (e.g., data aggregation/fusion). To cope with the different delays of packets, which possibly travel along different paths from different sources, the sink needs to wait a minimum period of time to collect reports before concluding anything.

- Stability -to avoid unnecessary reaction to transient phenomena that could cause oscillation, a sink should not respond too quickly to events and therefore should define an appropriate “observation period” over a longer time scale.

In short, congestion detection based on the reporting rate is inherently slow and end-to-end in nature, and therefore, cannot cope well with transient hotspots in sensor networks.

4.2. Congestion Handling [Feedback And Control]

This is a 2-step mechanism:

- Congestion Feedback
- Congestion control (action)

Based on congestion control algorithms classification done in **[[35]]** and according to control theory point of view, we divided congestion control solutions into 2 groups. Open Loop and Closed Loop.

4.2.1. Open –Loop

These are the solutions in which control decisions of algorithms do not depend on any sort of feedback information from the congested spots in the network, and so they do not monitor the state of the network dynamically. The congestion control algorithm serves as a controller or control actuator, purely based on its own knowledge of local sensor node. They can be further classified as explicit control algorithms at the source or intermediate sensor nodes of the network. This is used by **[[32]]** where the congested node in the visibility of a Virtual Sink redirect the traffic accordingly when working in the node initiated congestion detection mode. Authors of **[[36]]** use open loop control as well. The aggregator checks for its congested local state and accordingly adjust the degree of that aggregation.

4.2.2. Closed Loop

In contrast, closed loop solutions are based on the concept of a feedback loop. This approach has three parts when applied to congestion control:

- Monitor the network to detect when and where congestion occurs.
- Pass this information to places where action can be taken.
- Adjust network operation to correct the problem.

The feedback can be either global or local; global means that the feedback information goes all the way from destination to source (end to end) where local means the feedback information comes only from immediate neighbors. With the provision of feedback these algorithms are able to monitor the network performance dynamically. The feedback may be explicit (sent explicitly as different or piggybacked messages) or implicit in other case. The explicit feedbacks can be further classified as persistent (if available all times) and responsive feedback (if only triggered under certain conditions). Furthermore many explicit closed loop algorithms they are anticipatory or reactive to congestion, whereas in the anticipatory stage such algorithms tend to achieve congestion avoidance whereas reactive strategy is a congestion recovery scheme that responds to conditions of network congestion.

4.3. Feedback Mechanisms

4.3.1. Explicit Responsive Feedback

- Using Extra Control Messages

This approach is rarely used because extra packets consume sensors limited energy resources. This technique is followed by **[[17]]** where backpressure messages are used in case of congestion detection.

4.3.2. Explicit Periodic Feedback

This approach is used in **[[29]]** where the upstream nodes of a flow periodically notify their congestion level to downstream (toward the sink) nodes by embedding their perceived congestion level into the header of data packets.

4.3.3. Implicit Feedback

In this approach no explicit feedback happens rather an implicit one occurs from the behavior of the sensor network. Specifically in [37] a node receives an implicit ACK when it hears its downstream neighbor forward a recently-sent packet so this way can infer when congestion occurs. A problem occurs if queue is large when time-out cannot be correctly estimated and also this method does not work when data aggregation happens and the node suppresses some data.

- Piggybacked (as implicit feedback)

Different forms of piggybacking is used in literature. In this case a congestion bit is inserted in the packet (message) header:

- In inverse going traffic (sink to source). In [27] something alike happens when new transmission rate is included in the packet forwarded from root(sink) toward the leafs of a tree based sensor network
- Taking advantage of synchronous link layer NACKS to inform sender if the receiver's buffer excided threshold as happens in [26].
- Overhearing outgoing packets from neighbor downstream nodes as happen in [17 , 14 , 31].
- In RTS/CTS. In [30] proposal the authors suggest that congestion state is inserted in the RTS/CTS signals. The congestion information distributed by the MAC-signals may either include only those cost estimates (path , queue length)belonging to the final destination of the currently transmitted packet, or it may include much more than this, namely the cost estimates to every single node of the network.

4.4. Control Mechanisms

Generally speaking the presence of congestion means that the load is (temporarily) greater than the resources (in a part of a sensor network) it can handle. The following control schemes may be used: decrease the load, increase the resources and MAC enhancements.

Coming back to the WSNs singularities MAC Layer enhancements could help more in the direction of type A congestion**.** If the packet generation rate is sufficiently small, simultaneous transmission of packets becomes independent of the rate. Rather, it depends on the time at which each mote generates the packet. A good way to reduce this type of congestion is to perform phase shifting, an observation made by the authors in **[[37]]**. Small amounts of phase shifting can be performed by introducing slight jitters at the data-link layer. In **[[37]]** also the application layer itself introduces phase shifts. While jittering at the data-link layer aims to cause small transmission variations between neighbouring motes, we think that phase shifting at a higher layer can be achieved on a larger time scale.

Decreasing the load or increasing resources help more in the direction of Type B of congestion Control as helps in emptying buffers of intermediates sensor nodes.

More precisely this mechanisms appear in WSNs in the following forms:

4.4.1. Decrease the load

This goal can be achieved in several ways:

4.4.1.1. Rate Control: In this case load is decreased by decreasing the rate of sources generating information. The rate by which sensor nodes transmit their own sensor reading can be controlled. Alternatively if this is not possible due to application constraints, the number of nodes generating at this fixed rate can be controlled. This control can be executed in an end to end (global) **[[17]]** fashion or locally **[[17, 25, 14, 26]]**. In the end-to-end case the ultimate receiver, for example the sink causes the transmitting nodes to reduce their rate. The sink uses some feedback mechanism like for example acknowledgments or dedicated signaling packets. This can be done to all the sources or to a portion of them or another way on deferent level to different sources. In the local case , a node X might be signaled by its next hop forwarder Y, that Y's buffers are full. Then Node X can reduce its rate or propagate the overflow signal further backward. This signal can be either explicit or implicit one.

A different approach of rate control is proposed in **[27]** where a new rate is propagated through the tree based sensor network as an information. This procedure continuous until reaching the leafs.

In **[24]** operation is determined by the current network state based on the reliability achieved and congestion condition in the network. It adjusts the reporting frequency (f) of the source nodes aggressively if the received reliability is lower than required. If the reliability is higher than required the reporting rate is conservatively reduced in order to conserve energy.

Authors in **[31]** adapt the rates according to an AIMD control law.

Something similar to TCP could work on sensors as transport layer, if this strategy was accompanied by using highly reliable link layers to avoid losses through channel errors and to keep TCP's assumption that losses occurred due to congestion.

One common problem of rate-based congestion control in sensor networks is the difficulty for an upstream sensor to determine the right amount of rate reduction in response to a downstream congestion. Also another thing is that the traditional AIMD approach (additive increase multiplicative decrease) relies on periodic rate adjustment. Due to environmental dynamics (e.g., background radio interference, multi-path fading, change in the number of active neighbors), the bandwidth available to a congested sensor changes all the time, which would constantly cause upstream sensors to perform rate adjustment. That's why is much more desired to have the upstream sensors to quickly adapt their rates to near-optimal ones without explicit, slow-converging rate-based control.

Another problem is that actual desired target rate depends on the accuracy requirements of the user and general trade-off is that relaxing these requirements tends to reduce the frequency and duration of congestion states.

4.4.1.2. Packet Dropping: This is another way to reduce the forwarding traffic to the sensor network. When a forwarding node having full buffers receives a new packet, it must clearly drop from the buffer the last packet or an older one. All data generated in

wireless sensor networks may not be alike; some data may be more important than others and hence may have different delivery requirements.

Congestion leads to indiscriminate dropping of data, i.e. data of high importance might be dropped while others of less importance are delivered. A node can make a better decision when it has information about the packet importance.

One option is to label each packet with an explicit priority value and let the forwarder to drop the packet of the lowest priority. Priority signaling could be done by inserting to the packet header some kind of information (1 or 2 bits), about the significance and the geographical position of the source that generated the specific packet. For example high important events data could take high priority. The high priority data should receive better service, such as higher delivery ratios and minimal delays. It should also experience low jitter, especially for real-time data. The low priority data, such as periodic temperature readings or measurements of environmental conditions away from the critical area, do not need any special service. In fact, some low priority messages may be dropped or significantly delayed without severe consequences.

These issues are presented in **[33]** where the authors propose a mechanism that it spans the entire application stack; from the MAC protocol to the routing protocol layer and up through the application level code. The mechanism ‘s goal is to examine the complete state of the node, and determine which data should be sent over the radio to best satisfy the application requirements. This mechanism appropriately allocates bandwidth between the many competing tasks on a node giving priority to high priorities tasks.

In heterogeneous sensor networks where environments are employing different sensor for all tasks a message to be sent to the sink must be given a priority, so that bandwidth will be allocated accordingly to satisfy high priority event data. A such approach is used in **[38]** where the message is forwarded to a percentage of its direct neighbors and to an even lower percentage of remotely accessible nodes, so that the high priority messages are delivered to the sink.

A similar approach is used in **[26]** where in the bottleneck-congested node transmission queue a static priority-based queuing is applied and high priority flows packets evict the low priority flows packets.

In **[39]** a prioritization – based packet scheduling is proposed according to the requested velocities decided, which are based on deadlines and distance.

4.4.1.3. In-Network Processing and aggregation: This kind of processing has a direct effect on reducing network load. Since a sensor network is deployed toward specific applications, forwarders know the data they forward and can compress, drop, or aggregate it accordingly.

- Aggregation

Data-aggregation can be used, in order to reduce the amount of information traveling throughout the network. Data-aggregation is a way to counteract network congestion. We believe this is a synergistic approach because it leverages a unique characteristic of sensor networks to solve the congestion problem; Rather than using conventional back-pressure or rate regulation techniques, exploiting correlated events could help to mitigate congestion. So data coming from correlated events (sub-regions of WSN) can be aggregated to specific relays aggregator sensors. This way after receiving some kind of congestion detection, aggregation happens to specific sensor nodes. An adaptive technique, as proposed by **[36] ,** could be more effective. In this case the number of packets whose information will be aggregated (degree of aggregation) may dynamically change according to congestion level, so an acceptable fidelity level is maintained. The problems with this method is how to choose where data aggregators have to be placed and how many / which of them .

- Clustering

In **[40]** clustering techniques also supported with aggregation and prioritization (different traffic classes) is used to alleviate congestion.

The sensors in a cluster adjust their rates as per the relative level of importance of the events to be reported and the congestion state en route the sink, thus improving the timeliness of data delivery for high importance flows and the efficiency with which the available bandwidth is shared between the flows.

4.4.1.4. Application Adaptation

Applications play an important role in preventing congestion. When the networking stack is not ready to accept additional data, it signals applications via send failures. It is then up to the application to respond appropriately. Some applications will simply wait until the stack is ready again. Others may adjust their send rate via an AIMD controller or similar mechanism. Generally, applications will only allow small numbers of packets outstanding in the networking stack at once. Doing so prevents locally-generated traffic from starving route-through traffic.

4.4.2. Increase Resources

In sensor networks, various types of resources exist such as:

- (end-to-end) channel capacity
- remaining energy
- active nodes (i.e. whose radio is on.)
- radio transmission power
- packet buffers in the queue (i.e. memory)

Resources in sensor networks are correlated in the sense that changing the availability of one resource affects the availability of other resources. For example, if the transmission power of a node increases, the remaining energy of a node is drained quickly and the channel capacity is affected. If the channel capacity increases, the packet buffer occupancy is also lowered.

According to **[41]** , the choice of resource control is necessitated by the following two factors: (1) unlike in traditional networks, we cannot simply reduce the source traffic because these data are critical to the applications; and (2) sensor networks usually are densely deployed and thus have the capability of provisioning more resources when needed. Resource control scheme is desired to have two phases: (1) to increase resource provisioning as soon as congestion occurs; and (2) to reduce the resource budget as soon as congestion subsides.

Proposed scheme adjusts the resource provisioning based on the congestion level so that we can both increase the capacity, by having more sensor nodes forwarding data (put more sinks or wake up nodes) or having more routing paths, to alleviate the

congestion, as well as reduce the capacity after the congestion, to conserve the energy consumption.

According to congestion types different techniques for resource control are suggested.

4.4.2.1. Add more sinks

This way the capacity of the network is increased as the traffic is directed to the specified sink. All this information from all the sinks is later associated together. A similar approach is used from **[[32]]** when some sensor nodes under congestion conditions become virtual sinks using multi-radio capabilities and forward traffic toward the physical sink in a secondary uncongested network. A similar approach is also used in **[[26]]** where the results say that adding more sinks in the sensor network the network capacity is improved as the load balancing that happens to help on this way.

4.4.2.2. Wake up nodes

During a dormant state, the sensor network usually turns off a large number of nodes to extend the network lifetime. However, these nodes need to wake up periodically to check whether they are needed, such as when a node on the routing path runs out of battery. The time between two subsequent wake-ups is referred to as the *sleep interval*. The choice of the sleep interval is of great importance because it is undesirable to wake up too frequently or too infrequently. Numerous studies have focused on how to calculate an optimal sleep interval. In order to make the backup nodes congestion-conscious, authors of **[[29]]** propose to adapt the sleep interval based on their congestion level.

Every time when a backup node wakes up, it measures its congestion level. If it observes that current congestion level is greater than the previous measurement, it will shorten its sleep interval and wake up more often. As soon as its local congestion level is above a threshold, it will significantly reduce its sleep interval and becomes “alert”.

As soon as there is a hot spot, the backup nodes around the hot spot will become active forming one or more additional path (*multiplexing paths)* so that incoming traffic can be distributed over the original path and the multiplexing paths.

4.4.2.3. Careful Routing – Load Aware

Several issues have been proposed in literature.

In **[23]** a dynamic Load Aware routing (DLAR) protocol is proposed that consider intermediated node routing loads (output queue length) as the primary routing selection metric. It also monitors the congestion status of active routes (piggybacking load information to data forwarded to the sink) and reconstructs the path when nodes of the route have their interface queue overloaded. Three route selection algorithms are proposed;(1)least sum load;(2)average buffered packets along the path to the sink;(3)number of congested intermediate nodes along the path.

In **[28]** a neighbor sensor node becomes the new packet forwarder in case it overhears a congestion bit from the normal forwarder. When the congestion state ends the normal forwarder is activated again. This way traffic bypass congested nodes and no packet loss occurs.

In **[30]** a distributive routing & congestion control has been proposed, that uses routing costs and congestion state (queue length) to decide next hop selection. It couples the MAC- and routing layer of wireless multi-hop ad hoc communication. Before one-hop forwarding a packet, the sending as well as receiving node MAC-block their respective neighbors distributing information about their congestion state and routing cost estimates, which the latter then use for updates. This distributive scheme turned out to be very efficient.

Such an approach is used in **[25]** where congestion avoidance is achieved through multi-path selection toward the sink. The selection of next forwarder is done according to its congestion state.

In **[42]** the authors use a routing protocol called congestion aware routing that uses multi-pathing routing, where the high priority traffic is delivered to the specific sink through the congestion area and low priority traffic outside that area to the other sinks, avoiding congestion and losses of high priority data providing for them better service.

In **[29]** a resource control scheme is suggested; when an event occurs, and when congestion occurs sleeping sensor nodes wake up and help in alleviating congestion by forming traffic multiplexing. More routing paths are created and share load in a round robin manner until congestion stops. Alternative path creation is achieved using the

following 3 criteria (1) congestion level; (2) energy level; (3) proximity to the hot spot. The alternate paths are deconstructed when congestion alleviation is achieved.

In **[32]** sensor nodes become temporary virtual sinks and reroute traffic through a secondary (another radio frequency) to the one physical sink.

4.4.3. MAC Enhancements

Although sensors can react to congestion using the above network-layer mechanisms, they cannot always react to congestion fast enough to prevent buffer losses without some help from the MAC layer. Local prioritization at each individual node is not sufficient in wireless networks because packets from different senders can compete against each other for a shared radio communication channel. To enforce packet priorities, MAC protocols should provide *distributed prioritization* on packets from different nodes. Extensions (e.g., [DIFF802.11]) of the IEEE 802.11 wireless LAN protocol have been investigated to provide distributed prioritization.

A standard CSMA MAC layer gives all sensors competing to transmit an equal chance of success. However, during times of congestion, this approach can lead to reduced performance due to a congested sensor's inability to quickly propagate congestion control feedback to its neighbors. For example, consider a high fan-in scenario, where several sensors are forwarding through a common parent. On average, the parent sensor will gain access to the channel only after half its neighbors have transmitted. However, since the parent is congested, it may not have enough buffer space available to store packets from its children. Hence the parent has no choice but to drop packets that its children forward it.

Consequently, it is imperative that congested sensors have prioritized access to the wireless medium.

Several MAC Layer enhancements have been proposed as solution for congestion problem in sensor networks. In **[8 , 26]** the authors propose a priority-based MAC where it is ensured that high priority packets that are waiting to be sent their nodes will win any competition at the MAC Layer. This is done for example by making the length of each sensor’s randomized backoff a function of its local congestion state.

To address these issues in [14] they adopted a technique proposed by [43]. Specifically if a sensor is congested, its backoff window is one-fourth the size of a non-congested sensor's backoff window, making it more likely that a congested sensor will win the contention period, allowing queues to drain and increasing the likelihood congestion control information will propagate throughout a sensor's neighborhood.

In [39]802.11 sets a DIFS counter once the communication channel has become idle. Before sending an RTS (Request To Send) packet, a node will wait a random period of time between 0 and DIFS. To prioritize this process they set the DIFS parameter based on the packet priority:

DIFS = BASE_DIFS * PRIORITY

Packets with a higher priority (corresponding to a smaller PRIORITY value) on average choose a smaller waiting period.

Backoff Increase Function 802.11 doubles its backoff window, CW, to extend a node's waiting period when a transmission collision occurs. 802.11 is modified to increase CW in accordance with the packet priority1:

CW=CW*(2+(PRIORITY-1)/MAX_PRIORITY)

MAX_PRIORITY is the maximum value of priority (corresponding to the lowest priority). The backoff counter of a node with a pending lower priority packet increases faster than a node with a pending packet with a higher priority.

The above two mechanisms give high priority packets high probability to get the channel in both the contention avoidance and contention phases.

In [37] it is proposed to use random delay before listening, so not to be synchronized from same time event detection. .

In [34] a different enhancement is proposed. Furthermore the classical Type A and B congestion, congestion occurring to RTS/CTS blocking affect must be faced. In [34] authors suggest a simple solution to the false-blocking problem, called *RTS* Validation. A node that uses *RTS* Validation defers for an entire packet transmission period if *DATA* packet transfer begins, but defers only for a short time if no transmission takes place when *DATA* packet transmission is expected. By means of simulation, they have shown that the use of *RTS* Validation improves the network performance in three aspects: it eliminates congestion by stabilizing the network throughput at high load, it

increases the peak throughput by 60%, and it significantly reduces the average delay. *RTS* Validation is a backward compatible solution and thus can be implemented incrementally.

Some others try to avoid the use of RTS/CTS packets in MAC layer to eliminate overhead, and suggest an alternative to face the hidden terminal problem. In **[37]**they attempt to avoid hidden terminal problem, and to do so, they assume that packets will be routed after some processing time x. A child node is able to avoid a potential hidden node problem with its grandparent. The idea is that if a child node hears the end of its parent's transmission at time t, it should expect that its grandparent will route its parent's packet starting at time t + x. Therefore, if the child node can restrain from transmitting from time t to t + x + PACKETTIME, the hidden node problem can be reduced. In fact, if the child node detects such situation it should perform a backoff to change its phase such that it will not encounter the same situation the next time it transmits.

4.4.4. Cross Layer Design Optimization

The majority of congestion control algorithms state that cross-layer interactions between transport layer and MAC layer are imperative for efficient congestion detection and hence congestion control. In **[17]**, channel load information from the MAC layer is incorporated into congestion detection and control mechanisms. In a converse approach, the authors in **[37]** propose transmission control scheme for use at the MAC layer in WSN. In **[24]** , congestion detection is performed through buffer occupancy measurements. In **[43]** , the backoff window of each node is linked to its local congestion state. Furthermore in **[30]** they proposed incorporate information from MAC and network Layer. Congestion detection uses buffer (network layer)or channel load(mac layer) to control routing (network layer).In **[42]** application layer (traffic information) is used as metric decision for the routing protocol. Furthermore, **[14]** compares the buffer occupancy-based and channel load-based congestion detection mechanisms. It has been advocated that MAC layer support is beneficial in congestion detection and control algorithms.

The schemes referenced above are either directly applied or indirectly related to congestion control in WSNs. Complete mechanisms support all 3 sub mechanisms of

congestion detection, feedback. A classification of all the referenced mechanisms is depicted in Table 4-1.This is accomplished in accordance to **[[35]]** work.

According to Table 4.1, which is a summary of known congestion control schemes in the literature most of them are included in closed loop category as they use feedback. This seems to be more efficient as monitor the network performance dynamically. Detection method most frequently used is that of buffer occupancy. This doesn't appear to be accurate enough in case MAC reliability is not used as when packet drops occurs due to collisions buffer drains and size is decreased. Well known schemes use a combination of channel utilization and buffer occupancy, as seems to be the most effective method. Fidelity measurement at the sink is also used but is more slow method, as end-to-and and processing delay is added at the sink.

Talking for Feedback methods local ones are most frequently used as transient congestion problems are most usual in wireless sensor networks and quick reaction is needed in these situations.

Control Methods used in different schemes is the most important part. The use of rate control is very common and in most cases is effective enough to control congestion but with a cost on application fidelity, as most of the times during the crisis state less sensing data are delivered to the sink than the application requires. An alternative is resource provisioning. As in sensor networks "redundant" sources can be used to achieve larger network capacity, this seems to be very promising control method. Of course these methods have high cost on energy consumption, so several measures have to be taken to avoid this problem. In – network processing is another promising method. This is because of the data-centric nature of wireless sensor networks. As sensor nodes can process data before forwarding them, using aggregation or other data processing functions, can reduce the forwarded traffic when congestion happens. The effectiveness of this method of course again is limited from the sink's application requirements. Methods that use MAC enhancement is not very promising when work alone, but in combination with other methods can be very efficient. Note that MAC enhancement is important as shared medium must be controlled in a manner and give priorities to congested flows of the sensor field. Finally Careful Routing and Load balancing is not always efficient. Except

that more sinks are needed, balanced traffic could create other points of congestion in the network.

4.5 Problem Statement

As shown in Chapter 3, Congestion leads to both waste of communication and energy resources of the sensor nodes and also hampers the event detection reliability because of packet losses.

In real WSNs can manifest in 2 main ways.

- Excessive Packet loss in the radio itself.
- Overflowing queues on nodes.

Hence, it is mandatory to address the congestion in the sensor field to prolong the network lifetime, and to provide the required quality of service (QoS) that WSN applications demand.

As seen from the literature, several control schemes appears, most showing not to be effective enough, on controlling congestion. Even rate control methods that appear to be the most effective they cannot meet sink's application fidelity requirements during crisis state. In that conditions Recourse provisioning methods show to be more promising but with a large cost on energy consumption.

What is missing is a global framework that can combine these two schemes to succeed congestion alleviation and meet sink's application requirements in one hand and conserve energy on the other.

SCHEME	Category		Detection Method			Feedback		Control Method					
	Open Loop	Closed Loop	Buffer overflow	Channel utilization	Fidelity at the sink	Local	End to end	Rate control	Resource provisioning	In networking Processing	Careful Routing – Load Balancing	MAC Enhancement	Packet Dropping
CODA [17]		√	√	√	√	√	√	√					
ESRT[24]		√	√		√		√	√					
MITIG[14]		√	√	√		√		√				√	
CCAF [27]		√	√			√		√					
CALWB[25]		√	√			√		√			√		
DIFFUSSION[44]										√	√		
RAP[39]												√	√
SIPHON[32]	√				√		√		√		√		
CONCERT[36]	√		√	√						√			
TRANSCONTROL [37]		√				√		√				√	
BANDMANAG [26]		√	√			√		√				√	√
LOCDRIVEN [38]	√			√									√
IFRC [31]		√	√	√		√		√					
ADAPTRESCONTROL [29]		√				√			√		√		
WIRELADV [28]		√	√			√					√		
RTS/CTS/IND [34]												√	
CAR [42]											√		√
CLBCCMC[40]		√				√				√		√	

Table 4-1 WSNs Congestion Control Schemes Classification

Chapter 5

STUDY OF CONGESTION THROUGH SIMULATIONS – EVALUATION OF CONGESTION SOLUTIONS

5.1. Simulations

5.1.1. Introduction

This study is dived in two sections: Cases of Study A and B. The purpose Section A is to analyse the Congestion Problem in wireless sensor networks, identify the main symptoms, show problem regions and clarify the main causes of the problem. In section B several solutions for different types of congestion are evaluated including transmission random delay, addition of sinks and rerouting –multi-path routing techniques. All the above will contribute later to interesting conclusions and help on finding directives for controlling or alleviating congestion.

In particular in Section A of simulations, we investigate the effect of congestion using as performance metrics : packet drops, end-to-end delay, throughput and reliability. Also we examine how congestion affects energy consumption and fairness. A brief presentation of various congestion types is performed using simulations and finally various network parameters are tested as to how they affect congestion. The various scenarios were implemented and run in a recent version of NS-2 **[[45]]** simulator (version 2.29).

The purpose of Sections' B simulations is to confirm that specific proposed resource control solutions can be used successfully as part of our proposed Congestion Control Framework.

Section B of Simulations is based on some of the results of section A(congestion problem of various types, network parameters effect on congestion) and related work of Chapter 4. Here we study the congestion resolving power of three resource control strategies, based on wireless sensor network topology changes and control of number of sources. Three

simulation scenarios types are presented, showing the effect of different solutions to the congestion problem, using random transmission delay, adding more sinks and rerouting and multi-path routing. Each scenario type is divided in sub scenarios, each one modifying different parameters in order to observe how they affect the performance.

Several Scripts were written (see Appendix A) in order to transform the raw data produced by the simulator **(using new wireless trace format**) for the creation of the graphs presented in this chapter. Specifically:

- stats.txt (AWK scripting language). To measure the throughput, end to end delay, max delay, total packets sent, receive, number of collisions. energy.txt(AWK scripting language). Calculates total energy consumption for each node, average energy consumption, total network energy consumption.
- drops.txt(AWK scripting language). Calculates the number of packet drops due to MAC , Routing , Buffer Overflows for each sensor Node.
- matlab files . To plot data on graphs
- xgraph to plot data. To plot run time losses/performance
- NAM animator. To animate simulations.

5.1.2. Simulation Environment

It is important to mention that NS-2 provides some extensions for wireless sensor nodes, (Mannasim and NRL extension) that implement specific protocols, and the creation of events. For the purpose of this thesis, needed sensor node configuration capabilities and protocols are used from the basic version of NS-2 (Figure 5-1)with several modifications either on TCL file or .cc, .h files of specific objects. Sensor Node is a modified Mobile Node that extends Node Object.

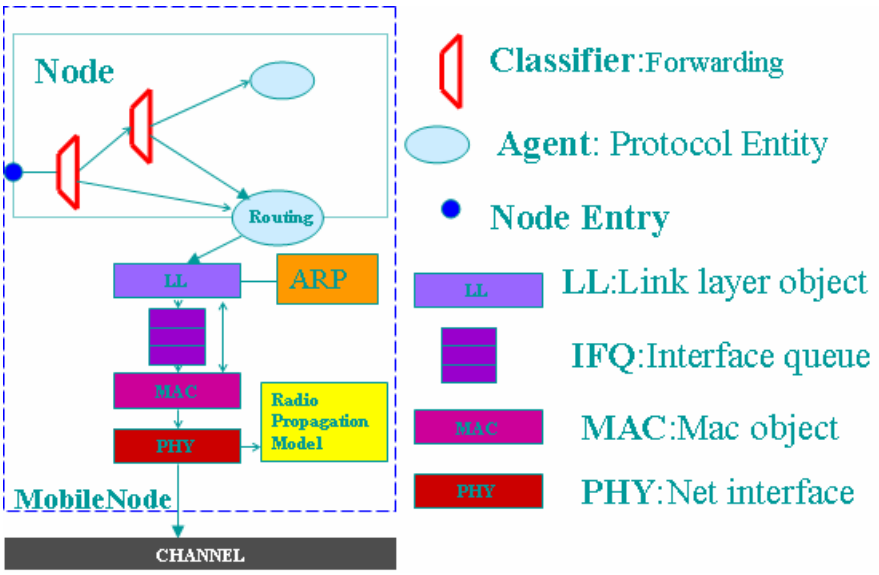


Figure 5-1 Mobile Node Object NS-2

Our Static Routing (StaticRT) is used as the routing protocol in almost all simulations except that when ad- hoc routing is used and it is clearly defined in tcl files(Appendix A)

StaticRT gives us an alternative routing protocol for our simulations as is fully static. The idea was to modify DSDV and build a completely static routing protocol. This helps in avoiding any influences of routing protocols to congestion investigation. Details about the implementation of StaticRT can be found in Appendix A.

IEEE 802.11 with the coordination function (DCF) is chosen as the MAC protocol. The data rate of IEEE 802.11 is set to be 2 Mbps. Of course this could be changed in NS-2 from TCL file . An example is given in(Appendix A.2). A unicast data packet destined to a neighbor is sent out after handshaking with request-to-send /clear-to-send (RTS/CTS) exchanges and followed by an acknowledgement (ACK) packet. The broadcast packets are simply sent out without handshake and acknowledgement. The implementation uses carrier sense multiple access with collision avoidance (CSMA/CA). both with and without RTS/CTS.

To imitate MICA 2 , sensor node, the energy model adopts the parameters shown in

Table 5-1 and accordingly we configured the sensor node(Appendix A.2):

STATE	POWER CONSUMTION
IDLE	0,0135W
RX	0,0135W
TX	0,02475W
SLEEP	0W
Total Energy	2Joules

Table 5-1 NS-2 Energy Model Parameters Setup

In all simulations, two-ray ground reflection, error free, model is employed as the propagation model. Parameters shown in Table 5-2 are tuned to make the receiving distance to be 30 meters and the sensing distance to be 30 meters to as to imitate real sensor nodes. This can be obtained by running the program ~ns/indep-utils/propagation/threshold.cc .

Each time a mobile node transmits a frame in a simulation, ns-2 uses a propagation model to calculate the receive power of the radio signal for every potential receiving node. Similar to real world, the frame is received correctly if the corresponding signal strength is not below the receive threshold of the network equipment (RXThresh). If the receive signal strength is below the receive threshold but it is above or equals the carrier-sense threshold (CSThresh), the frame is received with errors. Another frame arriving at the same time causes a collision if its signal power is not at least collision threshold (CPThresh) times below this frame's signal strength. All frames with a power below the carrier sense threshold are ignored by the receiver.

PARAMETERS	VALUES
X coordinate	0
Y coordinate	0
Z coordinate	1.5
Gt antenna transmit gain	1.0
Gr antenna receive gain	1.0
CPThresh	10.0
CSThresh(carrer sensing threshold)	2.13643e-07
RXThresh (receiving threshold)	2.13643e-07
bandwidth	2e6
transmit power	0.28183815
frequency	914e+6
Systems loss	1.0 db

Table 5-2 NS-2 802.11 Radio Parameters Setup

Before creating the simulator, we proceed to the above settings in TCL file(Appendix A.2)

A generated grid topology network is used for the simulations, and it is placed in a 500x500 square field. This is set in TCL file(Appendix A.2)

The sensor nodes are created again from TCL file. For example Node 2 is created with the code in Appendix A.2, in the specific coordinates and size on NS animator.

Sensor Nodes Buffer size is set to 50 packets maximum, and queue type is set to priority/drop tail queue(Appendix A.2)

Sensor nodes are static in all topologies. In the BASELINE tests only one sink is placed, sensor node 45. Figure 5-2 illustrates the topology of the 100-node simulation network.

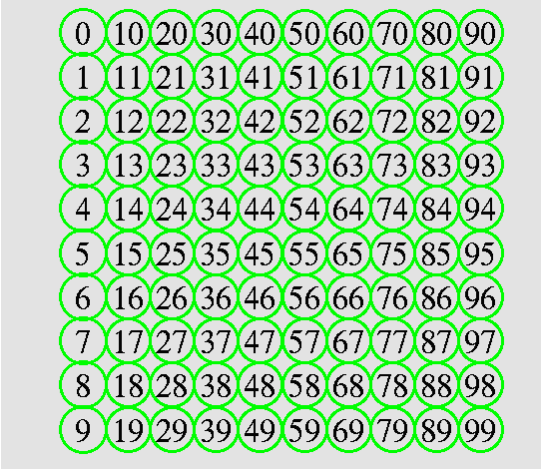


Figure 5-2 Wireless Sensor Network Topology

Sources are sensing the upper left corner of the sensor field grid and forward data to the sink. Source nodes are 0, 1 ,2,10,20 to emulate source to sink congestion. To investigate the impact of traffic load and congestion we study UDP connections. Every source is associated with a constant bit rate (CBR) traffic generator, which sends out packets at the given rate. The packet size is fixed at 100 bytes (to imitate real sensor packets). The start time of each connection is specified between 10 to 20 seconds. Source nodes report the event to the sink node at a pre-defined rate of 10,20,30,40,50,80,100,150,250 Kbps. Traffic pattern is set in TCL file as mentioned in Appendix A.2.

All the parameters unless explicitly specified, are exactly as described above. This is the configuration of the BASELINE test.

5.1.3 Cases of Study Section A (Simulation Scenarios Summary)

Various simulation scenarios and tests have been implemented, to investigate the Congestion Problem and it's symptoms on Sensor Networks and examine the effects of several network parameters to it.Simulations are grouped as followed:

- Using baseline scenarios we generally analyze congestion problem(in terms of throughput, delay, packets drops etc) in wireless sensor networks
- Second group of scenarios test congestion effect on fairness and energy consumption. Fairness is important when sensing the field regions is equally important.

Also energy consumption is very important as is the most limited resource of sensor node.

- In third group of scenarios using also baseline scenarios we investigate congestion problem of most well known Congestion Types. This will help to differentiate congestion problem for each type and accordingly take measures later to control or alleviate the problem.
- In forth group several scenarios are implemented to check the effect ov various parameters effect on congestion in wireless sensor networks. Particularly we test the effect of the number of sources using either fixed total reporting rate or not, RTS/CTS mechanism, maximum number of retransmissions on MAC Layer, sensor node buffer size, minimum initial Contention Window, average path length to the sink and finally ad-hoc routing protocols effect on congestion. These tests will hope to give us useful directives for network parameters configurations or adjustment that can help to control the congestion problem to some extend.

These scenarios are listed in Table 5-3:

Scenarios	Investigation	Test #	Notes
Congestion Problem Analysis			
100sensorfield5xX.tcl	Congestion Analysis of **BASELINE** scenarios	1	**BASELINE SCENARIOS**
Congestion Key Symptoms			
2fairsensor6hops10sX.tcl	Congestion effect on Fairness	2	X==50,80,150,400Kbps
100sensorfield5x50.tcl 100sensorfield5x150.tcl 100sensorfield5x350.tcl	Congestion Effect on Energy	3	
Congestion Problem for Most well known WSNs Congestion Types			
100snesorfield5x20.tcl	Congestion Type Near the Source	4	
100sensorfield2x80sink.tcl	Congestion Type Near the Sink	5	

Scenarios	Investigation	Test #	Notes
100sensorfield2x80merg.tcl	Congestion Type of merging Traffic	6	
100sensorfield2x80cross.tcl	Congestion Type of Cross Traffic	7	
Various Wireless Sensor Network Parameters effect on Congestion			
100sensorfield1x100.tcl 100sensorfield2x50.tcl 100sensorfield3x33.tcl 100sensorfield4x25.tcl 100sensorfield5x20.tcl 100sensorfield9x11.tcl	Number of Sources and Sensor field range effect on Congestion	8a	Keep aggregate traffic rate constant
100sensorfield1x50.tcl 100sensorfield2x50.tcl 100sensorfield3x50.tcl 100sensorfield4x50.tcl 100sensorfield5x50.tcl 100sensorfield6x50.tcl 100sensorfield9x50.tcl		8b	3x50 means 3 sources ,and traffic rate is 50Kbps
10sensorfield5xX norts.tcl	Effect of RTS ,CTS mechanism on Congestion	9	Comparing with 100snesorfield5xX.tcl
100sesnorfield5x5Xret0.tcl 100sesnorfield5x5Xret1.tcl 100sesnorfield5x5Xret4.tcl 100sesnorfield5x5Xret7.tcl 100sesnorfield5x5Xret10.tcl	Effect of maximum number of MAC retransmissions on Congestion	10	ret0 means no retransmissions ret 4 means max 4 retransmissions
100sensorfield5xXb5.tcl 100sensorfield5xX.tcl 100sensorfield5xXb100.tcl 100sensorfield5xXb200.tcl	Effect of Sensor Node Buffer Size on Congestion	11	b5 means buffer size is 5 packets
100sensorfield5xXcw64.tcl 100sensorfield5xXcw128.tcl	Effect of minimum Contention Window(CWmin) on Congestion	12	cw64 is for cwmin=64

Scenarios	Investigation	Test #	Notes
100sensorfield5x50_11.tcl 100sensorfield5x50_ 22.tcl 100sensorfield5x50_ 33.tcl 100sensorfield5x50_ 44.tcl 100sensorfield5x50_ 77.tcl 100sensorfield5x50_ 99.tcl	Average Path Length to the sink effect to congestion	13	Sink node is changing in each scenario as 11,22,33,44,77,99
100sensorfield5xXdsdv.tcl 100sensorfield5xXaodv.tcl	Ad hoc routing protocols effect on Congestion	14a	
100sensorfield5xXaodvinit.tcl	Initialization Traffic Effect on AODV Congestion	14b	

Table 5-3 Simulations Scenarios Summary

.

During this thesis, we summarize the several types of packet drops that play an important role in the simulations. The reasons for these packet drops are listed in Table 5-4. Congestion Related drops occur in the link-layer. The analysis of packet drops from NS-2 wireless trace format is done with our analysis program drops.txt, awk program. This program differentiates packet drops for each node, and calculates the totals.

Category	Reason	Description
Routing Layer Drop	Send Buffer Timeout (1)	Every data packets are saved in the send buffer before they're sent out. If a packet has not been sent out after certain timeout, it will be dropped. In most cases, a packet is timeout because no routes are found within the timeout. (The AODV implementation in NS-2 poses a 30-second limit on the time a packet can be buffered.
	No route(2)	If a packet is undeliverable (e.g. due to link error) and has been salvaged too many times, AODV will consider there is no route available and drop it.
	TTL reaches zero(3)	If the TTL value in a packet reaches zero, the packet will be dropped.
	Drop by misbehaved Nodes(4)	A misbehaved node could drop data packet, route reply, route error.
Link Layer Drop	IFQ full (1)	Packets are buffered in a network interface queue before they are sent out. If too many packets are generated before previous ones are sent, packets will be dropped
	ARP full(2)	Before a packet is sent, the MAC address of the destination node must be searched by ARP. If the ARP is full, the packets depending on those nodes will be dropped.
	MAC RET(3)	The wireless channel is so busy that the times of ***back*** *off* exceed the limit.
others	Simulation terminate	Packets that are saved in send buffer of routing layer and IFQ will be dropped when the simulation ends

Table 5-4 Packet Drop Types

5.1.3.1 TEST 1

In the BASELINE test we investigate the problem of congestion and its main symptoms. Various scenarios were implemented and named as :**100sensorfield5xX.tcl (Appendix A).**

Five sources (0,1,10,20,2)are sending CBR (UDP) traffic to one sink (node 45) using static routing as depicted in Figure 5-3. Event occurs between 10sec-20sec ,which is the crisis state of our wireless sensor network .

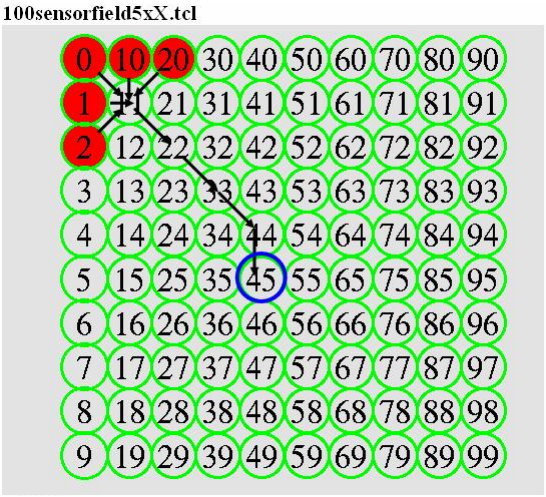


Figure 5-3 BASELINE topology

This scenario will be our baseline for further investigation later, when examining the effect of various network parameters on Congestion Problem. The following results will be compared later on with other simulation results.

As seen in Figure 5-4 packet drops occur between 10-20 sec when event happens and network is in crisis state.

Looking at Figure 5-5 , we can observe that over 125,5 p/s of reporting rate, network start drop packets and things get really worst when traffic rate goes over 250p/s.

As mentioned in section 3.5 we define two reporting rate thresholds, denoted as rlow and rhigh, which represent the reporting rates at which network behavior changes significantly. These rates are specific for BASELINE test (this specific network configuration).Threshold rlow is the rate at which network starts to be congested which in BASELINE test is about 125 p/s.The rhigh is the point where wireless sensor network reliability starts to saturate, here about 938p/s.

The region below rlow is called non-congested region, the region between the thresholds is called transition region and the region between beyond the second threshold is the highly congested region.

In Figure 5-6, it is important to note that maximum number of MAC Layer errors occurs around rlow as contention occurs before congestion starts. When the congestion starts buffer overflows(Figure 5-7) dominates packets drops.(saturates at 95% and MAC layer saturates at 5%).

As seen in Figure 5-5 the reliability of the event is about 10% in high congested region. From the above discussion we can conclude that when congestion happens some network symptoms appear and these are degradation of throughput(Figure 5-8), sharp increase in delay (Figure 5-9)and sharp decrease in event reliability(Figure 5-5).

At next 2 tests we are going to investigate two more symptoms of congestion in sensor networks which are energy consumption and unfair event data delivery.

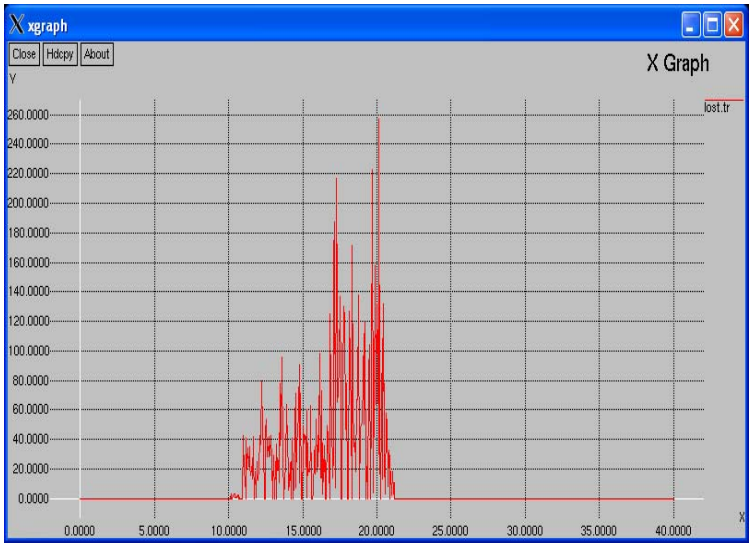


Figure 5-4 Simulation Time Packets Drops

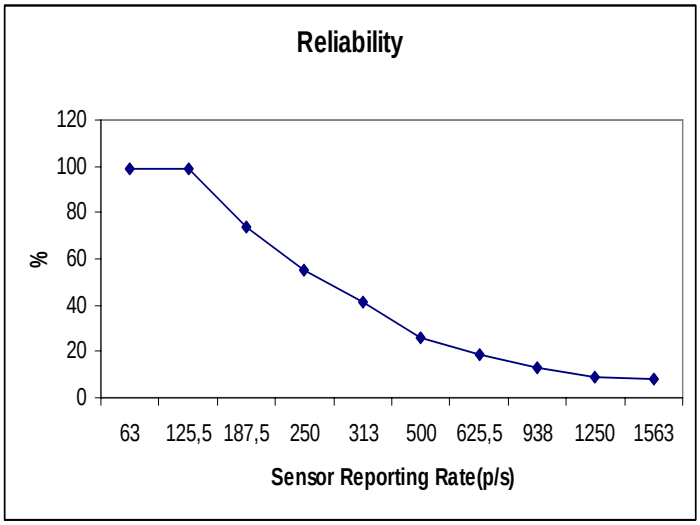


Figure 5-5 Reliability BASELINE

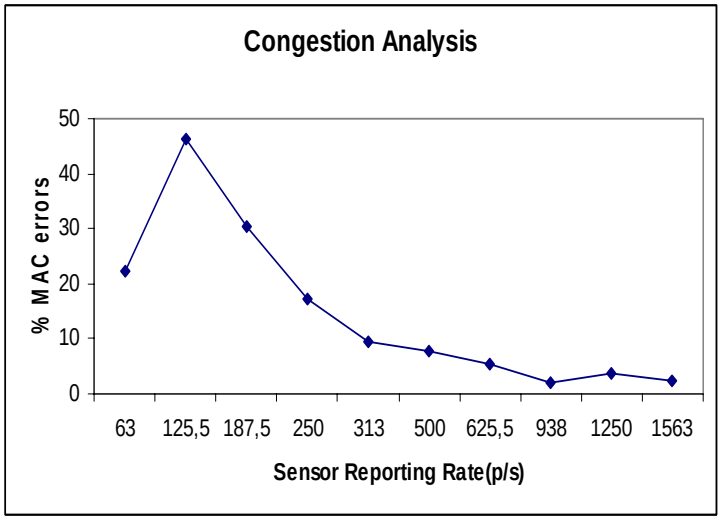


Figure 5-6 MAC Errors BASELINE

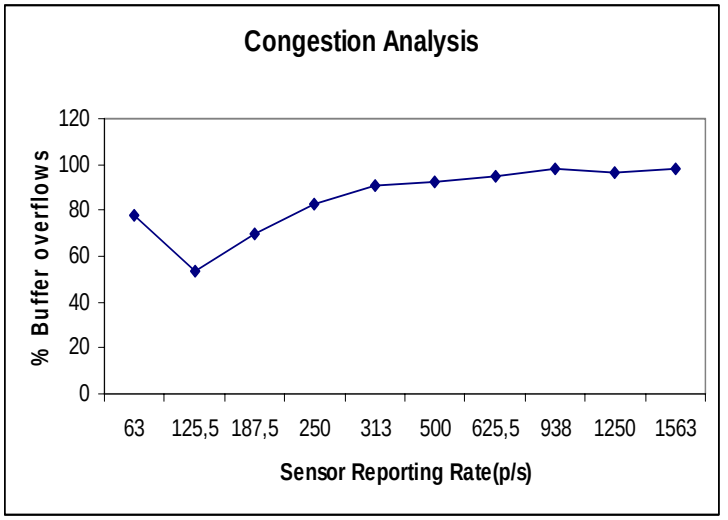


Figure 5-7 Buffer Overflows BASELINE

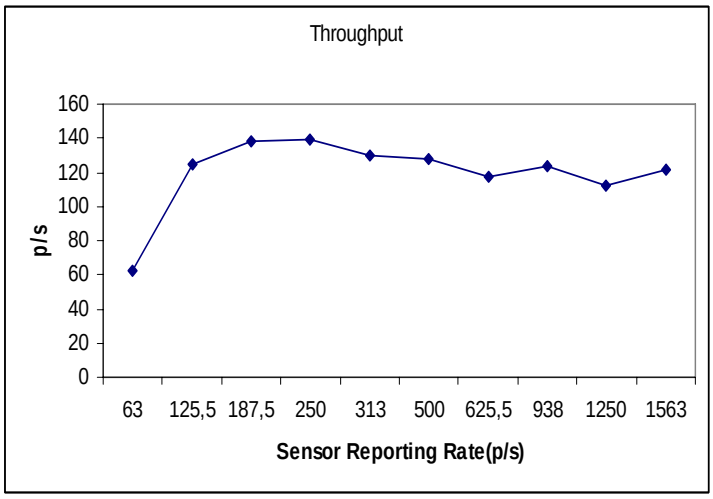


Figure 5-8 Throughput BASELINE

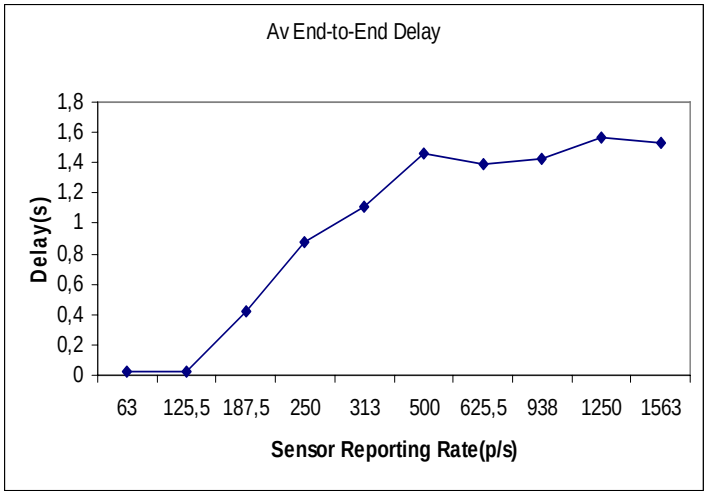


Figure 5-9 Av. End-to-End Delay BASELINE

5.1.3.2 TEST 2

During this test we investigate the way the congestion problem affects fair transmission of data, in multi-hop wireless sensor networks. Topology was completely changed from BASELINE test for practical reasons. Two sources are sending UDP traffic at a sink. Four scenarios, 2fairsensor6hops10sX.tcl**,** were implemented with sources generating CBR traffic with rate values of X: 50,80,150,400kbps. Static routing was used(StaticRT) with routing table:table2.txt.

The topology of Figure 5-10 was implemented and two flows 0 (source 8)and 1 (source 7)were compared for packet delivery ratio.

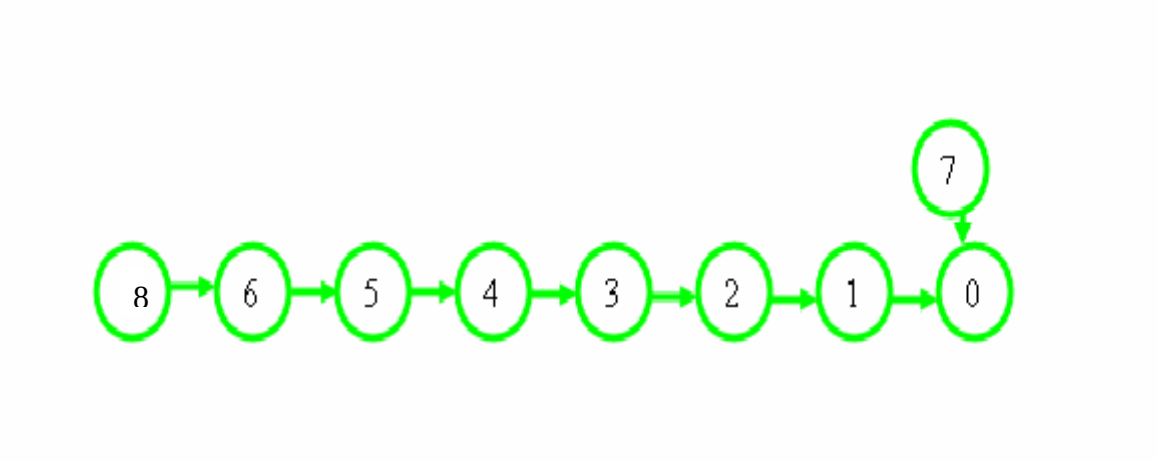


Figure 5-10 ‘’Fairness’’ Topology Network

In such multi-hop networks far sensor nodes suffer from starvation , due to traffic from nodes one hop away from the sink. As illustrated in Figure 5-11 flow1 gets disproportional larger amount of network bandwidth as offered load increases. The far node(flow 0) starts starving after aggregate network traffic rate reach 200p/s. At 400kbps the percentage of flow’s 1 packets that reach the sink is decreasing almost 40%, whilst flow 0 deliver all its packets to the sink. It is clear that source 8 due to multi-hop path to the sink decrease its end to end capacity, and when overcomes this, packet drops occurs, whilst source 7 which is 1 hop away from the sink gets more end to end capacity and can send traffic with higher data rate.

Results for each flow has been traced using awk command:

awk ‘$1==”s” && $19==”AGT”&&$35=”cbr”&& $39==”0”’ tracefile.tr >>data.txt

as $39 is the column for flow id information, s is for sending packets, AGT means trace at the application Layer, and cbr is for constant bit rate packets.

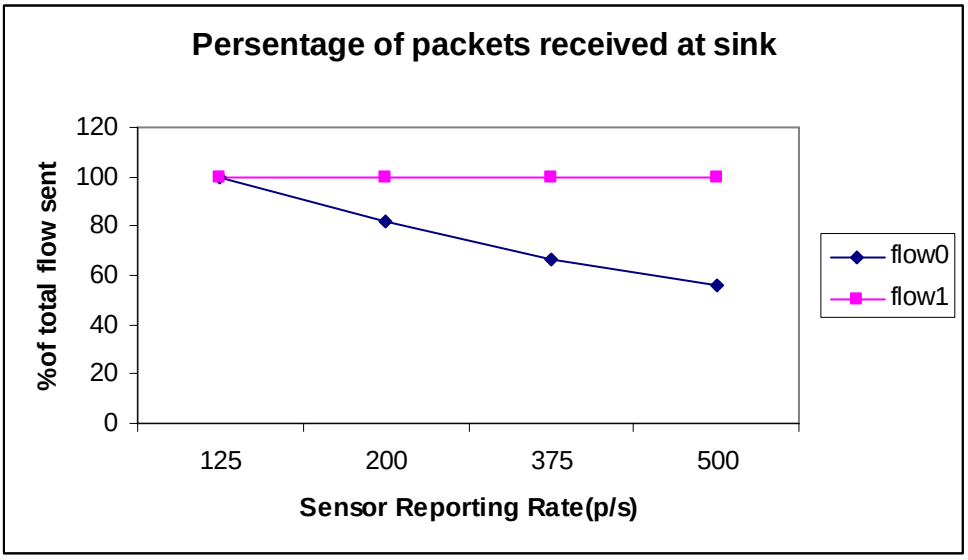


Figure 5-11 Congestion effect on Sensors Starvation

5.1.3.3 TEST 3

In this test we investigate the relation between Congestion & Energy Consumption.

BASELINE scenarios are used for simulations. Specifically 3 scenarios are compared 100sensorfield5x50.tcl , 100sensorfield5x150.tcl , 100sensorfield5x350.tcl , to investigate energy consumption on low, moderate and heavy congested sensor networks. Energy Model is used from NS-2 and is included in TCL file in the configuration of the node. Details has been given in section simulation environment. Energy consumption in NS-2 is calculated according to Equation 5-1:

$$E^{total} = \sum_{i=1}^{n} (p^{xmit} xt^{xmit}{}_{i} + p^{recv} xt^{recv} + p^{idle} xt^{idle})$$

Equation 5-1 Sensor Total Energy Consumption

From Figure 5-12 we see that sensor nodes 11,22 and 44 consume the most energy as they forward the most traffic. The node with maximum consumption is node 11 as it is the most congested and so most of the time(during crisis state) is in transmit and receive radio state. As the traffic rate increases there is an increase in total and average energy

consumption. Sharp increase starts at congestion starting at 125,5 p/s. This can be seen from Figures 5-13 and 5-14. Beyond the value at 625,5 p/s the energy consumption saturates as tx and rx times of all sensor nodes get maximum values.

The result depicted in Figure 5-15 is very interesting as average energy consumed per delivered packet is higher in no congested region. This is because when traffic rate is low, idle listening, keep consuming large amount of energy while less packets are delivered to the sink. As reporting rate is below 125,5 p/s average energy consumption per delivered packet decreases. Of cource when congestion starts(125,5p/s) energy consumed per deliver packet start increasing as less packets are received due to more packets drops.

The network wastes energy transmitting bits from the edge towards the sink only to be dropped(livelock). Figure 5-16 shows that energy consumption due to packet drops is getting increased as traffic rate increases and reach a percentage of 92% of tx-rx total energy and 20% of total energy consumption. To calculate these values we found out the energy consumed for transmit and receive states, using simulations(nullify Idle listening energy consumption) and from the percentage of packet drops we calculated the percentage of energy consumed for these losts .

In Figure 5-17 energy consumed on bottleneck node gives an indirect indication of congestion. When no congestion occurs energy consumption is low and when is fully congested energy consumption is saturated. Also as seen high congested networks dangerously decrease the network lifetime as bottleneck node consumes energy more quickly.

To examine the energy consumed per node's radio state we nullify the energy consumed for the other node's radio states in NS-2 energy model.(Analogous setup in TCL) That way we get the results of Figure 5-18, where most energy consumption happens due to idle state. This is why idle state takes most time of simulation as traffic is generated from nodes for a duration of 10sec.On the other hand we have to note that power consumed for idle mode is the same as that consumed for receiving mode. Comparing this behavior at different congested scenarios we can observe that in most congested networks energy consumption for idle state is decreased while that of transmission state is increased and it is completely normal.

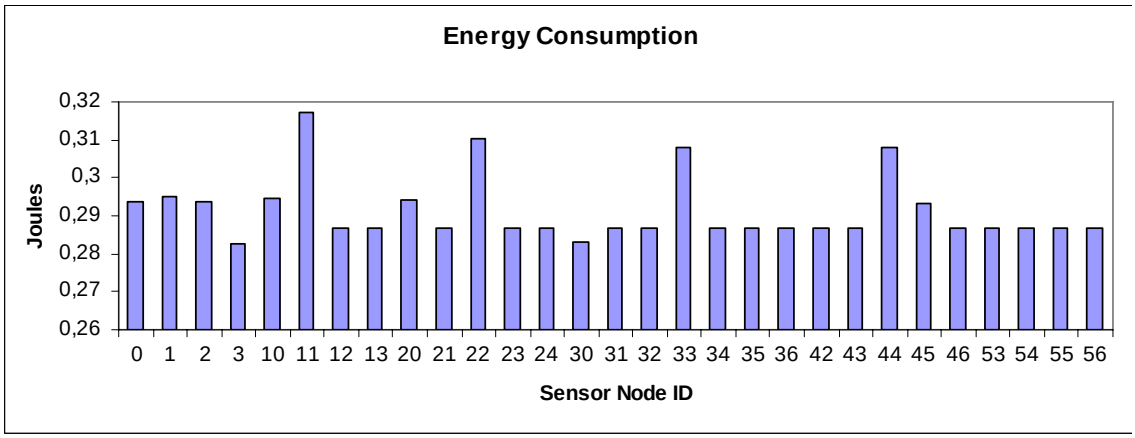


Figure 5-12 Energy Consumption Distribution

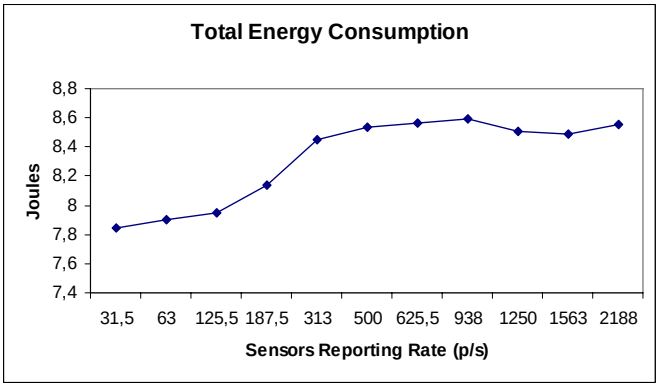


Figure 5-13 Total Energy Consumption

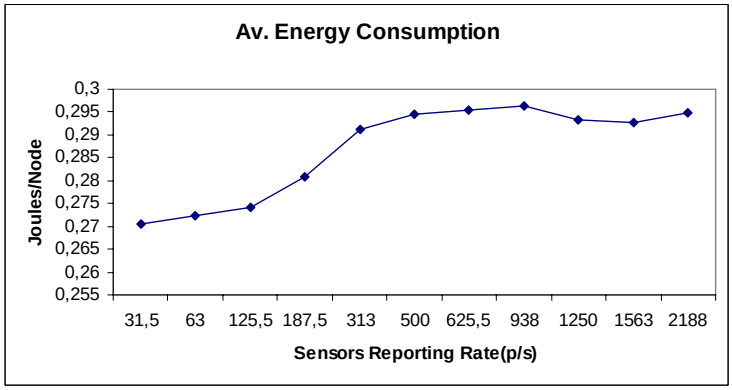


Figure 5-14 Average Energy Consumption

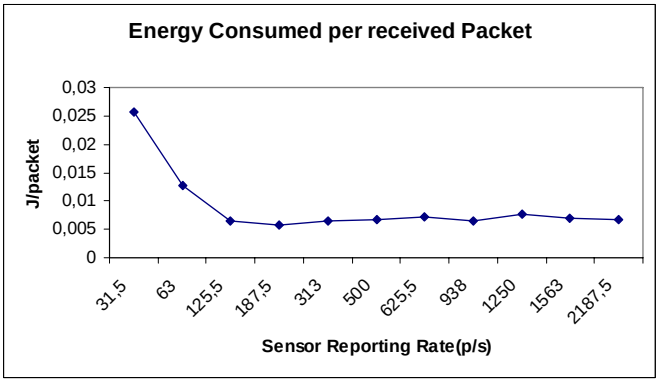


Figure 5-15 Energy Consumed Per Packet Received

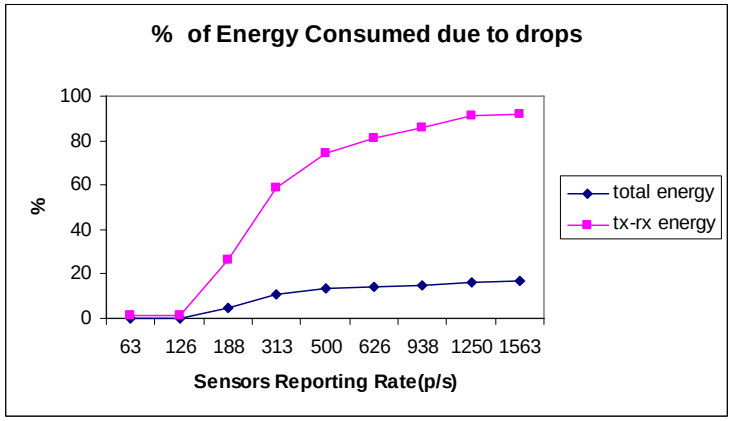


Figure 5-16 Energy Consumed on DROPS

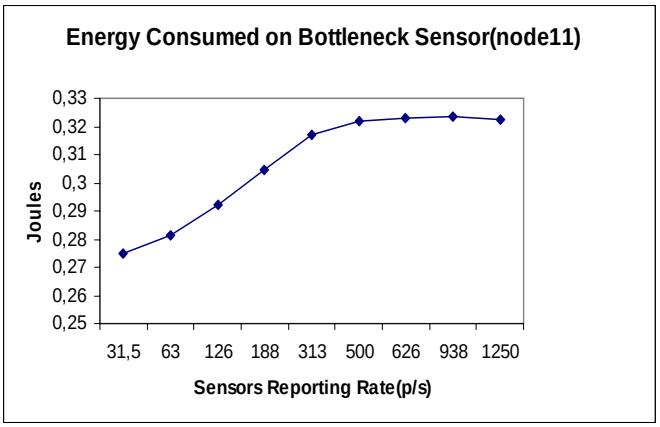


Figure 5-17 Energy Consumed on Bottleneck Node

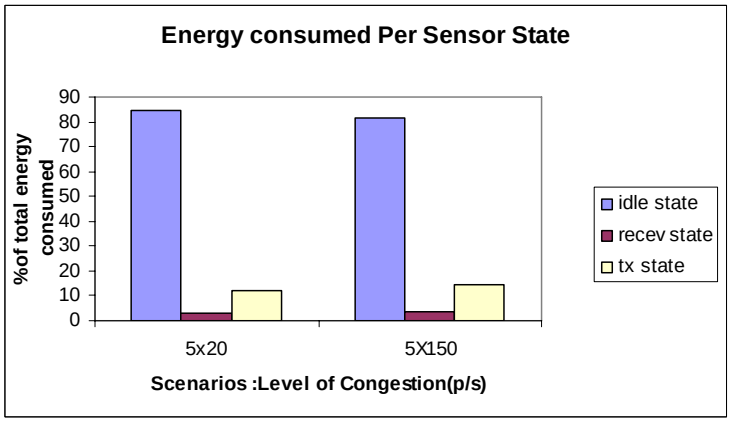


Figure 5-18 Energy Consumed per Sensor State

5.1.3.4 TEST 4

With this test we investigate one of the most well known congestion types in sensor networks – Near the Source Congestion. For that we used the BASELINE test and specifically the scenario with 5 sources sensing data simultaneously and generating data at a low rate of 20kbps each(100sensodrfield5x20.tcl).In the particular area around sources, 5 sources within the range of one another attempt to transmit simultaneously resulting in interference at the listening nodes and so to MAC and buffer overflows drops. Congestion analysis (Figure 5-19) gives the percentage of packet drops due to MAC and Buffer Overflows. It's obvious that in such situations MAC drops is an important amount of the congestion-related packet drops (about 50%). Figure 5-20 give the distribution of packet drops at several sensor nodes, showing that all sensing nodes have packet drops due to interference from each other(contention).

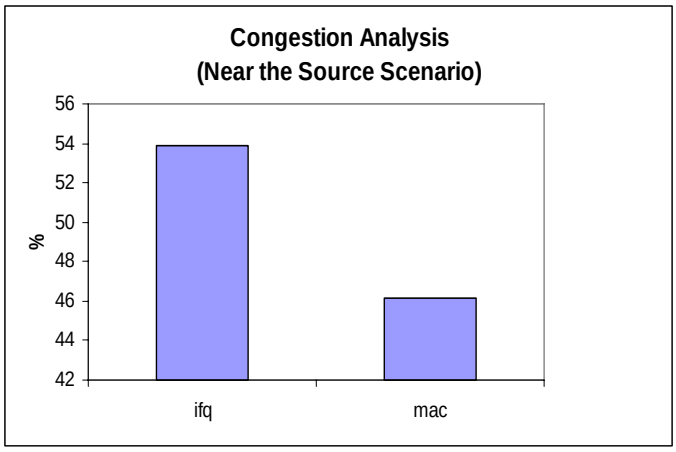


Figure 5-19 Near The Source Congestion Analysis

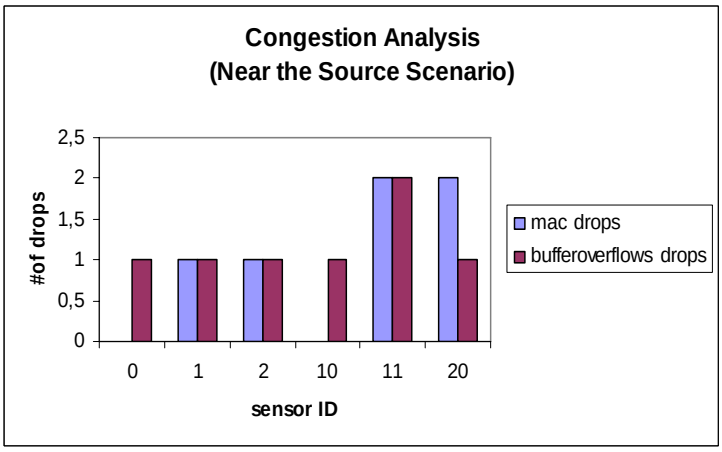


Figure 5-20 Near the Source Congestion Distribution

5.1.3.5 TEST 5

Here we provide an analysis of congestion problem when it occurs near the sink. We implemented such a scenario :100sensorfield2x80sink(Figure 5-21).Two sources are sending CBR traffic of 80kbps ,10sec, using the static routing protocol, through node 33. We observe from Figure 5-23 that congestion occurs and all the packet drops appear in an area around the sink(nodes 22, 33, 24).Both types of packet drops occurs (MAC and buffer overflows)as depicted in Figure 5-22.

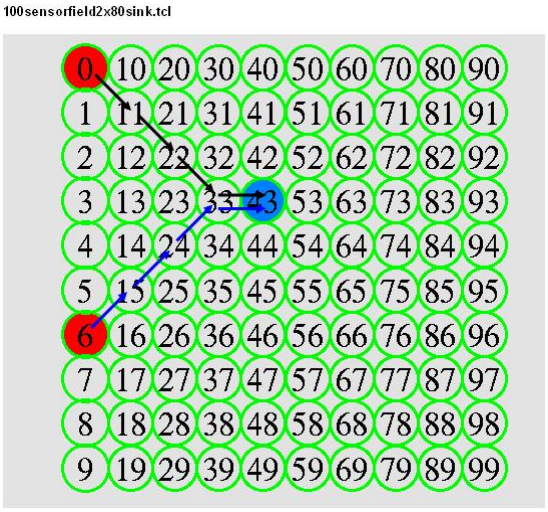


Figure 5-21 Near the Sink Topology

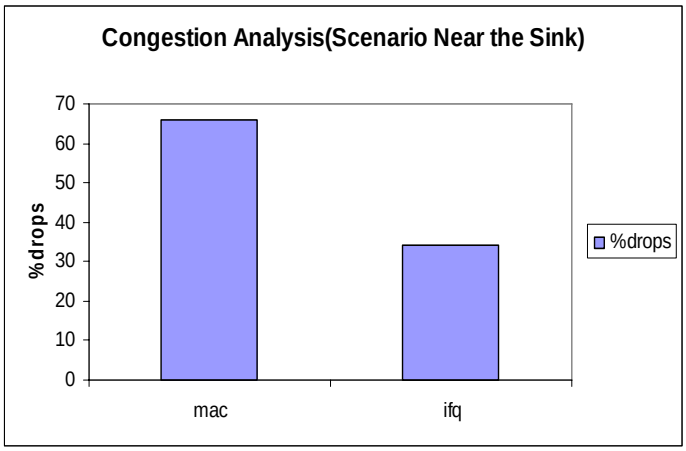


Figure 5-22 Near the Sink Congestion Analysis

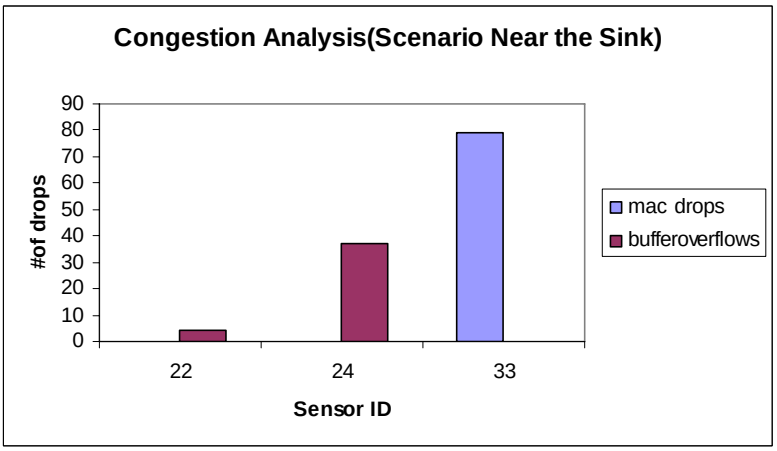


Figure 5-23 Near the Sink Congestion Distribution

5.1.3.6 TEST 6

One of the most common congestion problems occurs when the traffic from different areas in a sensor networks is merging. In the current test we simulated this phenomenon implementing the scenario : 100sensorfield2x80merg.tcl with the topology of Figure 5-24. Again CBR traffic is generated ,80kbps,(10sec) and static routing to forward the traffic to the sink. Nodes 4 and 40 are the sources and 94 is the sink.

From Figure 5-25 buffer overflows predominates packet drops (80%)and the area around merging point(Node 42, 24) is highly congested (Figure 5-26).

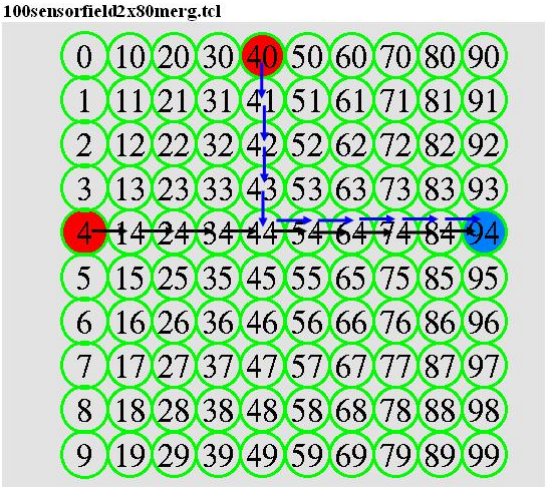


Figure 5-24 Merging Topology

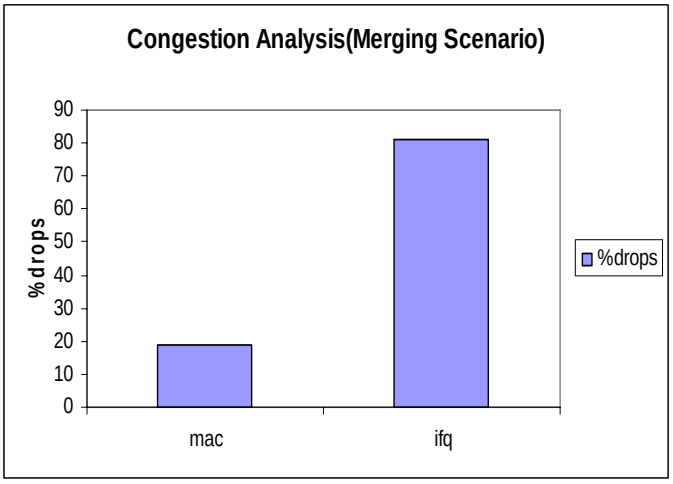


Figure 5-25 Merging Congestion Analysis

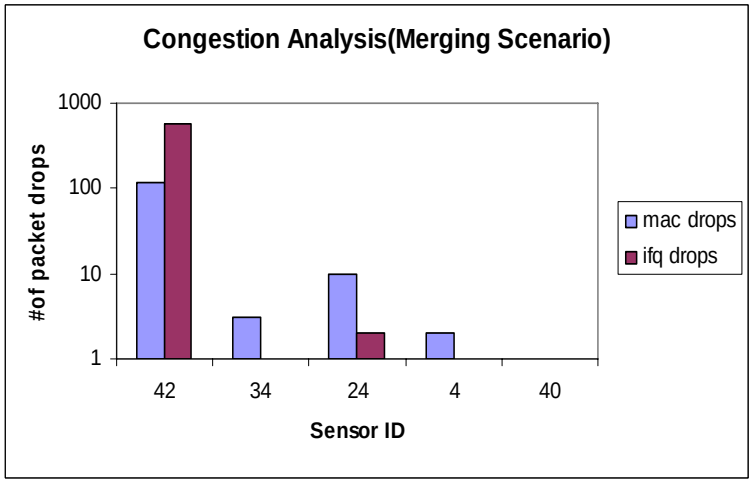


Figure 5-26 Merging Congestion Distribution

5.1.3.7 TEST 7

Some wireless sensor networks use more than 1 sink, either for load balancing or for practical reasons. This type of networks often observe congestion due to crossing traffic. In this test we investigate this kind of congestion problem with the scenario:100scensorfield2x80cross.According to the topology of Figure 5-27 sources 40 and 4 are sending traffic CBR traffic 80kbps for 10sec, using our' s static routing to sinks 49 and 94 respectively. From the results shown in Figures 5-28 and 5-29, buffer overflows again predominates packet drops and area around cross point is highly congested.

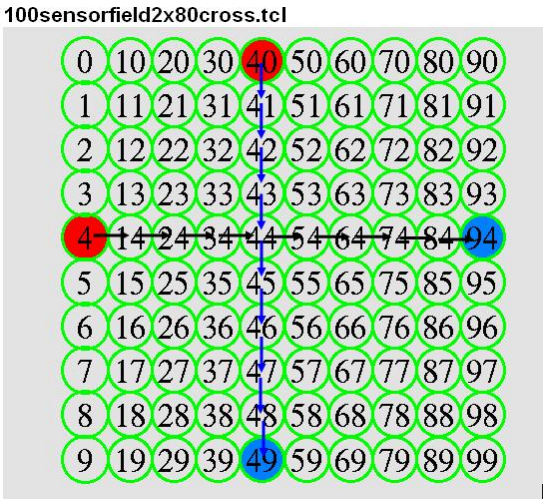


Figure 5-27 Cross Topology

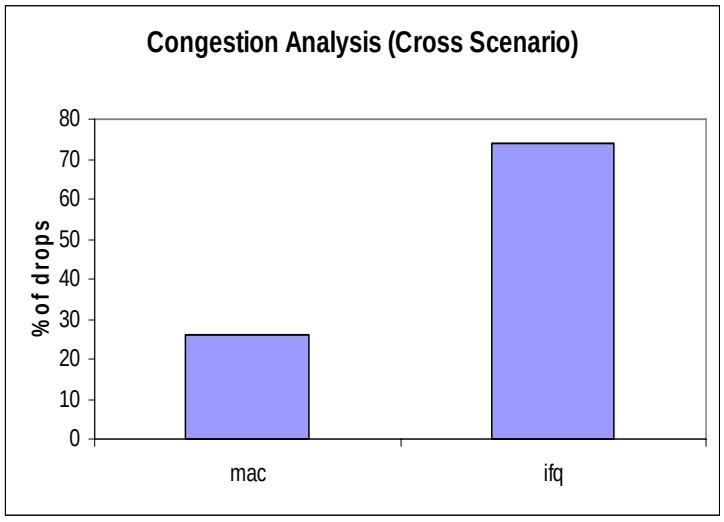


Figure 5-28 Cross Congestion Analysis

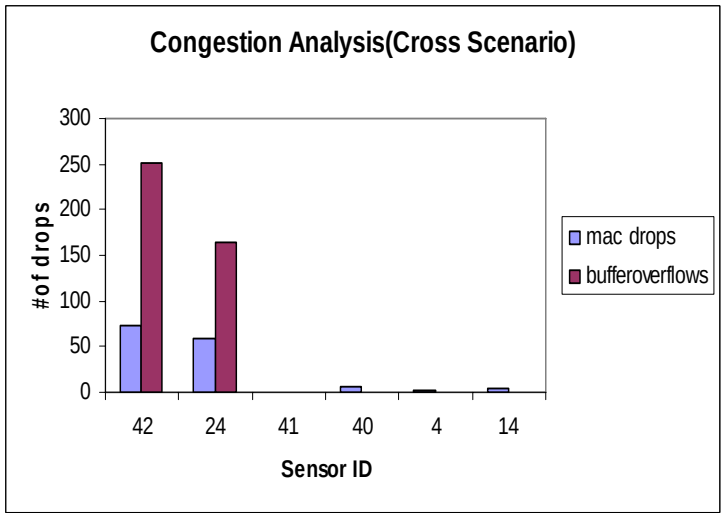


Figure 5-29 Cross Congestion Distribution

5.1.3.8 TEST 8

Number of traffic sources seems to be an important factor to the level of sensor network congestion. To examine the influence of number of sources to the creation of congestion we implemented several scenarios, based on BASELINE. Two different tests have been created for two different purposes:

a. Examine how the number of sources influence congestion when aggregate traffic rate is kept constant (100kbps).

In this case the scenarios have been created with 1,2,3,4,5 and 9 sources with 100kbps, 50kbps, 33.33 kbps, 25kbps, 20kbps and 11.11 kbps CBR traffic. Topology and routing is depicted in Figures 5-30 to 5-36.

Results in Figure 5-37 show that congestion problem as observed by reduced reliability is not affected to a large extend(~2%), rather the reliability is reduced very slowly when the number of sources is increased from 1 to 9 and contention is increased between the sources, causing more interference. That' s why , as shown in Figure 5-38, when more than 3 sources are sensing the same event at the same time, MAC drops and buffer overflows have almost the same impact on congestion.

b. Examine how the number of sources influence congestion when each sensor node generate traffic with the rate of 50kbps.

In this case things are different. Increasing the number of sources, automatically both contention and traffic load are increased much more, so there is a sharp decrease in reliability and so effect on congestion problem. In Figure 5-39 reliability has reduced more than 70% when there are 9 sources indicating a high effect on congestion. Also Figure 5-40 shows that buffer overflows predominates packet drops because traffic is too high and buffers are overflowed and packets are dropped before reaching the MAC Layer.

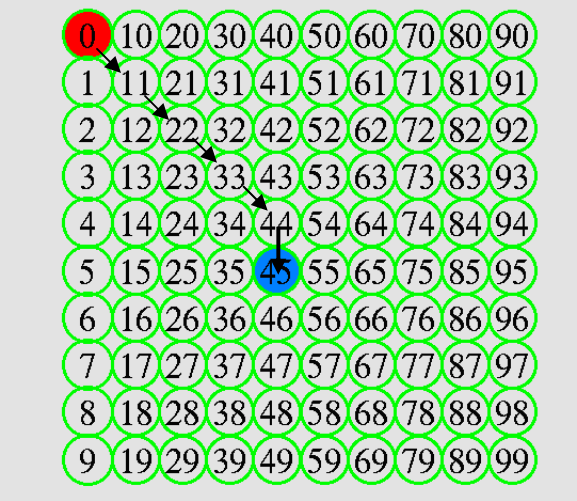


Figure 5-30 One Source Topology

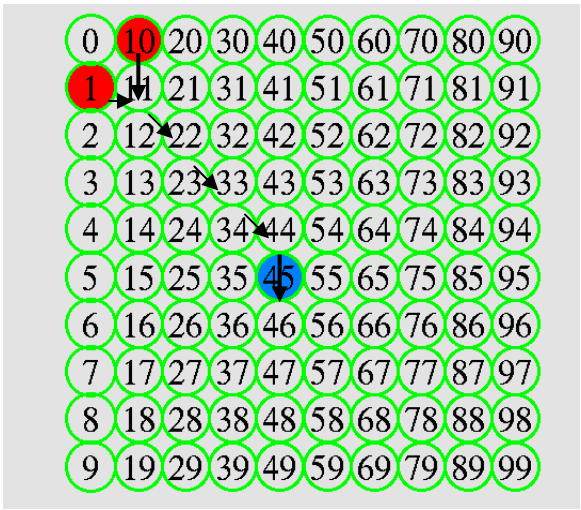


Figure 5-31 Two Sources Topology

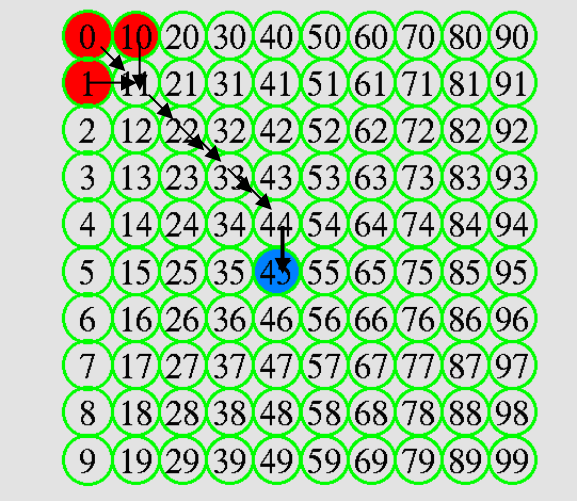


Figure 5-32 Three Sources Topology

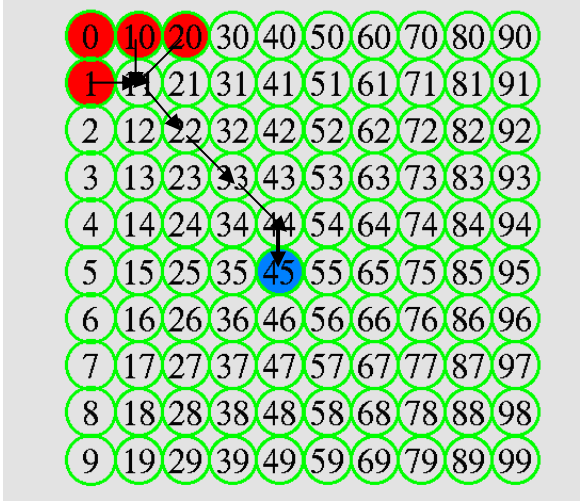


Figure 5-33 Four Sources Topology

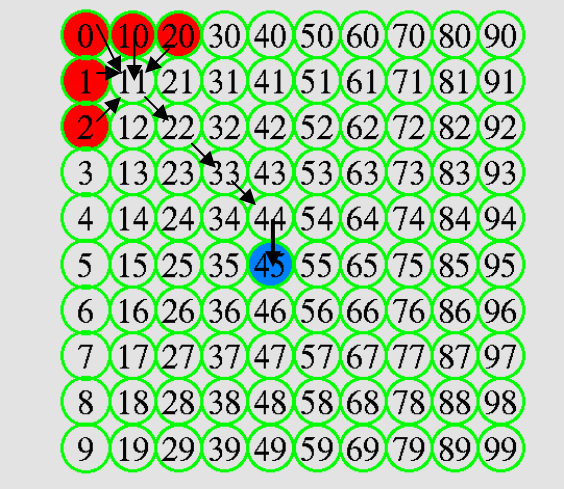


Figure 5-34 Five Sources Topology

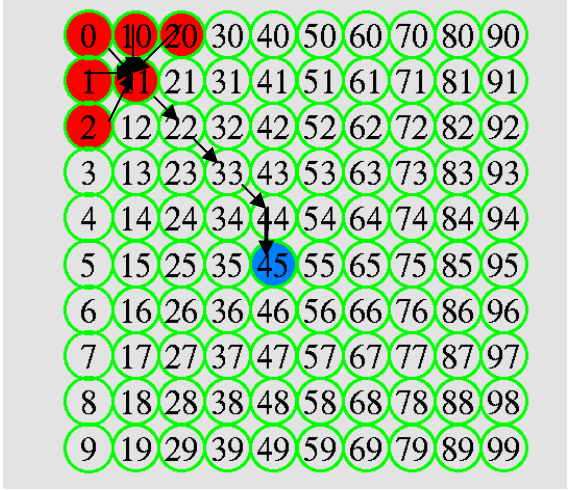


Figure 5-35 Six Sources Topology

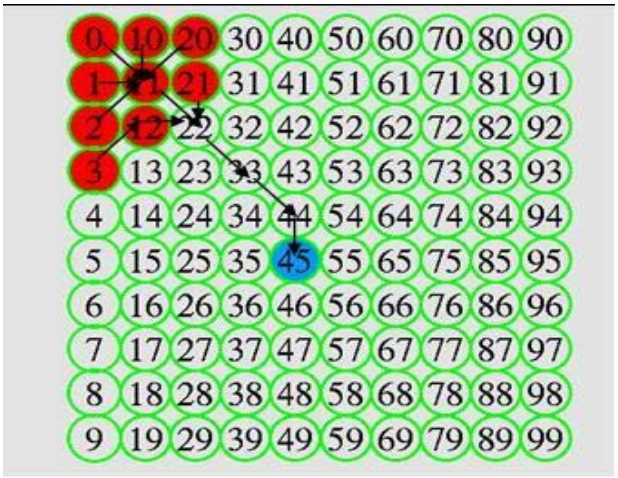


Figure 5-36 Nine Sources Topology

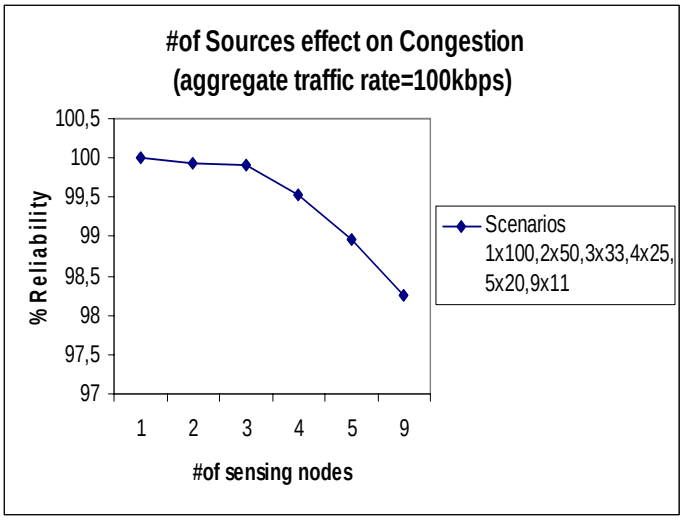


Figure 5-37 Reliability and number of Sources(100kps)

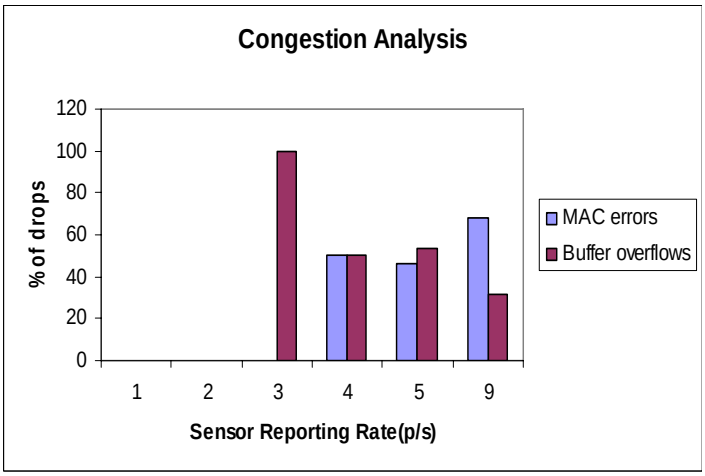


Figure 5-38 Congestion Analysis - number of Sources(100kbps)

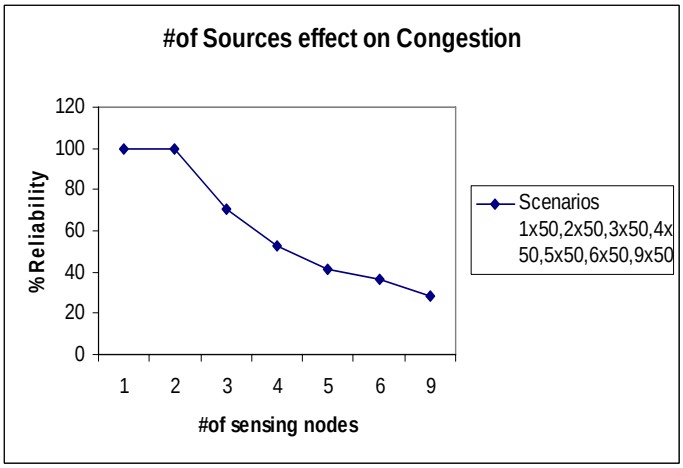


Figure 5-39 Reliability and number of Sources

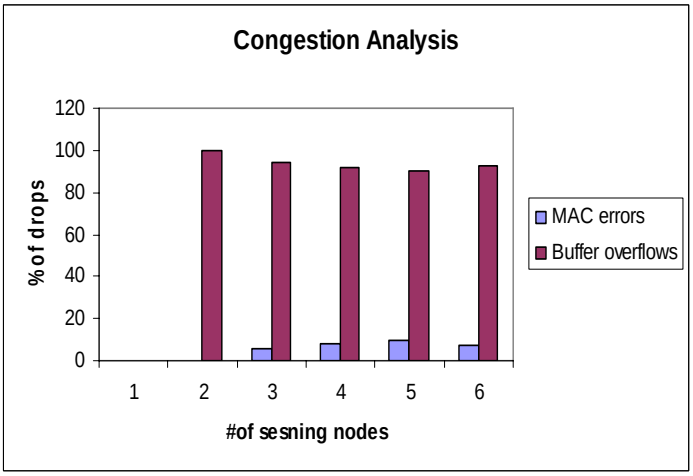


Figure 5-40 Congestion Analysis – number of Sources

5.1.3.9 TEST 9

During this test we investigate the effect of RTS/CTS mechanism on Congestion. Still using the BASELINE we have created the same scenarios except that we have disabled the RTS/CTS mechanism. Scenarios :10sensorfield5xX norts.tcl disabled the mechanism from tcl file .

Default value in NS-2 for RTSthreshold is 0 , and is set in ~tcl\lib\ns-default.tcl.

This setting is the size in bytes for a data packet in order to enable RTS/CTS control messages. If the data packet is larger than this threshold, RTS/CTS packets are sent too. In our simulations data packets size is set to 100 bytes. If we don't need to use RTS/CTS mechanism RTSThreshold must be set to more than 100 bytes eg 3000.

The results are compared with that of BASELINE counterparts where RTS/CTS mechanism is enabled.

As we can see from Figure 5-41 for reporting rate less than 125 p/s when congestion is mainly occurred due to MAC errors the use of RTS/CTS mechanism limit these errors as it minimizes hidden terminal problem. When the network is highly congested there is small influence on MAC errors degradation. But on the other hand (Figure 5-42)the inverse situation exceeds for buffer overflows. Using RTS/CTS packets more traffic is inserted in buffers that get full faster and number of packet drops due to buffer overflows increases.

When RTS/CTS mechanism is disabled, and reporting rate gets higher than 125 p/s where the buffer overflows start predominates, the little degradation on buffer overflows give better event reliability to our network. This is depicted in Figure 5-43 when beyond a rate of 200p/s there is a clear improvement of event reliability. Over and above average end-to end delay(Figure 5-44) is increased as RTS/CTS mechanism is enabled, as adds some overhead to the communication. This increase on delay is very important for delay intolerant sensor network applications that must choose to disable the mechanism. Briefly remind what rts/cts are needed for and the tradeoff for congestion control

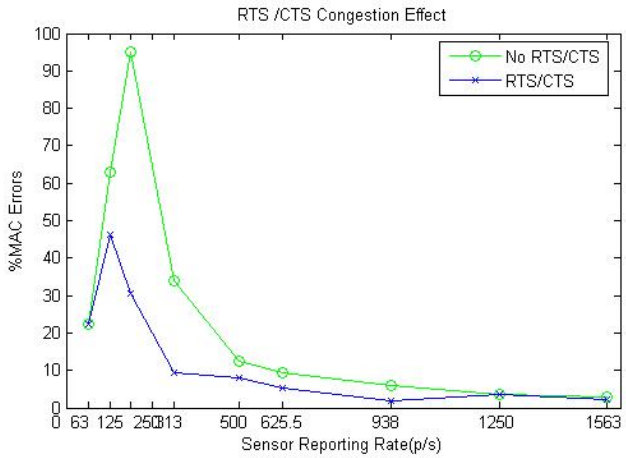


Figure 5-41 RTS/CTS MAC Errors

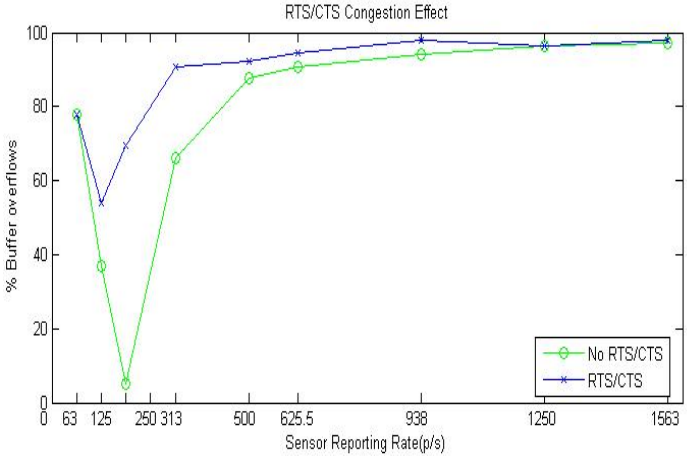


Figure 5-42 RTS/CTS Buffer Overflows

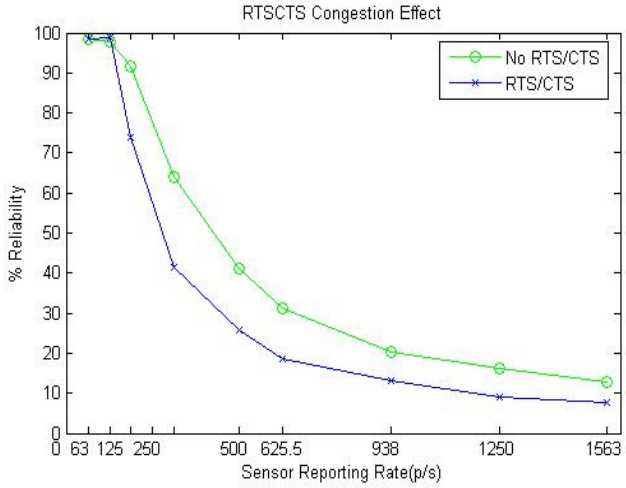


Figure 5-43 RTS/CTS Reliability

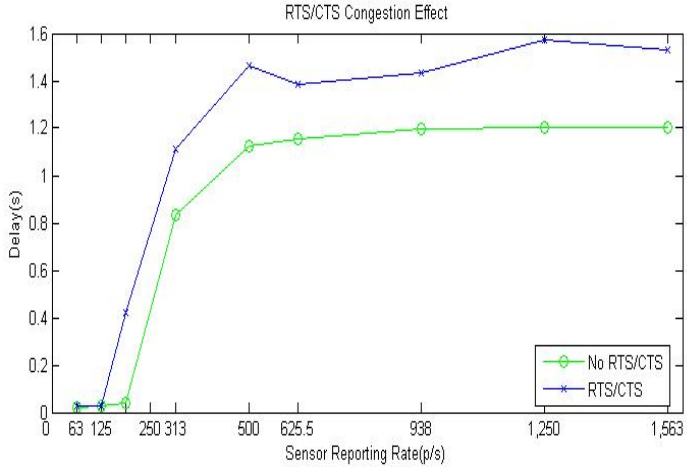


Figure 5-44 RTS/CTS End-to-End Delay

5.1.3.10 TEST 10

In this test we examine the effect of MAC reliability on Congestion in Sensor Networks and specifically the effect of maximum number of retransmissions. MAC 802.11 uses ARQ type of MAC Layer Reliability with default values maximum 4 retransmissions:

The ShortRetryLimit is specifying the maximum number of retransmissions for packets with size lower than that of RTSthreshold, and LongRetryLimit for size larger. In our

simulations when testing the effect of maximum number of retransmissions on congestion because the RTSthreshold is set to 0, the ShortRetryLimit has no effect. So we only set LongRetryLimit to 0,1,7,10 values.

We change the values to 0 ,1 ,7, 10 as the number of maximum retransmissions, implementing the scenarios: 100sensorfield5xXretY.tcl changing accordingly Y to 0,1,7,10.Other parameters are kept as that of BASELINE .

As we can see from Figures 5-45 and 5-46 when RTmax=4,7,10, has a positive effect to MAC errors for reporting rates below 250 p/s. At this situation decreases MAC errors for about 30% and increases buffer overflows errors to the same percentage(buffer drains slower due to retransmissions). When MAC Errors dominates other drops ,in this case for reporting rate lower than 150p/, the degradation on MAC errors gives a slight increase in reliability(Figure 5-47). But when the network get high congested RTmax=4,7,10 seems not to have any effect at all. This is because buffer overflows dominates(90%) irrespective of RTMax values. Consequently when the network capacity is highly exceeded, in addition to local reliability mechanisms, end to end congestion control and reliability mechanisms should be performed to improve event reliability. In Figure 5-48 the end to-end delay using RTmax =4,7,10 is larger when MAC errors dominates because as more packet failure occurs we have more retransmissions. This difference is decreased and saturate when buffer overflows dominate .Since these packets don’t reach MAC layer, the end to end latency is kept relatively constant.

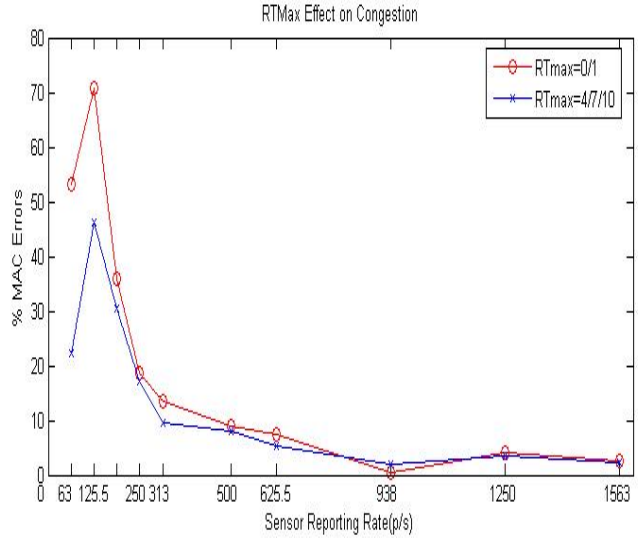


Figure 5-45 RTmax MAC Errors

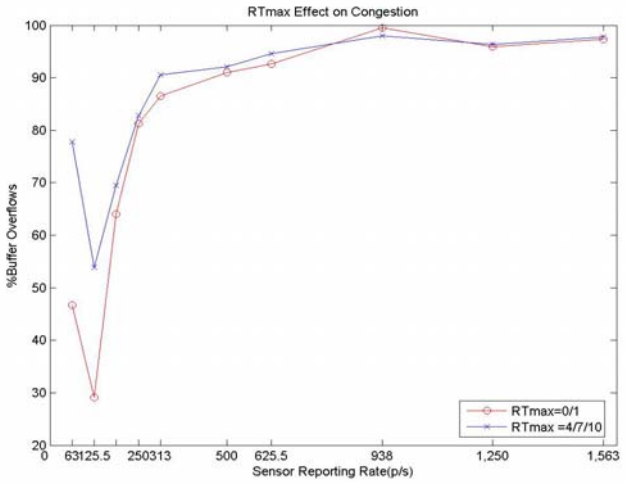


Figure 5-46 RTmax Buffer Overflows

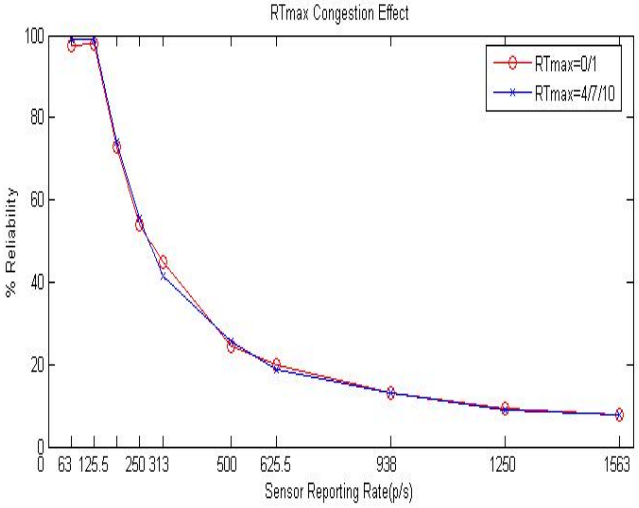


Figure 5-47 RTmax Reliability

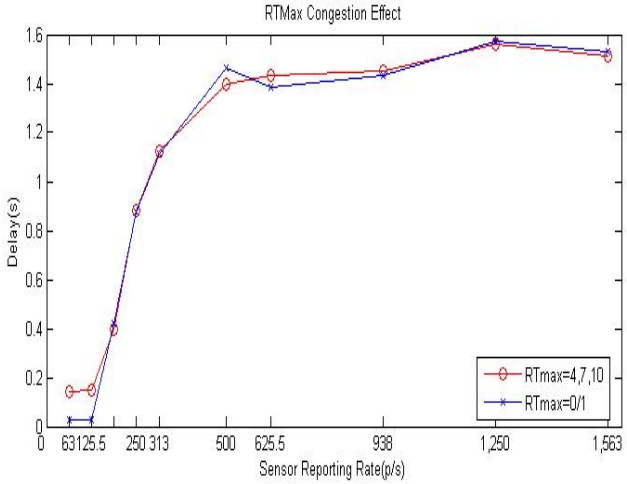


Figure 5-48 RTmax End-toEnd Delay

5.1.3.11 TEST 11

In this test , the impact of buffer size of the sensor node on the network congestion is investigated. At this test BASELINE scenarios are reproduced changing only the IFQ size (buffer).

This is done inside TCL file , just changing the value of ifqlength.

The scenarios implemented were: 100sensorfield5xXb5.tcl, 100sensorfield5xXb100.tcl , 100sensorfield5xXb200.tcl for buffer size =5,100 and 200 packets respectively. The results are also compared with BASELINE tests that uses 50 packets of buffer size.

Looking at the results increasing the buffer size has a negative effect on local contention and so to MAC errors. As shown in Figure 5-49 as buffer is increased the percentage of sent packets lost due to MAC errors increase. When the buffer size is small these packets are already dropped and are not passed to the MAC layer, leading to lower contention. On the other hand(Figure 5-50) decreasing buffer size leads to increment in buffer overflows, and as a result MAC errors decrease.

In Figure 5-51 we see that as the load increases and network is highly congested even for buffer sizes 200 packets, event reliability isn't increased as network wireless capacity is limited.

Another interesting result is that depicted in Figure 5-52, where end-to-end delay observed is very low when reporting rate is low (below 125p/s).For higher reporting rates when increasing the buffer size, this delay increases significantly as queuing delay increases too.

All the above led to very interesting conclusions. For applications that end- to- end delay is very important and when reliability can be afforded to be(95-98%) at maximum, lower buffer can be selected. This is contradictory to the conventional belief that the limited storage capabilities of sensor nodes always leads to problems.

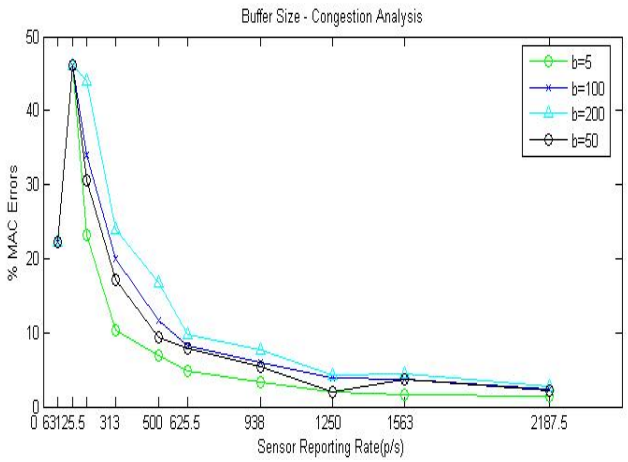


Figure 5-49 Buffer Size MAC Errors

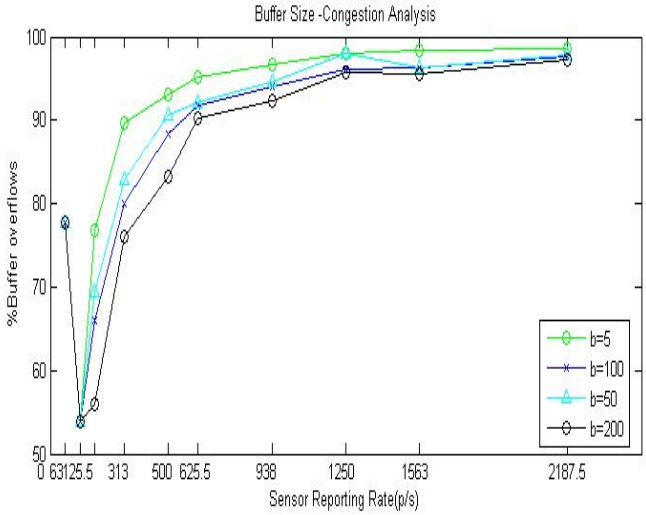


Figure 5-50 Buffer Size Buffer Overflows

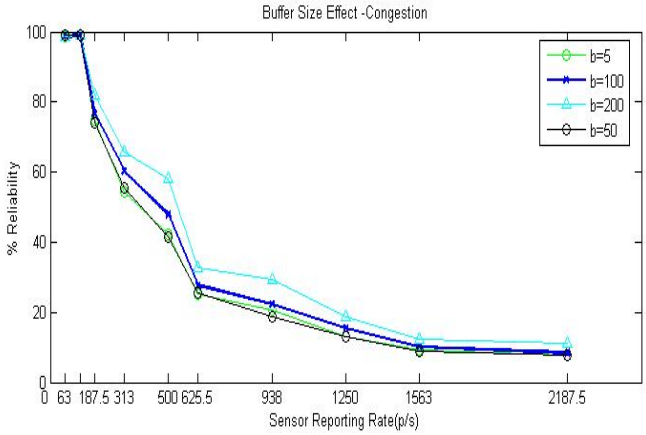


Figure 5-51 Buffer Size Reliability

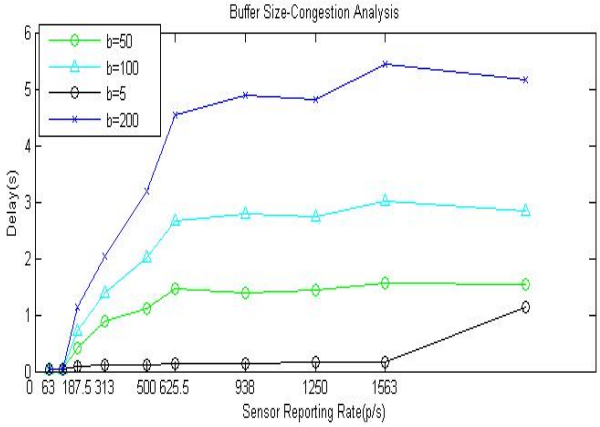


Figure 5-52 Buffer Size End-to-End Delay

5.1.3.12 TEST 12

In contention –based MAC protocols, the contention resolution mechanism is performed via contention window adjustments. Each node adjust its random backoff which is selected randomly between (o,cw). The initial size of window is set to Initial Contention Window size(CWmin). Now we investigate its effect on congestion. We changed it 128 in simulation scenarios 100sensorfield5xXcw128.tcl keeping other simulation parameters as in BASELINE scenarios. Default value for CWmin is 32 and is again set in ~tcl\lib\ns-default.tcl

It can be changed from TCL file.

The effect of CWmin on MAC errors and buffer overflows are depicted on Figures 5-54 and 5-55 respectively. As seen large CWmin has positive effect on MAC layer errors(again around 125,5 p/s) whilst the opposite occurs for Buffer overflows.

It is obvious from the results of Figure 5-53 that the difference in reliability increases as the reporting rate is increasing in low to medium reporting rate region. This is due to the unnecessary long contention window size at this region. In our scenarios seems that Cwmin =32 have good resolution impact to contention and that larger CWmin causes more congestion in low –to medium reporting rates with more packet drops occurred on buffer overflows, whilst in high reporting rates(high congested region, >938p/s) when buffer overflows pre- dominates, no significant influence at network performance occurs.

It is obvious that for networks that contention is too much higher, better reliability could occurs in congested networks regions as better contention resolution could help in faster draining of buffer nodes, and so less congestion.

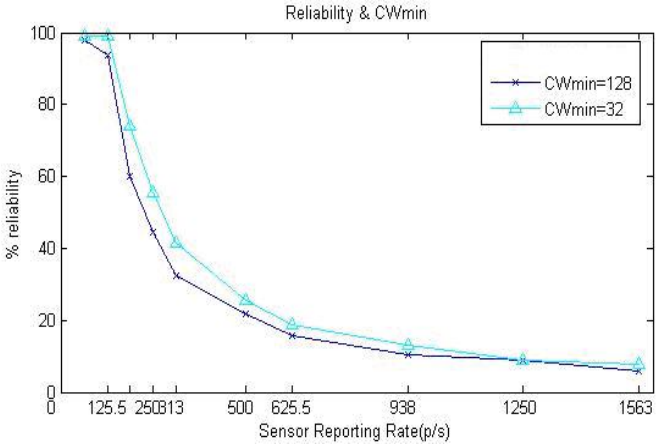


Figure 5-53 CWmin Reliability

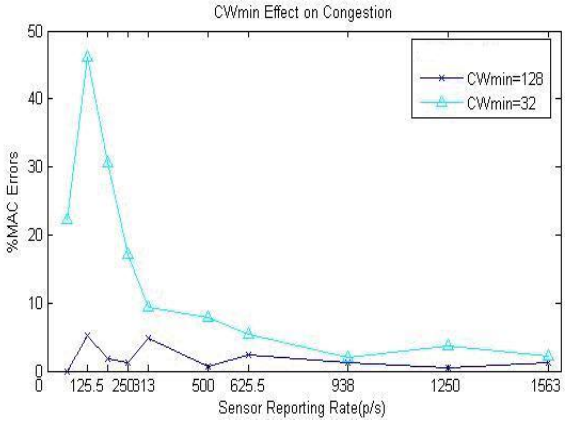


Figure 5-54 CWmin MAC Errors

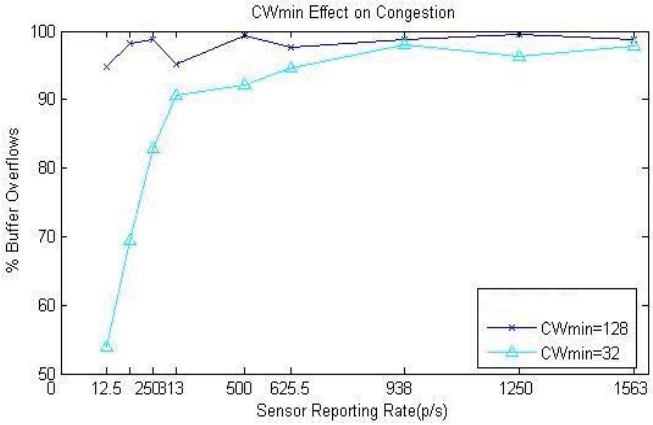


Figure 5-55 CWmin Buffer Overflows

5.1.3.13 TEST 13

Our scope in this test is to examine the effect the average Path Length to the sink has in the congestion problem. We reproduced BASELINE test and six scenarios :100sensorfield5x50_X.tcl are implemented changing each time the network sink(**X**) to 11,22,33,44,77,99 respectively and CBR traffic is kept fixed to50kbps. Static routing is again used as depicted in Figure 5-56 according to the specific sink.

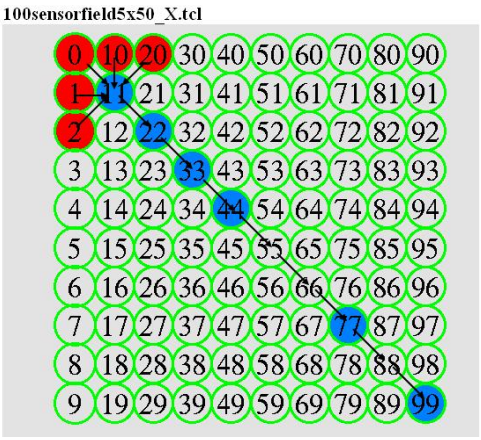


Figure 5-56 Average Path Topology

Using again the reliability as Congestion metric(as all packet drops are occurred due to congestion) according to Figure 5-57 we see that as average path length is increasing congestion gets worst. A sharp decrease in reliability occurs when average path length is is increasing from 1 to 3hops. This is related with the interference factor which is 3 in this test. When the average path length is more than 3 hops reliability and so congestion is slowly affected. Decreasing Reliability is explained from the fact that the end to end capacity is reduced as path to the sink is increased. It is important to note that putting more sinks in the wireless sensor network you can control the effect of average path length to congestion as if local communication predominates there is some kind of guarantee of minimum end –to-end end capacity. From the results of Fig 5-58 we can also see that as average path length increase Buffer Overflows predominates, while MAC drops are minimized to a specific value.

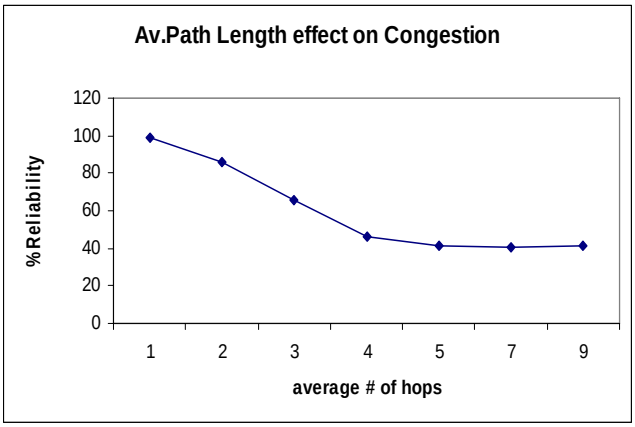


Figure 5-57 Average Path Reliability

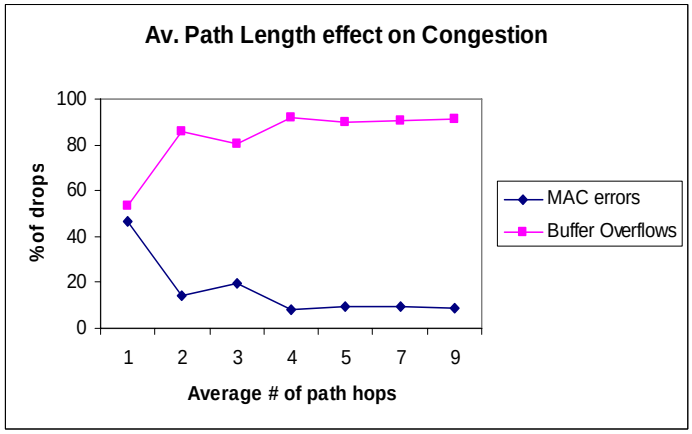


Figure 5-58 Average Path Congestion Analysis

5.1.3.14 TEST 14

So far we have investigated how several network parameters affect the congestion problem using a static routing protocol in order to eliminate any influences of the ad-hoc routing protocols.

At this test we examine how Ad hoc routing protocols affect Congestion. Surely a routing protocol greatly affects packet loss. All properties of a routing protocol, such as what routing information is maintained: the way in which the information is obtained, how to choose a route, etc.: may have different effects. The simulations are conducted by using DSDV and AODV routing protocols as most well known representatives of proactive and reactive ad –hoc routing protocols widely used in wireless sensor networks.

DSDV and AODV setup of parameters(Table 5-5 and 5-6 respectively) have been adopted from the work **[[46]]**. Modifications took place in representative files in NS-2 simulator.

DSDV		
PARAMETER	VALUE	FILE OF MODIFICATIONS
Periodic route update interval	15s	dsdv.h
Periodic updates missed before link declared broken	3	
Route advertisement aggregation time	1s	
Maximum packets buffered per node per destination	5	

Table 5-5 DSDV setup Parameters

AODV		
PARAMETER	VALUE	FILE OF MODIFICATIONS
Hello interval	1,5s	aodv.h
Active Route time-out	300s	
Route reply lifetime	300s	aodv_rqueue.h
Allowed HELLO loss	2	
Request retries	3	
Time between retransmitted requests	3s	
Time to hold packets awaiting routes	8s	
Maximum rate for sending replies for a route	1/s	

Table 5-6 AODV setup Parameters

In this test we firstly differentiate packet losses to mobility related(routing setup related) and congestion related .

From Table 5-4 Routing Setup Related packet losses are those associated with routing Layer.

TEST 14a.

In this test we investigate the effect of DSDV and AODV on Congestion Problem. We use BASELINE scenarios changing the routing agent to AODV and DSDV accordingly in TCL file.

```
set val(rp)          AODVorDSDV                ;# routing protocol
```

The scenarios implemented are 100sensorfield5xXaodv.tcl and 100sensorfield5xXdsdv.tcl using CBR traffic and values of BASELINE scenarios.

From Figure 5.59 we can obtain that for DSDV which is proactive routing protocol(sensor nodes are static) all the drops are congestion related. On the other hand reactive routing protocol AODV, suffer also from routing setup drops. These drops are decreased as offered load is increased while congestion related are increased.

As seen from Figure 5-60 using of DSDV has larger effect than AODV on Congestion, especially on high offered load. There is a difference of 10% on average.

Looking on the congestion drops distribution(Figure 5-61) (average # of drops of all scenarios of both routing protocols) we find out that packet drops refer to a small number of specific Sensor Nodes(1,2,3,10,11,12,20,21).As these routing protocols try to find to shortest path to the sink, they concentrate the traffic creating congestion.

The converged traffic load exceeds the capacity of those hosts. The difference of performance of these two protocols to congestion, may result from, with a very great chance, to the different route maintenance schemes used by DSDV and AODV. Both protocols use distance vector to represent routing information and choose the routes based on the shortest paths. However DSDV requires periodical updates of routing information. Every host has the most recent knowledge about routes. It is likely that the path chosen to forward packets is the currently shortest one. In contrast to DSDV AODV

picks up a path (usually the shortest one) when a host initiates a route discovery. The host keeps sending packets via this path until it breaks, even if shorter paths become available after route discovery. So much more traffic concentration happens with DSDV.

TEST 14b.

In this test we investigated the effect of initialization traffic on AODV routing protocol. Using the previous scenarios we inserted a kind of initialization traffic of 1kbps CBR at the starting of the simulation (5-10sec)and examined what influence this traffic could have in the Congestion Problem. Scenarios :100sensorfield5xXaodvinit.tcl has been implemented.

From the Figure 5-62 and 5-63 we see that initial traffic has too small positive effect on congestion problem. Only routing –setup related drops are mainly decreased. Although in very low(below 63 p/s) reporting rates congestion drops are also minimized.

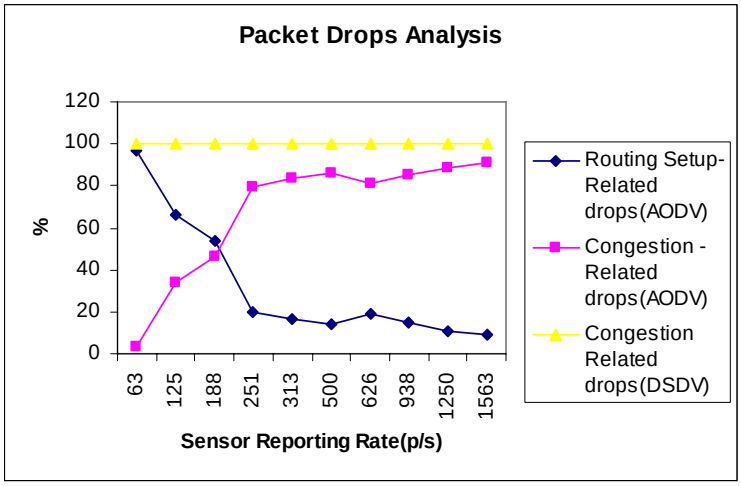


Figure 5-59 Ad-Hoc Routing Packet Drops

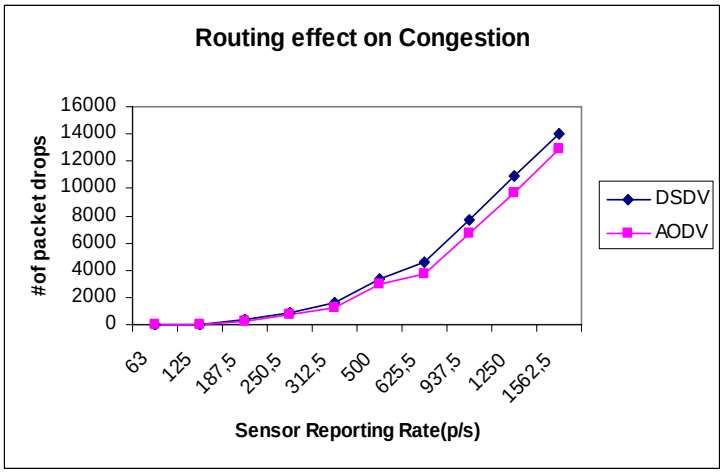


Figure 5-60 Ad Hoc Routing Congestion Analysis

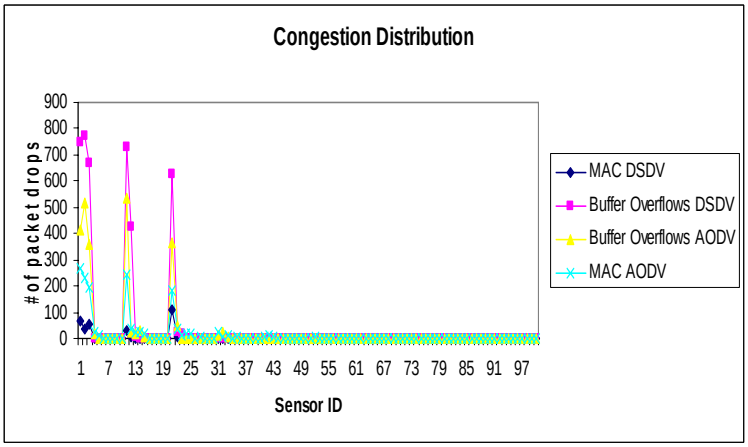


Figure 5-61 Ad Hoc Routing Congestion Distribution

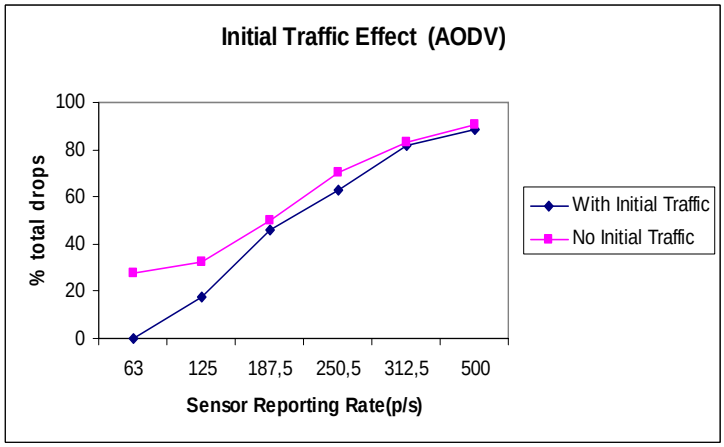


Figure 5-62 AODV Initial Traffic Effect on Packet Drops

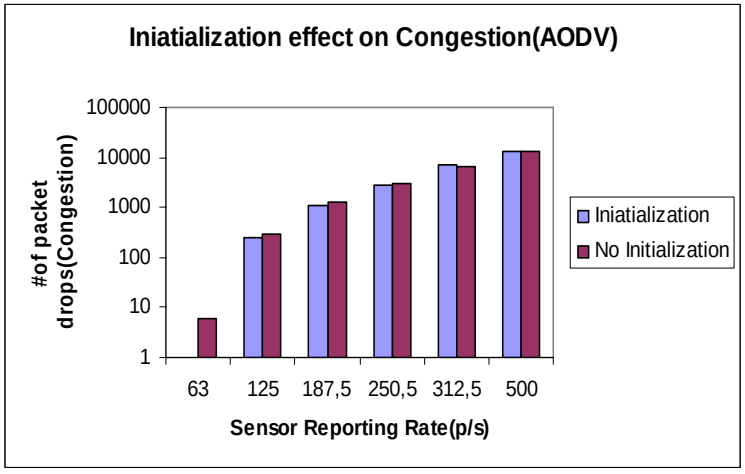


Figure 5-63 AODV Initial Traffic Effect on Congestion

5.1.4 Simulation Scenarios Section B(Cases of study)

5.1.4.1 Source Congestion Solution

In this type of scenarios we examine the influence of an introduced random transmission delay in the application layer of each source in order to decrease the influence of simultaneous transmissions. We include this method in resource control as we control the number of sources that transmit together, so we control time of transmission and not the rate sensor nodes transmit.(resources: active nodes at time)

As was seen from previous simulations and results one problem that causes congestion in wireless sensor networks, is the number of simultaneous sources that sensing the same event . This results in congestion –related packet losses near the sources either as MAC errors or as Buffer Overflows. The problem could be controlled in the phase of good planning as the peak values of traffic could be estimated from before. Various solutions can be followed toward this philosophy but in our scenarios we investigate the effect of a random transmission delay introduced in application layer. This delay introduce a time spacing on simultaneous generation of sensing data. The network topology is that of BASELINE scenario. Results are obtained under different traffic rates in the case all the five sources are transmitting simultaneously.

5.1.4.1.1 TEST 1

First we want to examine the effect the insertion of a transmission random delay has, which is specified from 0 to 1 sec. This delay is created using TCL **language function rand().**

Source Congestion/TEST1		
Scenarios	values	Notes
100sensorfield5x**X**rand.tcl	0-1s	**X** is the rate of traffic generated from each source with values 10,30,50,80,100,150,250kbps

Table 5-7 Source Congestion TEST 1

Using above scenarios(Table 5-7) we examined the effect of this delay to congestion. The results are compared with results of BASELINE scenarios: 100sensorfield5x**X**.tcl, that were implemented without delay.

The results shown in of Figure 5-64, show that the introduction of this delay has a positive effect (still small) on congestion related packet drops. In all congested scenarios (beyond 125,5p/s) drops decrease, reliability is increasing(Figure 5-65) and average end-to end delay is also decreased(Figure 5-66). The last observation show that even when we introduce a random transmission delay, average end-to-end delay is decreased due to the decrease in MAC contention and buffer queue delays. We can also observe that effect of this delay is larger when the aggregate reporting rate is between 313 and 1563 p/s since congestion is larger in this region .

Thus the source Congestion can be eliminated by careful scheduling between these sources.

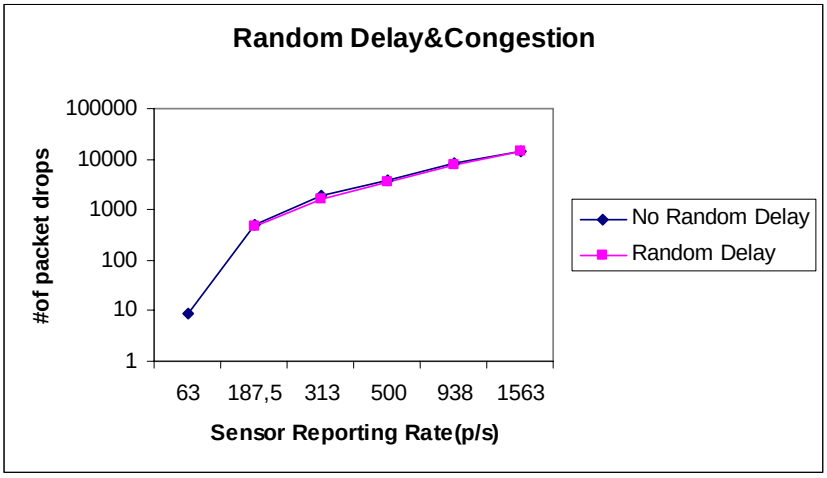


Figure 5-64 Random Transmission Delay Effect on Congestion

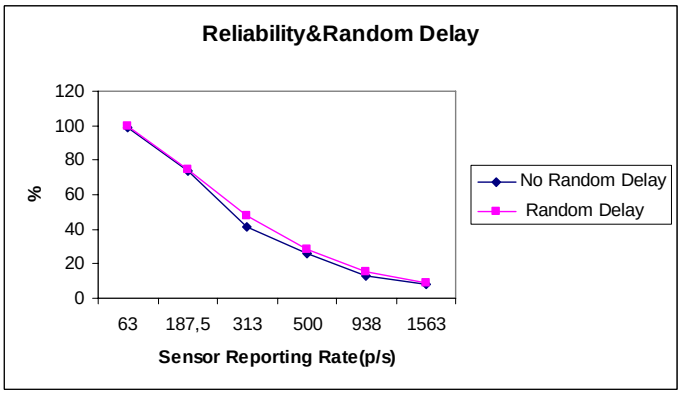


Figure 5-65 Random Transmission Delay & Reliability

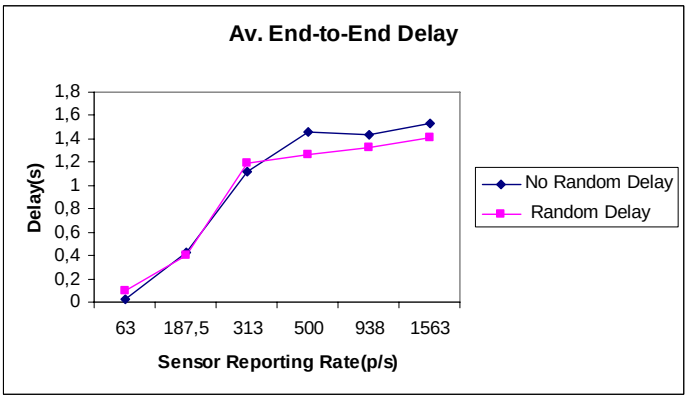


Figure 5-66 Random Transmission Delay & Av. End-to-End Delay

5.1.4.1.2 TEST 2

Next we examine the impact of random transmission delay duration to congestion. In this case the results have been taken for the high congested scenarios:100sensorfield5x150rand0-5.tcl and 100sensorfield5x150rand0-10.tcl.The results were compared with 100sensorfield5x150rand.tcl

Source Congestion/TEST 2		
Scenarios	values	Notes
100sensorfield5x150rand.tcl	0-1s	Duration is min 0sec, max 1sec
100sensorfield5x150rand0-5.tcl	0-5s	Duration is min 0sec, max 5sec
100sensorfield5x150rand0-10.tcl	0-10s	Duration is min 0sec, max 10sec

Table 5-8 Source Congestion TEST2

By comparing these 3 cases(Table 5-8) we led to very important conclusions. Firstly in reliability graph (Figure 5-67) the reliability is increased when increasing the random transmission delay duration .This increase is about 10% .On the other hand the increase in random transmission delay duration comes with a decrease in average end-to-end delay of 0,2 sec due to the elimination of congestion. This is depicted in Figure 5-68.

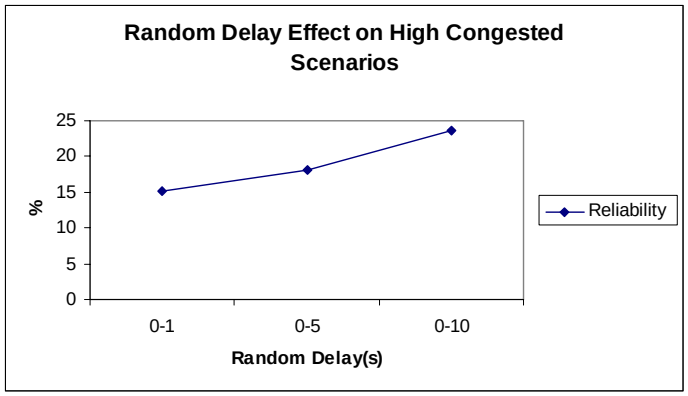


Figure 5-67 Random Transmission Delay Duration & Reliability

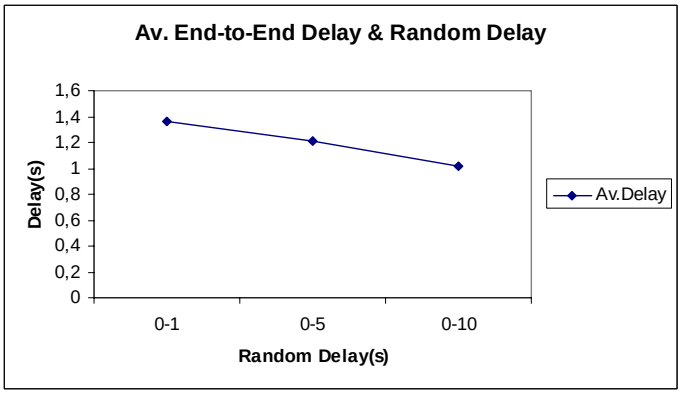


Figure 5-68 Random Transmission Delay Duration & Av.End-to-End Delay

5.1.4.2 Sink Congestion Solution

In this type of simple scenarios the effect of number and placement of sinks on congestion control is investigated. As we have seen in Chapter 3 this type of congestion create a hot spot near the sink where a lot of packets are dropped .A possible effective way of alleviating sink congestion could be the addition of multiple sinks uniformly scattered or positioned in specific places and the balance the traffic between the sinks. This is also possible in the phase of network planning.

5.1.4.2.1 TEST 1

Using the near the sink scenario :100sensorfield2x80sink.tcl as the base of this work, we created two more scenarios and added another second sink. Topology and network environment are depicted in Table 5-9 and Figure 5-69.

Sink Congestion/TEST1		
Scenarios	Sink nodes	Notes
100sensorfield2x80sink.tcl	43	**2 sources CBR at 80kbps**
100sensorfield2x80sinknode40.tcl	40,43	
100sensorfield2x80sinknode42.tcl	42,43	

Table 5-9 Sink Congestion TEST 1

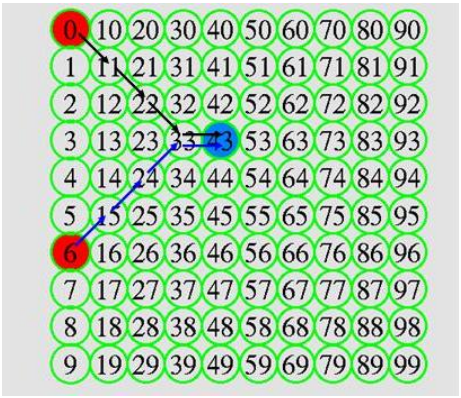


Figure 5-69 Near the Sink Topology1

Figure 5-70 Near the Sink Topology2

Figure 5-71 Near the Sink Topology3

Results are as expected. If we add more sinks and balance the load accordingly it is shown to be beneficial for network power to resolve congestion.. Of course no good planning and incorrect positioning of sinks may have the opposite results. As can be seen from Figure 5-72 reliability is only increased when the added sink/or sinks is not placed in the congested area and interference range with other flows. This is succeeded with the topology of Figure 5-70, whilst in topology of Figure 5-71 congestion problem worsen.

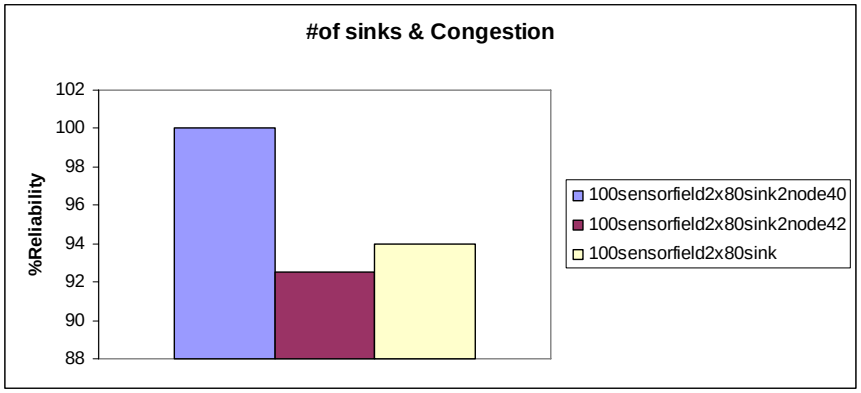


Figure 5-72 Number of Sinks Effect on Congestion

5.1.4.3 Forwarder Congestion Solution

In this type of scenarios we examine the effect of rerouting and multi-path routing in wireless sensor networks as a solution to congestion. It is important to note that all simulations suppose offline rerouting or multi-path routing because the algorithm 2 of our's framework is not implemented in NS-2.Specifically all the created scenarios with corresponding investigation scope are presented in Table 5-10:

Scenarios Rerouting /Multi-path Routing Summary			
Scenarios	Investigation	Notes	Test
100sensorfield3x**X**rout.tcl 100sensorfield3x**X**rerout.tcl	Congestion Alleviation using Rerouting - Energy Consumption	**X** is traffic rate(CBR) used in different scenarios taking values:10,20,30,40,50,80, 90,100Kbps / 3 Sources	**1**
100sensorfieldrout3**X**.tcl 100sensorfieldrerout3**X**.tcl	Comparison of Topology Aware and Dumb Rerouting effect on Congestion - Energy Consumption	**X** is traffic rate(CBR) used in different scenarios taking alues:30,50,70,80,90 / 3 Sources	**2**
100sensorfield**Y**route2x**X**.tcl	Congestion Alleviation using Multi-path Routing , Energy Consumption, Route Selection	Y=1,2,3, is the number of routes created **X** is traffic rate(CBR) used in different scenarios taking values:	**3**

Scenarios Rerouting /Multi-path Routing Summary			
Scenarios	Investigation	Notes	Test
		10,30,50,60,70,90,100 / 2 Sources	
100sensorfield**2**route2x**X**.tcl	New Route Selection Effect on Congestion Alleviation	**X** is traffic rate(CBR) used in different scenarios taking values:70,90/ 2 Sources	**4**
100sensorfield**Y**route2x**X100-0**.tcl	Traffic Distribution Effect on Congestion Alleviation	100-0 , 100% on detour path, 0% origin path,X=70,90/ 2 Sources	**5**
100sensorfield**Y**route2x**X80-20**.tcl		80-20 , 80% on detour path, 20% origin path X=70,90/ 2 Sources	
100sensorfield**2**route2x**X60-40**.tcl		60-40 , 60% on detour path, 40% origin path X=70,90/ 2 Sources	
100sensorfield**2**route2x**X66-33**.tcl		66-33 , 66% on detour path, 33% origin path X=70,90/ 2 Sources	
100sensorfield**2**route2x**X**.tcl		50-50 , 50% on detour path, 50% origin path X=70,90/ 2 Sources	
****In all scenarios parameters that are not explicitly defined are setup as BASELINE values ****			

Table 5-10 Scenarios Rerouting /Multi-path Routing Summary

5.1.4.3.1 TEST 1

For the first set of tests we followed the topology of Figure 5-73. Rerouting is depicted with dash line and is created in order to face congestion problem around node 34. Results are taken for different traffic source rates to show the behavior of rerouting in various levels of congestion.

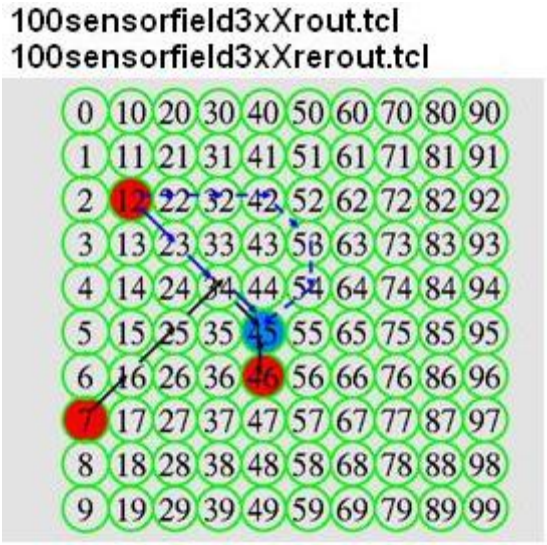


Figure 5-73 Rerouting Topology

Hereafter, we present the results of simulations:

From the Figure 5-74 it is important to mention that for an application at the sink with required fidelity of 2250 packets in the event time, only rerouting(resource control strategy)is able to succeed, without creating congestion problem. An ideal traffic control method would reduce traffic at the aggregate rate of about 225 packets/s and never would approach the required fidelity of 2250 packets. That is the grade importance of rerouting. That way required fidelity is always met.

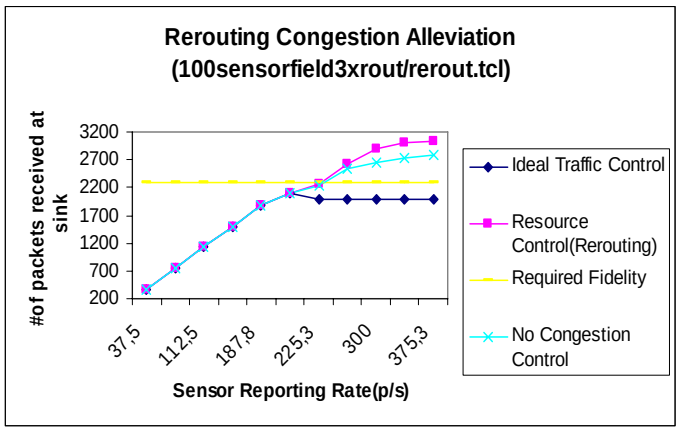


Figure 5-74 Rerouting Effect on Congestion

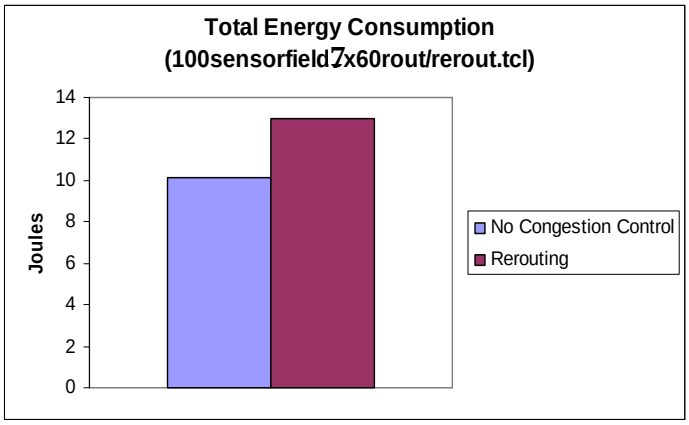


Figure 5-75 Rerouting -Total Energy Consumption

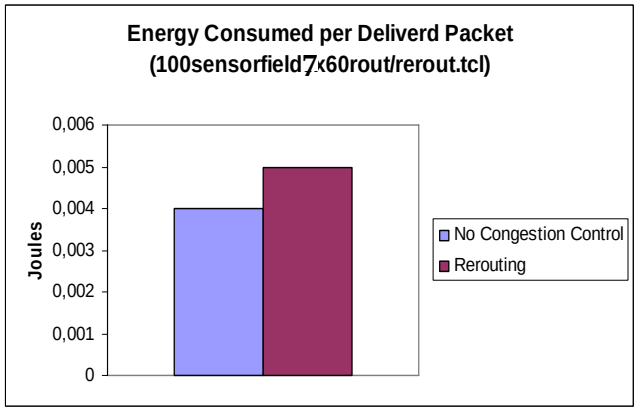


Figure 5-76 Rerouting -Energy Consumed per delivered Packet

It is very interesting to show the impact of rerouting to energy consumption. As more nodes are forwarding traffic more energy is consumed. For this test we choose scenarios 100sensorfield3x70rout/reroute.tcl where the fidelity requirement is met and we compare energy consumption in case of rerouting and without rerouting paths. Figures 5-75 and Figure 5-76 show that rerouting is more demanding on energy and that the energy consumed per delivered packet is also larger for the rerouting case. In simple words, rerouting scenario receives 3% more packets than no rerouting and consumes almost 30% more energy. Of course in high congested scenarios this situation is completely different. To conserve energy, after the problem pass, must return to the original path.

5.1.4.3.2 TEST 2

The next scenarios as previously mentioned examine the importance of topology aware rerouting (resource control)to alleviate congestion in wireless sensor networks. For these tests a different topology have been created as depicted in Figure 5-77. Rerouting is topology aware when the selection for the merging point is done as closed to the sink as possible. In Figure 5-78 this is noted with the blue line , whilst the yellow one show a case of topology unaware rerouting. Again results have been taken for different traffic rates(congestion levels).The 100sensorfieldrerout3X.tcl scenarios gives different results when using different routing tables to create different rerouting paths.

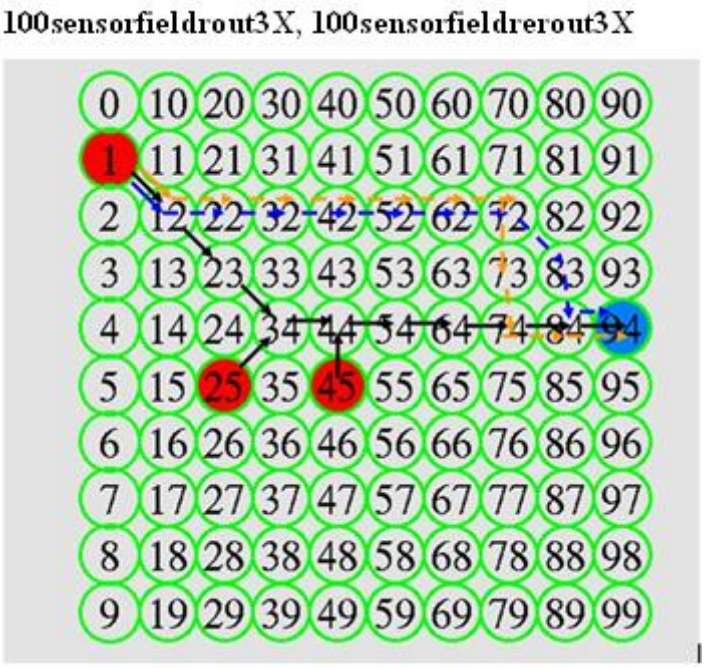


Figure 5-77 Topology Aware Rerouting

.

The reliability of a topology aware rerouting algorithm is the best as depicted in Figure 5-78. Next comes topology unaware rerouting and last is the case without any congestion control. Total energy consumption in Figure 5-79 is almost the same for topology aware and unaware routing which consumes on average about 3 joules more than that of no rerouting scenarios as more nodes are included in the paths and more traffic can be sent.

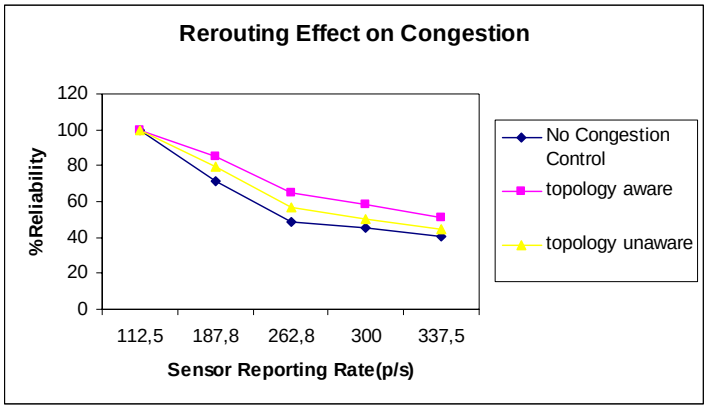


Figure 5-78 Topology Aware Rerouting Effect on Congestion

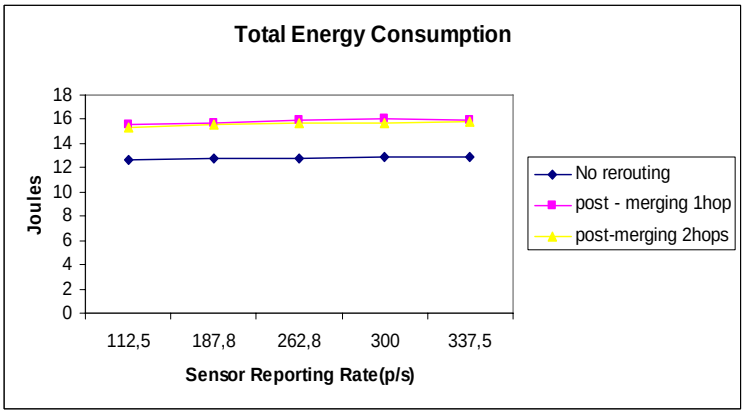


Figure 5-79 Topology Aware Rerouting & Total Energy Consumption

5.1.4.3.3 TEST 3

Coming to test 3 of these simulations, we investigate the effect of multi-path routing to congestion alleviation. When wireless sensor network overcomes a Cmin of network capacity, then rerouting isn' t efficient. In such situations traffic splitting on multiple paths to the sink must be the solution to congestion. Doing so, we have created the scenarios depicted in Figure 5-80.

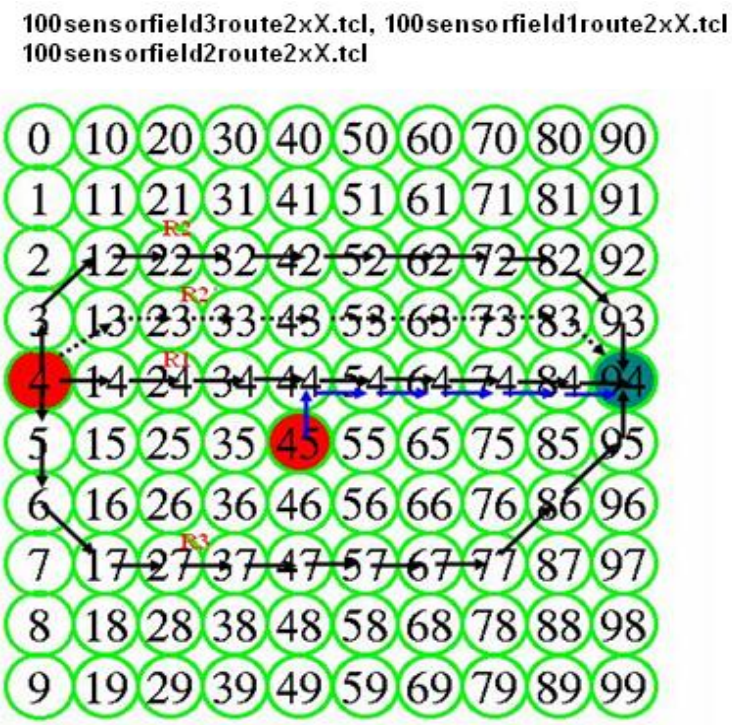


Figure 5-80 Multi-path Routing

Note: The 2-routes scenarios use R2 as second path.R2' is used in simulations of TEST 4.

Creating 1,2 and 3 routes from source 4 to the sink and distributing evenly the source 4 traffic to 1,2 and 3 routes respectively we can observe the congestion –related packet drops for each scenario. As can be seen from Figure 5-81 that scenarios with a single route give the most packet drops, then 2-routes scenario follows and last the 3-routes scenario has the best performance, as it can deliver more traffic load without drops, compared to the other scenarios.

It is very important to note that 2-routes scenarios succeeded about the same average energy consumed per delivered packet, with 1-route scenarios in the high congested region of the network , beyond 175,2 p/s(Figure 5-82).This means that it is sometimes possible , in high congested scenarios multi-path routing to offer packet energy efficiency comparable to that of the single route, because it can deliver much more packets to the sink during the event period.

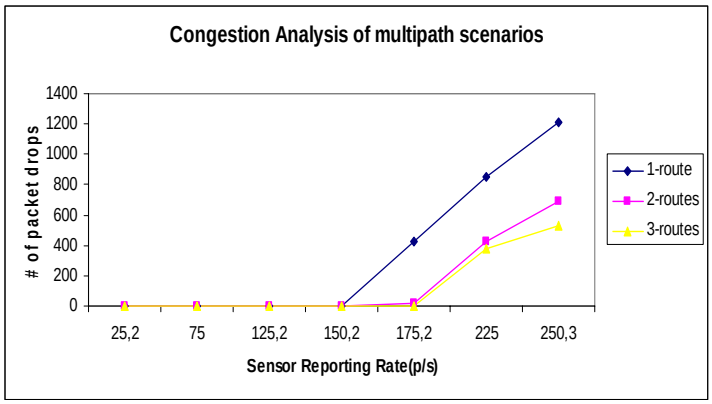


Figure 5-81 Multi-path Routing Effect on Congestion

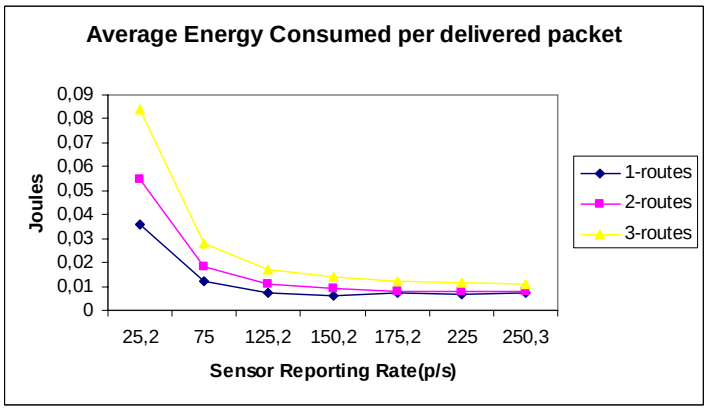


Figure 5-82 Multi-path Routing & Energy Consumed per Delivered Packet

5.1.4.3.4 TEST 4

In conjunction to test 3 , in order to investigate the effect of new routes selection, we created R2' (Figure 5-20) to be the second route toward the sink. The results were compared with that of route R2 simulated in test 3.Simulations have been created for most congested scenarios like 70,90,100Kbps source reporting rate.

As shown in Figure 5-83, comparing these 2 cases, packet drops are much more when the second route to the sink is that of R2'. It doesn't worth to create a such new route as the network performance is comparable with that of single route. The reason why this happens is that the new route is in interference range of the main route. This way the proximity to the congested flow causes problems than help on congestion alleviation.

5.1.4.3.5 TEST 5

In this test we investigate the effect on Congestion of the traffic distribution on multiple routes. Simulation have been run for high traffic source rates(70 and 90 kbps).Traffic distribution was organized so most of the traffic would follow the detour path (R2 on topology).

Results plotted in Figure 5-84 shows that traffic distribution on rerouting paths has significant effect on the performance of multi-path routing for congestion alleviation. The more percentage is given to detour path the better Congestion Alleviation results we have. This is because as detour path is less congested from origin path most of the traffic can be forwarded to the detour path without congestion packet drops.

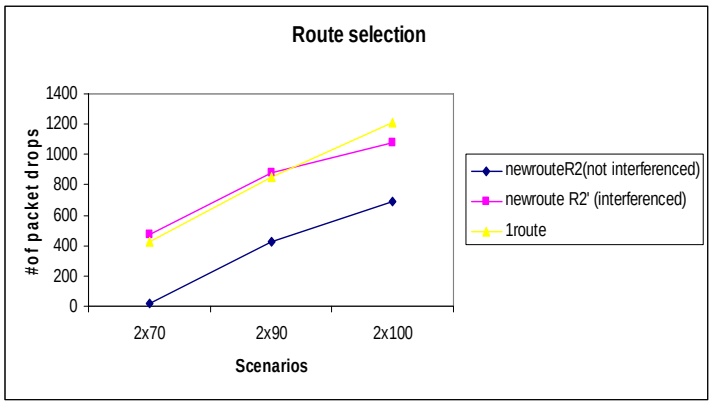


Figure 5-83 Route Selection Effect on Congestion

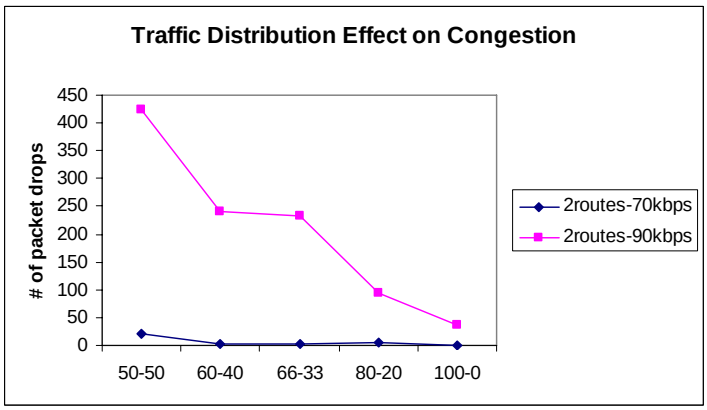


Figure 5-84 Route Traffic Distribution Effect on Congestion

5.1.5 Summary of results

Working with section's A simulation Studies, we found out that Congestion have negative effects on energy consumption and fairness. As simulation results show, percentage of energy consumed for packets drops, increases when congestion increases, and total energy consumed and energy consumed per delivered packet increases too. This is a strong reason why to use congestion control is sensor networks. Furthermore far sensor nodes suffer from starvation achieving low packet delivery ratio. This creates big problems to applications that sensing different but of equal worth phenomena, when congestion occurs. Also Congestion Types show different packet losses and so need different congestion solution approaches. It is also interesting to refer to the various test

of the effect of the different wireless Sensor Network parameters on congestion. It is shown that when well designed or controlled can help avoid or control congestion. According to the simulations :

- Number of sources: Controlling the number of sources sensing the same event we can control congestion problem. This could help to alleviate or control source congestion.
- RTS/CTS mechanism: In high congested networks RTS/CTS mechanism adds on congestion while indeed end –to end delay is increased. So it is better to be disabled.
- Maximum Number of MAC retransmissions: When MAC errors dominates other errors, large maximum number of retransmissions helps to accommodate more traffic, but this seems not have any effect in high congested networks, where buffer overflows predominates.
- Sensor Node Buffer Size: As the network load increases buffer sizes even 200 packets, cannot accommodate the traffic and show the same results as smaller buffer sizes. On the other hand when large buffer sizes are used, there is large increase in end-to-end delay. Thus for applications where reliability can be afforded to be around 95% and end-to-end delay is very important, lower buffer can be selected.
- Minimum Contention Window: Larger minimum contention window has a negative impact at low to medium reporting rates, whilst at higher rates there is not significant influence at network performance.
- Average Path Length from the sink: Lower average path length has positive effects on congestion. This give us also a good directive to add more sinks, so decrease the average path length.
- Ad-hoc routing protocols: DSDV shows not to behave well in congested networks as always shortest path to the sink is find and there is large traffic concentration. AODV gives better results so it is recommended to be used instead of DSDV. Furthermore AODV shows even better results when in low traffic an initial traffic is generated before the network crisis state.

Following some of the contributions of section's A results, but also inspired from related work on congestion control we evaluate different congestion control solutions for three different congestion types. Particularly the following results appear:

- Using random delay transmission at application layer we achieved to have less sources sending data simultaneously. This way Source congestion alleviated .Larger duration of this delay gives better results.
- For sink congestion again inspired from previous results we added more sinks to achieve load balancing and keep path length at normal values. Results were positive when the placement of extra sinks was correctly decided in order to avoid interference from congested flows.
- For forwarder congestion rerouting and multi-path routing was used as tested solution. Results have under some conditions positive effects on congestion. Particularly topology aware rerouting or multi-path routing is surely more efficient and can help alleviating congestion. On the other hand as shown from the results these kinds of techniques are highly energy consuming so a shrinking phase must follows when congestion pass. Multi-path routing have to be used only when rerouting is not efficient and more paths are needed to accommodate traffic. This is the most energy consumable solution and again a shrinking phase must exist. The performance of this solution is affected from the traffic distribution on each path and the selection of new paths. As simulation results shows, more traffic must be distributed to the detour paths as are less congested, and the new route selection must take in account topology to avoid interference.

Chapter 6

Sensor networks Congestion Control Framework

6.1 Description and Objectives of proposed Framework

According to chapter 5 results, in section A cases of study, just configuring some network parameters we can control congestion in some extend. But of course a global – completed solution (framework) is needed to strongly alleviate congestion in wireless sensor networks.

In this section we will propose a congestion control framework using formal techniques like rate control and resource control in combination. Although congestion control in wireless sensor networks has been investigated, the combination of rate control and resource control in a congestion controller still be an open issue. One may attribute this to the focus of wireless sensor network engineers on alleviate congestion using mainly rate control methods without thinking on how sink's application requirements are to be met, and also because of large energy consumption of resource control methods. As mentioned in Chapter 4, recently some attempts have been made to develop congestion controllers using resource control methods but no one used a hybrid technique. So is still a challenging unresolved problem.

A such hybrid framework can eliminate disadvantages of each of the techniques when used as standalone and become a strong and effective framework to alleviate congestion.

Our objective is to propose such a hybrid framework at theoretical level that can effectively give solutions to congestion problems in sensor networks, always meet application's fidelity requirements and also conserve energy. Also this framework must give solutions for three different sensor networks types(Figure 6.1), source congestion, sink congestion and forwarder congestion.

This framework is based on previous results of Chapter 5, and related work of Chapter 4:

- Firstly and only throttle the traffic when applications fidelity requirements are met and / or energy consumed is more than the available budget.
- Use of resource control, and take advantage of "redundant" resources in sensor networks using a family of strategies :

1. At Run –Time create more routing paths (e.g. by waking up nodes) in order to share the load when congestion is detected. This showed to be beneficial according to results of Chapter 5, for Forwarder congestion type.
2. Using good planning (offline) insert uniformly more sinks in the sensor field and share the traffic to them. Correct placement of sinks showed to be important. This showed to be effective for Sink Congestion Alleviation.
3. Using good planning (offline), control the number of sources sensing an event, thus the generated data. Random delay transmission is used, as showed to have very effective results on Source Congestion Alleviation.

- Turn off the extra resources as soon as the source traffic decreases to save energy.

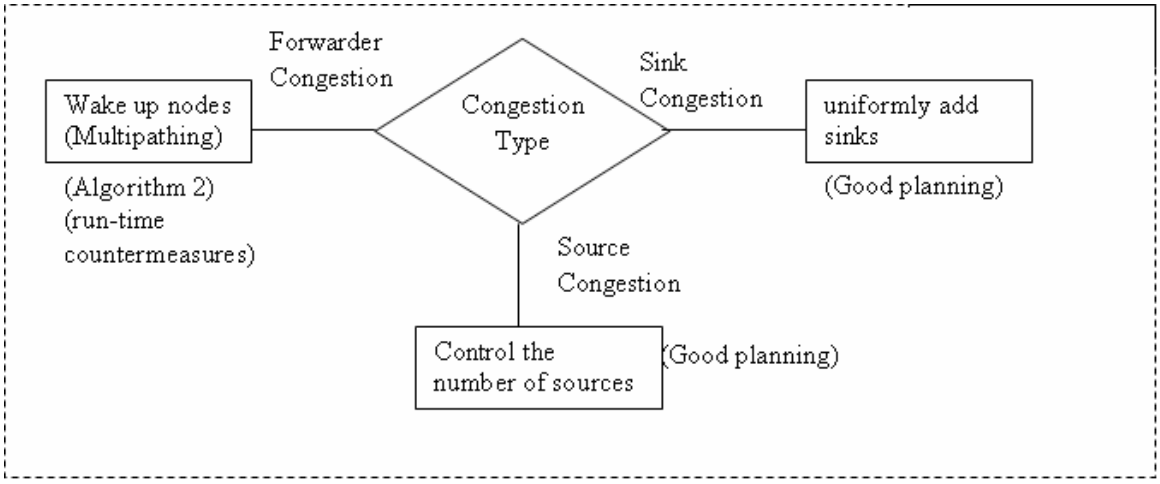


Figure 6-1 Framework Congestion Solutions

Figure 6-2 depicts the main components of our framework. Our Framework operates with offline and run-time modules. According to results of evaluating congestion Types solutions in Chapter 5(Section B cases of study), source congestion and sink congestion could be controlled – eliminated with a good planning from sensor network engineers. This work must be done inside offline module were the source-sink congestion solutions tested in Chapter 5 are included.

In run-time module run time measures are taken to control Forwarder type congestion problem. Here the hybrid technique is proposed for congestion alleviation.

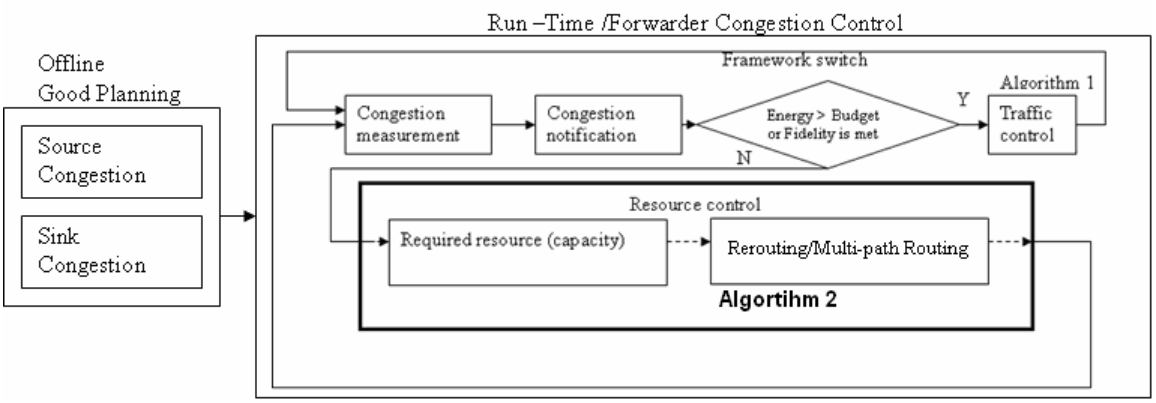


Figure 6-2 Framework Description

6.2 Framework Components Design

In the following Framework explanation, any reference for Ti(t) means the aggregate incoming traffic volume at a node and for Ri(t) the effective resource capacity.

- Offline Module

Includes congestion controllers for source congestion and sink congestion types. It is based on good planning from sensor network engineers. Specific solutions evaluated in Chapter 5 is adopted. Particularly it uses:

- Random Transmission Delay to reduce the number of sources transmitting simultaneously to alleviate Source Congestion
- Add uniformly Sinks for load balancing to alleviate Sink Congestion.

- Run-Time Module

At run time firstly we have to measure-detect congestion. This is done in congestion measurement function. Then congestion notification(feedback) must trigger the use of congestion control scheme. Before congestion controller operates our framework switch choose between resource control scheme and rate control scheme. Algorithm 1(the well known AIMD) is used in case rate control is chosen and algorithm 2 in case of resource control. Algorithm 2 constructs rerouting paths and or use multi-path methods. With few words finds traffic distributor sensor node, traffic merger, detour path and control the traffic distribution on the detour paths and the original one. Finally when congestion problem is alleviated deconstructs detour paths and routes the traffic to the original route, conserving energy.

6.2.1 Congestion Measurement(Congestion Detection)

Congestion detection in sensor networks has been studied in **[14 , 47]**. As pointed out in both papers, the actual congestion level around a node cannot be accurately reflected by the buffer occupancy alone, which is a popular metric in traditional wired networks. Therefore, our congestion control scheme must measure not only the buffer occupancy but also the channel loading **[14]**. A node's congestion level is an aggregation of these two metrics. As soon as the congestion level hits the upper watermark, it declares congestion, and becomes a *hot spot node*.

6.2.2 Congestion Notification(Feedback)

When congestion at a node is detected , the congested node I sends a tri-nary feedback as shown in Equation 6-1.

$$b_i(t) = \begin{cases} 0 & \text{if} \quad \text{cong.level} < \text{cong.thresh} - \text{wl}(\text{no congest.}) \\ -1 & \text{if} \quad \text{cong.thresh} - \text{wl} \le \text{cong.level} \le \text{cong.thresh} - \text{wu} \\ 1 & \text{if} \quad \text{cong.level} > \text{cong.thresh} - w_u\,(\textit{conges.}) \end{cases}$$

$$\text{wl} > \text{wu} > 0$$

Equation 6-1 Source Control –Tri-nary Feedback [41]

This feedback is end-to-end for traffic control function and 1-step backward to the first uncongested node, when resource control is used.

Binary feedback can only tell if there is congestion or not. In contrast, non-binary (eg tri-nary)feedback carries more information, which can indicate the congestion level(using watermarks), which is a more accurate method. Where *wu* and *wl* are constants and $wu \geq 0$, $wl \geq 0$, and $wl \geq wu$. These values are surely dependent on the topology and density of the network, so must be accordingly adjusted.

When node *i*'s congestion level falls below the lower watermark, the congestion feedback is set to 0 indicating that there is no congestion. When it exceeds the upper watermark, the congestion feedback function returns 1 indicating congestion.

6.2.3 Framework switch

The Framework switch is where the decision between resource control or traffic rate adaptation operation is taken following the criteria below: The criteria for selecting a resource control policy are as follows:

- *Fidelity:* One of the motivations for resource controlling is to transfer as much incoming traffic as possible, at least above the required fidelity level *F* (in bits per unit of time), to one or more sinks during a crisis state, so that the delivered data can produce a meaningful view of the sensed event and subsequently incur necessary actions by the application that extracts the delivered data from the sinks.

- *Energy Efficiency:* Increasing the effective resource capacity can improve the quality of service observed by the application, but it may also increase the total energy consumption due to the higher data transmission rate and the maintenance overhead of the increased resource provisioning. In addition, if the increased resource capacity still does not accommodate the incoming traffic, some of the traffic should be discarded due to congestion, thereby nullifying the energy expended for receiving the traffic from the previous hop nodes.

- *Packet Energy Efficiency:* Packet energy efficiency indicates the average energy consumed by the network to successfully forward a packet from a source to a sink. The goal of the resource control strategy is how to systematically adjust the effective resource capacity available to a node in conjunction with the incoming traffic volume while satisfying above criteria. Seeking for optimum solution using resource

control an algorithm must achieve the highest fidelity level with the lowest energy consumption, thereby maximizing the packet energy efficiency, which combines those two key criteria together.

- *Bit Energy Efficiency* The problem of maximizing the packet energy efficiency, which is measured from the whole network, can be reduced to the problem of minimizing the *bit energy* consumed by each individual node in order to successfully forward 1 bit to the next hop if the packet size is constant. The bit energy consumed by a node is calculated by dividing the total energy consumption by the total outgoing traffic that are successfully received by the next hop node. Therefore, the goal of the resource control strategy under fidelity and energy constraints during the event period of *D* is represented as follows:

$$\text{Minimize } \frac{\int_0^D E_I(t)dt}{\int_0^D \min(T_I(t), R_I(t))dt}$$

Equation 6-2 Minimized Bit Energy Consumed

where $Ei(t)$ is the total energy consumed by node *i* at time *t* and $min(Ti(t),Ri(t))$ indicates the outgoing traffic from node *i* at time *t*.

This analysis is used in **[41]** and results show that the bit energy consumed by a node i is minimized when the node's effective channel capacity is equivalent to the aggregate incoming traffic, ie Ri(t)=Ti(t).

- *The traffic controlling can be triggered:*
  - when the cumulative energy consumption after the resource control is triggered, exceeds the energy budget per event.
  - when the cumulative outgoing traffic exceeds the required fidelity amount.

6.2.4 Control Scheme Type (Hybrid)

The approach of controlling traffic for congestion avoidance has been extensively studied, mostly through the active queue management (AQM) in wired networks **[[48]]**.Two main criteria for the known traffic control policies are resource utilization and fairness. In Figure 6-3 , the area above the efficiency line corresponds to overload scenarios while the area below to underloaded scenarios. The resource line and fairness line intersect to Ri/2,Ri/2 which corresponds to optimal scenario **[[48]]**

However the rationale behind resource control strategies is distinctively different from that of traffic control. Traffic control strategies assume a fixed resource provisioning , Ri at node i. This constraint does not halt in sensor networks. The available capacity at a node is elastic due to its dynamic duty cycle, interference and etc, thereby represented as time varying function Ri(t).As a result resource control schemes seek to satisfy the fidelity level requirement of each flow even during congestion by assigning additional resources to the flow that has higher fidelity level requirement without taking resources away from other flows.

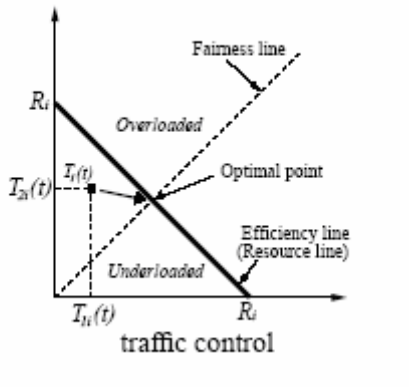


Figure 6-3 Traffic Control [[1]]

When using traffic control our scheme runs Algorithm 1 that simply implements the well known AIMD algorithm , shown in Equation 6-3 as adopted from **[[1]]**.

$$T_i(t) = \begin{cases} T_I(t) + a_I & if \quad b_i(t) = 0 \\ m_{DI} T_I(t) & if \quad b_i(t) = 1 \end{cases}$$

Equation 6-3 AIMD algorithm

When resource control is used we define the resource control policy as in Equation 6-4.Algorithm 2 runs according to this policy. This Algorithm creates the needed network capacity using the solutions we evaluated in Chapter 5 with rerouting , topology aware rerouting and multi-path routing. It also deconstruct rerouting and multi-path routing when congestion problem pass. **Algorithm 2 MUST** use lessons of Chapter 5, for resource control, as to distribute more traffic to the detour paths, use topology aware rerouting(merging as near the sink as possible)and choose new paths to not interfere with the original path.

$$R_i(t+1) = \begin{cases} removeresources & if \quad b_i(t) = 0 \\ do\,nothing & if \quad b_i(t) = -1 \\ addresources & if \quad b_i(t) = 1 \end{cases}$$

Equation 6-4 Resource Control policy

In resource control only Ri(t), effective channel capacity is adjustable. The fist backward uncongested node on the routing path of the congested flow, that receive the feedback reacts accordingly. In case value is -1, no congestion has occurred but incoming traffic is almost equal to effective channel capacity so effective channel capacity is kept unchanged. When feedback is 1 congestion has occurred and effective channel capacity which at the moment is less than the aggregate incoming traffic must be increased to reach aggregate incoming traffic, using multi-path and rerouting techniques. In case feedback is 0 deconstruction of the detour paths must occurs as congestion passed and energy conservation is needed.

Chapter 7

CONCLUSION AND FUTURE WORK

7.1 Conclusion

The Thesis addresses the Congestion Problem and Congestion Control in WSNs. It first identifies the several network aspects, some of which are unique to sensor networks, that affect congestion, study congestion symptoms for different Wireless Sensor Network Congestion Types and moreover, since congestion affects on energy consumption and fairness, these are also examined.

This Thesis then confirms by simulations that specific congestion solutions can be very effective when used for congestion control.

Finally we propose a theoretical Congestion Control Framework that takes account of previous investigations / factors that influence congestion. This framework follows at run-time a hybrid technique that uses rate control or resource control aiming to alleviate congestion, and hence meet application's fidelity requirements and conserve energy.

This Thesis makes several own contributions that are listed below:

- we presented a new theoretical Congestion Control Framework
- we showed by simulations the effect on congestion using resource control via topology changes, coming up to important directives.
- implemented exhaustive tests on how various network parameters affect Congestion.
- implemented StaticRT, routing protocol in NS-2, to show the impact of rerouting and multi-path routing on Congestion.

- achieved appropriate modifications in NS-2, to imitate sensor nodes-sensor networks.
- developed awk programs, to calculate energy, packet drops, and other useful statistics for this Thesis.

The main results are summarized below :

1. resource control using topology changes is confirmed to be capable of controlling congestion in all simulations, under some conditions.

2. as simulation results show, source congestion can be alleviated via transmission delay at the application layer. Larger duration of this delay gives better results.

3. sink congestion can also be alleviated using more sinks for load balancing and keeping path length at normal values. The placement of extra sinks must be correctly decided in order to avoid interference from congested flows.

4. for forwarder congestion, rerouting will not always have positive effects on congestion. Topology aware rerouting or multi-path routing is surely more efficient and can help alleviating congestion. As shown from the results it is highly energy consuming so a shrinking phase must follow when congestion pass.

5. multi-path routing can alleviate congestion when rerouting is not efficient and more paths are needed to accommodate traffic. This is the most energy consumable solution and again a shrinking phase must exist. The performance of this solution is affected from the traffic distribution on each path and the selection of new paths. As simulation results show, more traffic must be distributed to the detour paths as are less congested, and the new route selection must take in account topology to avoid interference.

6. Congestion have negative effects on energy consumption and fairness. As simulation results show, percentage of energy consumed for packets to that drop increases when congestion increases, and total energy consumed and energy consumed

per delivered packet increases too. Furthermore far sensor nodes suffer from starvation achieving low packet delivery ratio.

7. Congestion Types show different packet losses and so need different congestion solution approaches.

8. Wireless Sensor Network parameters, when well designed or controlled can help avoid or control congestion.

9. According to the simulations :

- Number of sources: Controlling the number of sources sensing the same event we can control congestion problem.
- RTS/CTS mechanism: In high congested networks RTS/CTS mechanism adds on congestion while indeed end –to end delay is increased. So it is better to be disabled.
- Maximum Number of MAC retransmissions: When MAC errors dominates other errors, large maximum number of retransmissions helps to accommodate more traffic, but this seems not have any effect in high congested networks, where buffer overflows predominates.
- Sensor Node Buffer Size: As the network load increases buffer sizes (even 200packets) , cannot accommodate the traffic and show the same results as smaller buffer sizes. On the other hand when large buffer sizes are used, there is large increase in end-to-end delay. Thus for applications where reliability can be afforded to be around 95% and end-to-end delay is very important, lower buffer can be selected.
- Minimum Contention Window: Larger minimum contention window has a negative impact at low to medium reporting rates, whilst at higher rates there is not significant influence at network performance.
- Average Path Length from the sink: Lower average path length has positive effects on congestion.
- Ad-hoc routing protocols: DSDV shows not to behave well in congested networks as always shortest path to the sink is found and there is large traffic concentration. AODV gives better results so it is recommended to be used instead of DSDV. Furthermore AODV shows even better results when in low traffic an initial traffic is generated before the network crisis state.

7.2 Future Work

Early results given by the simulations and tests of the Thesis, are very promising and encourage us to investigate toward the hybrid framework even further.

This can include the following:

- Full development of the hybrid framework in NS-2.
- Formal analysis and evaluation of the on-line hybrid framework
- extension to large scale (larger sensor scenarios)

This Thesis has shown several promising results on congestion control either just configuring specific network parameters or using specific congestion solutions related with resource control. Based on that results a hybrid congestion control Framework is proposed at theoretical level, and can be considered a good basis to build upon for further research in the direction of congestion control is WSNs.

References

[1] Dah-Ming Chiu and Raj Jain, "Analysis of the increase and decrease algorithms for congestion avoidance in computer networks. Computer Networks and ISDN Systems", 17, 1989.

[2] Ion Stoica, Scott Shenker, Hui Zhang, Core-Stateless Fair Queueing: "A Scalable Architecture to Approximate Fair Bandwidth Allocations in High Speed Networks", SIGCOMM 1998.

[3] Bianchi,G. Dipt. di Ingegneria Elettrica, Palermo Univ, **"** Performance analysis of the IEEE 802.11 distributed coordination function", IEEE Journal ,2000.

[4] W. Ye, J. Heidemann, and D. Estrin, "An Energy-Efficient MAC Protocol for Wireless Sensor Networks" in Proceedings of IEEE INFOCOM'02, June 2002.

[5] Jinyang Li, Charles Blake, Douglas S. J. De Couto, Hu Imm Lee and Robert Morris, "Capacity of Ad Hoc Wireless Networks", Mobicom'01 2001.

[6] Charles E. Perkins and Pravin Bhagwat, "Highly dynamic Destination-Sequenced Distance-Vector routing (DSDV) for mobile computers". In Proceedings of the SIGCOM '94 Conference on Communications Architecture, protocols and Applications, pages234-244, August 1994. A revised version of the paper is available from http://www.cs.umd.edu/projects/mcml/papers/Sigcomm94.ps. (1998-11-29).

[7] Charles E. Perkins, Elizabeth M. Belding-Royer, and Samir Das. "Ad Hoc On Demand Distance Vector (AODV) Routing" IETF RFC 3561.

[8] C. T. Ee and R. Bajcsy, "Congestion Control and Fairness for Many-to-One Routing in Sensor Networks," in proc. of ACM SenSys 2004, November 2004.

[9] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A Survey on Sensor Networks," IEEE Communication Magazine, August 2002.

[10] Antoniou A, Sergiou Ch, "Congestion Control in WSNs" Final Project EPL 606 Dept. Computer Science, Univ of Cyprus, 2006.

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

[12] M.A.M. Vieira, C.N. Coelho. Jr., D.C. da Silva Jr., and J.M. da Mata, "Survey on Wireless Sensor Network Devices," IEEE, 2003.

[13] J. Feng, F. Koushanfar, and M. Potkonjak, "System-Architecture for Sensor Networks Issues, Alternatives, and Directions," ICCD'02, 2002.

[14] B. Hull, K. Jamieson, and H. Balakrishnan, "Mitigating Congestion in Wireless Sensor Networks," in proc. of ACM SenSys 2004, November 2004.

[15] S. Tilak, N. B. Abu-Ghazaleh, and W. Heinzelman. Infrastructure tradeoffs for sensor networks. ACM Workshop on Wireless Sensor Networks and Applications (WSNA), Sep. 2002.

[16] E. J. Duarte-Melo and M. Liu. "Data-Gathering Wireless Sensor Networks: Organization and Capacity" , Computer Networks, 43, 2003.

[17] C.-Y. Wan, S. B. Eisenman, and A. T. Campbell, "CODA: Congestion Detection and Avoidance in Sensor Networks," in proc. of ACM SenSys'03, November 2003.

[18] S. Ratnasamy, D. Estrin, R. Govindan, B. Karp, S. Shenker, L. Yin, and F. Yu. Data-centric storage in Sensornets. In the first ACM International Workshop on Wireless Sensor Networks and Applications (WSNA), 2002.

[19] Jaewon Kang, Yanyong Zhang, and Badri Nath, "End-to-End Channel Capacity Measurement for Congestion Control in Sensor Networks," In Proceedings of the Second International Workshop on Sensor and Actor Network Protocols and Applications (SANPA, formerly IEEE SNPA) in conjunction with MobiQuitous held in cooperation with IEEE Computer Society and ACM SIGMOBILE, 2004.

[20] Mehmet C. Vuran, Vehbi C. Gungor, Ozgur B. Akan, "On the Interdependence of Congestion and Contention in Wireless Sensor Networks", Third International Workshop on Measurement, Modeling, and Performance Analysis of Wireless Sensor Networks, San Diego, CA, 2005.

[21] Johnson, Maltz, Hu, "The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR)", Interent Draft,16 April 03.

[22] V.Park , S. Corson, "Temporally-Ordered Routing Algorithm (TORA) Version 1,IEFT RFC 2026", July 2001.

[23] S.-J. Lee and M. Gerla, "Dynamic load-aware routing in ad hoc networks," in Proc. IEEE International Conference on Communications (ICC '01), vol. 10, pp. 3206–3210, Helsinki, Finland, June 2001.

[24] Y. Sankarasubramaniam, O. Akan, and I. Akyildiz, "ESRT: Event-to-Sink Reliable Transport in Wireless Sensor Networks," in proc. Of 4th ACM Symposium on Mobile Ad Hoc Networking & Computing (MobiHoc'03), June 2003.

[25] S. Chen, N. Yang "Congestion Avoidance based on Light-Weight Buffer Management in Sensor Networks" ,University of Florida, Gainesville, FL 32611, USA.

[26] Hull, B., Jamieson, K., and Balakrishnan, H, "Bandwidth management in wireless sensor networks", Tech. Rep. 909, MIT Laboratory for Computer Science, July 2003.

[27] C. T. Ee and R. Bajcsy, "Congestion Control and Fairness for Many-to-One Routing in Sensor Networks," in proc. of ACM SenSys 2004, November 2004.

[28] Kiran Yedavalli, "Using Wireless Advantage for Congestion Control in Wireless Sensor Networks, "USC Technical Report CENG-2005-13, December 05.

[29] JaeWon Kang, Yanyong Zhang, Badri Nath, "Adaptive Resource Control Scheme to Alleviate Congestion in Sensor Networks", In the First Workshop on Broadband Advanced Sensor Networks.

[30] Glauche, Ingmar,Krause, Wolfram,Sollacher, Rudolf,Martin," Distributive routing and congestion control in wireless multihop ad hoc communication networks", Elsevier ,2004.

[31] Rangwala, R. Gummadi, R. Govindan, and K. Psounis, "Interference-Aware Fair Rate Control in Wireless Sensor Networks", SIGCOMM 2006.

[32] Chieh-Yih Wan, Shane B. Eisenman, Andrew T. Campbell, Jon Crowcroft: "Siphon: overload traffic management using multi-radio virtual sinks in sensor networks", SenSys 2005: 116-129.

[33] Victor Shnayder ''Application Aware Congestion Control in Wireless Sensor'', Harvard University.

[34] Ray, S. Carruthers, J.B. Starobinski, D. , "RTS/CTS-induced congestion in ad hoc wireless LANs", Volume: 3, On page(s): 1516- 1521 vol.3, 16-20 March 2003.

[35] Cui –Qing Yang and Alapati V.S Reddy, " A Taxonomy for congestion Control Algorithms, in Packet Switching Networks", IEEE August 1995.

[36] Laura Galluccio , Andrew Campel, Sergio Palazzo, "Concert: aggregation-based congestion control for sEnsoR networks", Sensys05 p 274 - 275 , 2005

[37] Alec Woo and David Culler, "A transmission control scheme for media access in sensor networks," in Proceedings of the ACM/IEEE International Conference on Mobile Computing and Networking, Rome, Italy, July 2001, pp. 221–235, ACM.

[38] Falko Dressler, "Locality Driven Congestion Control in Self-Organizing Wireless Sensor Networks," Proceedings of 3rd International Conference on Pervasive Computing (Pervasive 2005): International Workshop on Software Architectures for Self-Organization, and Software Techniques for Embedded and Pervasive Systems (SASO+STEPS 2005), Munich, Germany, May 2005.

[39] C. Lu, B.M. Blum, T.F.Abdelzaher, J.A. Stankovic, and T. He, "RAP: A Real-Time Communication Architecture for Large-Scale Wireless Sensor Networks," RTAS, September 2002.

[40] Kyriakos Karenos, Vana Kalogeraki and Srikanth V. Krishnamurthy, "Cluster-based Congestion Control for Supporting Multiple Classes of Traffic in Sensor Networks", IEEE 2005.

[41] Jaewon Kang, Yanyong Zhang Badri Nath,"Analysis of Resource Increase and Decrease Algorithm in Wireless Sensor Networks", 11th IEEE Symposium on Computers and Communications (ISCC'06) pp. 585-590.

[42] Raju Kumar_, Hosam Rowaihy_, Guohong Cao_, Farooq Anjum†, Aylin Yener‡ and Thomas La Porta, "Congestion Aware Routing in Sensor Networks " TR 2006.

[43] Imad Aad, Claude Castelluccia: "Differentiation Mechanisms for IEEE 802.11", INFOCOM 2001: 209-218.

[44] 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.

[45] http://www.isi.edu/nsnam/ns/

[46] P. Johansson, T. Larsson, N. Hedman, B. Mielczarek, and M. Degermark, "Scenario-based performance analysis of routing protocols for mobile ad-hoc networks," in the 5th Annual International Conference on Mobile Computing and Networking (MOBICOM), (Seattle, WA), pp. 195–206, Aug. 1999.

[47] J. Kang, Y. Zhang, and B. Nath, "Accurate and energy-efficient congestion level measurement in ad hoc networks", In Proceedings of IEEE WCNC, Mar. 2005.

[48] Sally Floyd, Kevin Fall, "Promoting the use of end-to-end congestion control in the Internet", IEEE/ACM Transactions on Networking,1999.

Appendix A

A.1 Sample Scenario Code written in TCL script language.

```
###scenario:100sensorfield5x100.tcl
####Author:Antonis Antoniou
# Define options
#=======
set val(chan)         Channel/WirelessChannel    ;# channel type
set val(prop)         Propagation/TwoRayGround   ;# radio-propagation model
set val(netif)        Phy/WirelessPhy            ;# network interface type
set val(mac)          Mac/802_11                 ;# MAC type
set val(ifq)          Queue/DropTail/PriQueue    ;# interface queue type
set val(ll)           LL                         ;# link layer type
set val(ant)          Antenna/OmniAntenna        ;# antenna model
set val(ifqlen)       50                         ;# max packet in ifq
set val(nn)           100                        ;# number of mobilenodes
set val(rp)           StaticRT                   ;# routing protocol
set val(x)            500                        ;# X dimension of topography
set val(y)            500                        ;# Y dimension of topography
set val(finish)       40                         ;# time of simulation end
#
================================================================
=======

# Main Program
```

```
#
# ======================================================================
# =======

# Initialize Global Variables

set ns [new Simulator]
Antenna/OmniAntenna set X_ 0
Antenna/OmniAntenna set Y_ 0
Antenna/OmniAntenna set Z_ 1.5
Antenna/OmniAntenna set Gt_ 1.0
Antenna/OmniAntenna set Gr_ 1.0
Phy/WirelessPhy set CPThresh_ 10.0
Phy/WirelessPhy set CSThresh_   2.13643e-07
# INTERFERENCE DISTANCE KAI COMMUNICATION DISTANCE ARE SAME
### distance 30m
Phy/WirelessPhy set RXThresh_   2.13643e-07
### distance 30m
Phy/WirelessPhy set bandwidth_ 2e6
Phy/WirelessPhy set Pt_ 0.28183815
Phy/WirelessPhy set freq_ 914e+6
Phy/WirelessPhy set L_ 1.0
Mac/802_11 set dataRate_ 2Mb
# Enable RTS
Mac/802_11 set RTSThreshold_ 0  ## RTS/CTS is Enabled

#############Mac/802_11 set CWMin_ 256       ;# These parameters are setup for cw
############Mac/802_11 set CWMax_ 1023

set tracefd [open 100sensorfield5x100.tr w] ###tracedile for this scenario
set log [open output2.tr w]
set f1 [open received.tr w]
set f2 [open lost.tr w]
set namtrace [open out2.nam w]

$ns trace-all $tracefd
$ns use-newtrace
$ns namtrace-all-wireless $namtrace $val(x) $val(y)

# Define a 'finish' procedure

proc finish {} {
   global ns tracefd log namtrace f1 f2
   $ns flush-trace
   close $tracefd
   close $namtrace
```

```
    close $f1
    close $f2
    close $log

    exec nam out2.nam &  ####animate results on mam

}


# Set up topography object

set topo       [new Topography]

$topo load_flatgrid $val(x) $val(y)

# Create God

create-god $val(nn)

#Create channel
set channel1_ [new $val(chan)]

#  Create the specified number of mobilenodes [$val(nn)] and "attach" them to
#the channel.
#  Here two nodes are created : node_(0) and node_(2)

# Configure nodes according to scenario....

        $ns node-config -adhocRouting $val(rp) \
                        -llType $val(ll) \
                        -macType $val(mac) \
                        -ifqType $val(ifq) \
                        -ifqLen $val(ifqlen) \
                        -antType $val(ant) \
                        -propType $val(prop) \
                        -phyType $val(netif) \
                        -channel $channel1_ \
                        -topoInstance $topo \
                        -agentTrace ON \
                        -routerTrace ON \
                        -macTrace ON \
                        -movementTrace OFF \
                        -energyModel EnergyModel \
                       -idlePower 0.2 \
                       -rxPower 1.0 \
                       -txPower 2.0 \
```

```
                -sleepPower 0.00 \
                -initialEnergy 1000

# Provide initial (X,Y, for now Z=0) co-ordinates for mobilenodes

#================================
#          - Node -
#        creating nodes
#================================


set node_(0) [$ns node]
$node_(0) set X_ 233
$node_(0) set Y_ 247
$node_(0) set Z_ 0.0
$ns initial_node_pos $node_(0) 20


set node_(1) [$ns node]
$node_(1) set X_ 233
$node_(1) set Y_ 227
$node_(1) set Z_ 0.0
$ns initial_node_pos $node_(1) 20
……
……..
…..
set node_(99) [$ns node]
$node_(99) set X_ 413
$node_(99) set Y_ 67
$node_(99) set Z_ 0.0
$ns initial_node_pos $node_(99) 20


# Setup traffic flow between nodes
# UDP connections between nodes 0,1,10,20,2 and node_(45)


#================================
#            TRAFFIC
#================================

(CBR-UDP)
set udp0 [new Agent/UDP]
$ns attach-agent $node_(0) $udp0
set null0 [new Agent/Null]
$ns attach-agent $node_(45) $null0
```

```
$ns connect $udp0 $null0
$udp0 set fid_ 0
set cbr0 [new Application/Traffic/CBR]
$cbr0 attach-agent $udp0
$cbr0 set type_ CBR
$udp0 set packetSize_  120
$cbr0 set packet_size_ 100
$cbr0 set rate_ 100Kb
##$cbr0 set random_ false
$ns at 10.0 "$cbr0 start"
$ns at 20.0 "$cbr0 stop"
```

(CBR-UDP)

```
set udp1 [new Agent/UDP]
$ns attach-agent $node_(10) $udp1

$ns connect $udp1 $null0
$udp1 set fid_ 1
set cbr1 [new Application/Traffic/CBR]
$cbr1 attach-agent $udp1
$cbr1 set type_ CBR
$udp1 set packetSize_  120
$cbr1 set packet_size_ 100  ###packet size
$cbr1 set rate_ 100Kb
##$cbr1 set random_ false
$ns at 10.0 "$cbr1 start"  ###trafffic starting time
$ns at 20.0 "$cbr1 stop"   ###traffic stop
```

(CBR-UDP)

```
set udp2 [new Agent/UDP]
$ns attach-agent $node_(1) $udp2

$ns connect $udp2 $null0
$udp2 set fid_ 2
set cbr2 [new Application/Traffic/CBR]
$cbr2 attach-agent $udp2
$cbr2 set type_ CBR
$udp2 set packetSize_  120
$cbr2 set packet_size_ 100
$cbr2 set rate_ 100Kb
##$cbr2 set random_ false
$ns at 10.0 "$cbr2 start"
$ns at 20.0 "$cbr2 stop"
```

```
(CBR-UDP)
set udp4 [new Agent/UDP]
$ns attach-agent $node_(20) $udp4

$ns connect $udp4 $null0
$udp4 set fid_ 4
set cbr4 [new Application/Traffic/CBR]
$cbr4 attach-agent $udp4
$cbr4 set type_ CBR
$udp4 set packetSize_  120
$cbr4 set packet_size_ 100
$cbr4 set rate_ 100Kb
##$cbr4 set random_ false
$ns at 10.0 "$cbr4 start"
$ns at 20.0 "$cbr4 stop"


(CBR-UDP)
set udp5 [new Agent/UDP]
$ns attach-agent $node_(2) $udp5

$ns connect $udp5 $null0
$udp5 set fid_ 5
set cbr5 [new Application/Traffic/CBR]
$cbr5 attach-agent $udp5
$cbr5 set type_ CBR
$udp5 set packetSize_  120
$cbr5 set packet_size_ 100
$cbr5 set rate_ 100Kb
##$cbr4 set random_ false
$ns at 10.0 "$cbr5 start"
$ns at 20.0 "$cbr5 stop"
#=================================
#
#=================================

# Define node initial position in nam


# Telling nodes when the simulation ends
for {set i 0 } {$i < $val(nn) } {incr i} {
$ns at $val(finish) "$node_($i) reset";
}

# Ending nam and the simulation
```

```
$ns at $val(finish) "$ns nam-end-wireless $val(finish)"
$ns at $val(finish) "finish"
$ns at 40.01 "puts \"ns EXITING...\" ; $ns halt"

puts "Starting Simulation..."
$ns run
```

A.2 NS-2 Modifications (TCL)

Set routing protocol:

```
set val(rp)             StaticRT                ;# routing protocol
$ns node-config -adhocRouting $val(rp) \
```

Set data rate(example):

```
Mac/802_11 set basicRate_ 200kb // this is for control packets

Mac/802_11 set dataRate_ 100kb //rate for data packets
```

MICA 2 mote Energy parameters configuration:

```
# Configure nodes

        $ns node-config
                        -energyModel EnergyModel \
                        -idlePower 0.0135 \
                        -rxPower 0.0135 \
                        -txPower 0.02475 \
                        -sleepPower 0.00 \
                         -transitionPower 0.0 \
                        -transitionTime 0.0 \
                        -initialEnergy 2
```

IEEE 802.11 Radio Parameters Configuration:

```
Antenna/OmniAntenna set X_ 0
Antenna/OmniAntenna set Y_ 0
Antenna/OmniAntenna set Z_ 1.5
Antenna/OmniAntenna set Gt_ 1.0
Antenna/OmniAntenna set Gr_ 1.0



Phy/WirelessPhy set CPThresh_ 10.0
```

```
Phy/WirelessPhy set CSThresh_   2.13643e-07
Phy/WirelessPhy set RXThresh_   2.13643e-07
Phy/WirelessPhy set bandwidth_ 2e6
Phy/WirelessPhy set Pt_ 0.28183815
Phy/WirelessPhy set freq_ 914e+6
Phy/WirelessPhy set L_ 1.0
```

GRID Topology generation:

```
set val(x)          500             ;# X dimension of topography
set val(y)          500             ;# Y dimension of topography
```

Node 2 creation example:

```
set node_(2) [$ns node]
$node_(2) set X_ 233
$node_(2) set Y_ 207
$node_(2) set Z_ 0.0
$ns initial_node_pos $node_(2) 20
```

Setting Node Buffer maximum size in packets:

```
set val(ifq)          Queue/DropTail/PriQueue    ;# interface queue type
set val(ifqlen)        50                 ;# max packet in ifq
```

Setting Traffic pattern ::

```
set udp5 [new Agent/UDP] ### create UDP agent
$ns attach-agent $node_(2) $udp5  ###and attach this agent to node 2
set null0 [new Agent/LossMonitor] ### crate a sink, loss monitor object that can track
##losses
$ns attach-agent $node_(45) $null0 ##node 45 is set as the sink node
$ns connect $udp0 $null0
set cbr5 [new Application/Traffic/CBR]  ##create a traffic generator
$cbr5 attach-agent $udp5  ### and attach it to udp5
$cbr5 set type_ CBR
$udp5 set packetSize_  120
$cbr5 set packet_size_ 100; ### packet size is set to 100bytes
$cbr5 set rate_ 10Kb ;  ### cbr rate is set to 10kbps
##$cbr4 set random_ false
$ns at 10.0 "$cbr5 start" ####traffic is generated between 10 and 20 sec
$ns at 20.0 "$cbr5 stop"
```

Disable RTS/CTS Mechanism:

```
Mac/802_11 set RTSThreshold_  3000              ;# bytes
```

Setting of MAC maximum number of retransmissions :

```
Mac/802_11 set ShortRetryLimit_      7            ;# retransmittions
Mac/802_11 set LongRetryLimit_       4            ;# retransmissions
```

Setting node's buffer size:

```
set val(ifqlen)        5                  ;# max packet in ifq
```

Setting CWmin:

```
Mac/802_11 set CWMin_ 127       ;# could be any value less than the CWmax.
```

A.3 Sample C++ Code for Implementation of a new Static Routing Agent (StaticRT).

We implement the new routing agent by modifying codes from DSDV source code in ns-2. As DSDV in ns-2 defined two classes, the static routing agent also uses two classes.

```
classStaticRTable; //routingtable
class Static_ent;  // routing table entry
```

Routing Table Class.

Routing table is a class that store routing table entries. Each entry should have at least dst, next-hop and metric. Often in DSDV or AODV, it has sequence number to determine if it is a newer routing message. However, for static routing, this is unnecessary. To use this new agent, change opt(rt) from AODV to StaticRT in wireless simulation script.

In almost every routing protocol design, packet queue is used to store those packets with valid dst but without found routes yet. This is unnecessary for static routing design. So, the router will no longer need to have queues for each neighbors, we remove IP layer queues in "PacketQueue" member variable in class "Static_ent".

Routing Agent Class.

The most important task for a routing agent is to detect link and refresh routing table with special routing signaling messages such as periodic Advertisements. However, in static routing, there is no such a burden. Only two things need to be addressed:

1) Start-up function, called in the command "start-Staticrt". In this function, we call makeRoutingtable function. routing information will be "fscanf" from a file and adding to the tables. Here, we are going to use "fscanf" to read three parameters. "int32" "int32" "uint16",to represent dst, hop, metric. The filename must be specified . Different nodes read different sections of the routing table file.

The special format of this file is:

<src><dst> <next-hop><number of hops>
0 3 1 3
.....

So, only those entries where 'src' addr is equal to myaddr will be read.
The auto-generation of routing table could be easily implemented by an algorithm from topology file

2) Packets Forwarding functions: according to the routing table, also need to handle "broadcast" packets.
"Forwarding" means determining the next-hop of the packet. Once this field of the header is set, it is done. The agent's interfaces to MAC are complex:
1) Message interface: packets going to mac with next-hop set, ready for layer 2 transmission.
2) Function interface: once link of loss is detected, the MAC will "callback" (inform) routing agent to do correspondingly. In this Static Routing agent, this would be dummy. we do nothing in "mac_callback" function.

Interface (Linkage) to Otcl.

When a wireless node is created, its default routing agent is specified in the arguments, such as opt(rt). In /tcl/lib/ns-lib.tcl. the **"create-wireless-node"** procedure will check the routing protocol type. Here we can add a new type as "StaticRT" in the "switch" block. Also add "set ragent [$self create-StaticRT-agent $node]". For example,

```
 switch -exact $routingAgent_ {
            DSDV {
                    set ragent [$self create-dsdv-agent $node]
            }
            StaticRT {
                    set ragent [$self create-StaticRT-agent $node]
            }
   ...
   }
```

the "**create-StaticRT-agent**" procedure is also need to be defined. In this procedure, a "$self at 0.0 "$ragent start-StaticRT" " will be given to the routing agent. Therefore, we have to initialize routing table in the start-StaticRT command.

```
Simulator instproc create-StaticRT-agent { node } {
        # Create a fix-path routing agent for this node
        set ragent [new Agent/StaticRT]
        # Setup address (supports hier-addr) for dsdv agent
        # and mobilenode
        set addr [$node node-addr]
        $ragent addr $addr
        $ragent node $node
        if [Simulator set mobile_ip_] {
                $ragent port-dmux [$node demux]
```

```
        }
        $node addr $addr
        $node set ragent_ $ragent
        $self at 0.0 "$ragent start-StaticRT"    ;# start updates
        return $ragent
}
```

After this, all packets will come in the "recv" function entry, and the agent will just route packets according to the table.

Benefits of using it

Big benefits is that routing messages are suppressed, and link loss are also ignored. So, it provides a basic and fair platform for comparing MAC protocol performance with the **ALWAYS SAME**"route.!!!

This is important because routing protocol depends on MAC to propagate routing messages, If MAC is modified, the routing path could also be changed and the performance comparison is no longer accurate!

Source Code

There are two classes implemented. So, there are totally 4 files;

- Static Routing Agent Class
  - Staticrt.cc
  - Staticrt.h
- Table Class
  - Statictb.cc
  - Statictb.h

After adding those 4 files, we also changed ns-lib.tcl file and Makefile before we “make” ns-2.

A.4 Awk code used for sensor network statistics

Stats.txt, calculates end-to-end delay, average throughput, total packets sent, total packets received, total drops, total collisions, average traffic rate, max delay , reliability etc.

```
###stats.txt######
#
# Parse a ns2 wireless trace file and generate the following stats:
# - number of flows (senders)
# - time of simulation run
# - number of packets sent (at the Application)
# - number of packets received (at the Application)
# - number of packets dropped (at the Application)
# - number of collisions (802.11)
# - average delay
# - average throughput
# - average traffic rate (measured)
```

```
#
# Last updated: April 25, 2007 ,Antonis Antoniou
function average (array) {
   sum = 0;
   items = 0;
   for (i in array) {
      sum += array[i];
      items++;
   }
#   printf("DEBUG  sum is %d, items is %d\n", sum, items);
   if (sum == 0 || items == 0)
      return 0;
   else
      return sum / items;
}

function max( array ) {
   for (i in array) {
      if (array[i] > largest)
         largest = array[i];
   }
   return largest;
}

function min(array) {
   for (i in array) {
      if (0 == smallest)
         smallest = array[i];
      else if (array[i] < smallest)
         smallest = array[i];
   }
   return smallest;
}
BEGIN {
   total_packets_sent = 0;
   total_packets_received = 0;
   total_packets_dropped = 0;
   first_packet_sent = 0;
   last_packet_sent = 0;
   last_packet_received = 0;
}
{
   event = $1;
   time = $3;
   node = $9;
   type = $19;
```

```
    reason = $21;
    packetid = $41;

# strip leading and trailing _ from node
    #sub(/^_*/, "", node); for old trace format...
    #sub(/_*$/, "", node);

    if ( time < simulation_start || simulation_start == 0 )
        simulation_start = time;
    if ( time > simulation_end )
        simulation_end = time;

    if ( reason == "COL" )
            total_collisions++;


    if ( type == "AGT" ) {
        nodes[node] = node; # to count number of nodes

        if ( time < node_start_time[node] || node_start_time[node] == 0 )
            node_start_time[node] = time;

        if ( time > node_end_time[node] )
            node_end_time[node] = time;

        if ( event == "s" ) {
            flows[node] = node; # to count number of flows
            if ( time < first_packet_sent || first_packet_sent == 0 )
                first_packet_sent = time;
            if ( time > last_packet_sent )
                last_packet_sent = time;
            # rate
            packets_sent[node]++;
            total_packets_sent++;


            # delay
            pkt_start_time[packetid] = time;
        }
        else if ( event == "r" ) {
            if ( time > last_packet_received )
                last_packet_received = time;
            # throughput
            packets_received[node]++;
            total_packets_received++;
```

```
            # delay
            pkt_end_time[packetid] = time;
        }
        else if ( event == "D" ) {
            total_packets_dropped++;
#           pkt_end_time[packetid] = time; # EXPERIMENTAL
        }
    }
}
END {
    print "" > "throughput.dat";
    print "" > "rate.dat";
    number_flows = 0;
    for (i in flows)
        number_flows++;

    # find dropped packets
    if ( total_packets_sent != total_packets_received ) {
        printf("*Dropped Packets!\n\n");
        for ( packetid in pkt_start_time ) {
            if ( 0 == pkt_end_time[packetid] ) {
                total_packets_dropped++;
#               pkt_end_time[packetid] = simulation_end; # EXPERIMENTAL
            }
        }
    }

    for (i in nodes) {
        if ( packets_received[i] > 0 ) {
            end = node_end_time[i];
            start = node_start_time[i - number_flows];
            runtime = end - start;
            if ( runtime > 0 ) {
                throughput[i] = packets_received[i]/ runtime;
                printf("%d %f %f %d\n", i, start, end, throughput[i]) >> "throughput.dat";
            }
        }
        # rate - not very accurate
        if ( packets_sent[i] > 2 ) {
            end = node_end_time[i];
            start = node_start_time[i];
            runtime = end - start;
            if ( runtime > 0 ) {
                rate[i] = (packets_sent[i]) / runtime;
                printf("%d %f %f %d\n", i, start, end, rate[i]) >> "rate.dat";
            }
```

```
    }
  }

  # delay
  for ( pkt in pkt_end_time) {
    end = pkt_end_time[pkt];
    start = pkt_start_time[pkt];
    delta = end - start;
    if ( delta > 0 ) {
      delay[pkt] = delta;
      printf("%d %f %f %f\n", pkt, start, end, delta) >> "delay.dat";
    }
  }

  # offered load
  total_runtime = last_packet_sent - first_packet_sent;
  if ( total_runtime > 0 && total_packets_sent > 0)
    load = ((total_packets_sent)/total_runtime) / 8000000; # n=o overhead



    x=((total_packets_received)/(total_packets_sent)*100)

    #printf ("%d \n",total_collisions) >> "collisions.txt"
    #printf ("%5.2f \n",average(delay)) >> "delayrtmax.txt"
    #printf ("%5.1f \n",x) >> "reliabrtmax50.txt"

    # printf ("%d ",total_packets_sent) >>"reliabcw128.txt"
    # printf ("\n") >>"reliabcw128.txt"
}
```

A.5 Awk code used for differentiation of packet drops.

Specifically drops.txt, calculates MAC Errors, Buffer Overflows, Routing Errors for each sensor node, and total number of errors for corresponding type in the scenario.

```
### created by Antonis Antoniou
##      last updated:25 Apr 07
###   drops.txt

BEGIN {
      ###number of nodes
      n=100
      send=0
      recev=0
}
```

```
{
	# Trace line format:
		event = $1
		time = $2
		node_id = $9

		##drops due to buffer overflows
		if((event=="d")&&($19=="IFQ")&&($35=="cbr"))
		dropsIFQ[node_id]=dropsIFQ[node_id]+1;

		if (dropsIFQ[node_id]=="")
		dropsIFQ[node_id]=0

		if((event=="d")&&($21=="IFQ")&&($35=="cbr"))
		dropsRTRIFQ[node_id]=dropsRTRIFQ[node_id]+1;

		if (dropsRTRIFQ[node_id]=="")
		dropsRTRIFQ[node_id]=0

		##drops due to MAC failures
		if((event=="d")&&($21=="CBK")&&($35=="cbr")){
			dropsMAC[node_id]=dropsMAC[node_id]+1;

		}
				if (dropsMAC[node_id]=="")
		dropsMAC[node_id]=0

		if((event=="s")&&($19=="AGT")&&($35=="cbr"))
				send++;
		if((event=="r")&&($19=="AGT")&&($35=="cbr"))
				recev++;


}


END {
	# Compute drops for each node

	####output

	for (i in dropsIFQ) {
	totalIFQ=totalIFQ+dropsIFQ[i]
	}
```

```
	for (i in dropsMAC) {

	totalMAC=totalMAC+dropsMAC[i]
	}

	for (i in dropsRTRIFQ) {

	totalRTRIFQ=totalRTRIFQ+dropsRTRIFQ[i]
	}

	for (i=0; i<n; i++) {

		print("node",i, "IFQ drops",dropsIFQ[i]"\n")
		print("node",i, "MAC drops",dropsMAC[i]"\n")
		print("node",i, "RTRIFQ drops",dropsRTRIFQ[i]"\n")

	totaldrops=send-recev


		totalroute=totaldrops-(totalIFQ+totalMAC+totalRTRIFQ)
		}

		print("total drops",totaldrops "\n")
				print("total IFQ dr",totalIFQ "\n")
				print("total MAC dr",totalMAC "\n")
				print("total RTR IFQ dr",totalRTRIFQ "\n")
		print("total routing drops",totalroute "\n")

}
```

A.6 Awk code used for calculation of energy consumption.

Energy.txt is used to calculate energy consumption for each sensor node, the energy and node id for max consumption(bottleneck node), average consumption-total consumption for the network.

```
#####created by Antonis Antoniou
#####Last Modified 25 Apr 07
##### energy.txt

BEGIN {
	initialenergy = 2
	maxenergy=0
	n=100
	nodeid=999
}

{
```

```
	# Trace line format: energy
		event = $1
		time = $2
		if (event == "r" || event == "d" || event =="s"|| event=="f") {
		node_id = $9
		energy=$17
		}
		if (event=="N"){
		node_id = $5
		energy=$7
		}

	# Store remaining energy
	finalenergy[node_id]=energy
}
END {
	# Compute consumed energy for each node

	for (i in finalenergy) {

		consumenergy[i]=initialenergy-finalenergy[i]

		totalenergy +=consumenergy[i]
		if(maxenergy<consumenergy[i]){
		maxenergy=consumenergy[i]
		nodeid=i
		}
	}

	###compute average energy

	averagenergy=totalenergy/n


	####output

	for (i=0; i<n; i++) {
		print("node",i, consumenergy[i])
		}

		print("average",averagenergy)
		print("maximum",maxenergy,"node",nodeid)
		print("total energy",totalenergy)
}
```

A.7 Sample Matlab Code for the creation of the graphs.

```
load ('D:\thesis\simulations thesis\\delay5x10.data' );
load ('D:\thesis\simulations thesis\\delay5x30.data' );
load ('D:\thesis\simulations thesis\\delay5x50.data' );
load ('D:\thesis\simulations thesis\\delay5x80.data' );
load ('D:\thesis\simulations thesis\\delay5x100.data' );
load ('D:\thesis\simulations thesis\\delay5x150.data' );
load ('D:\thesis\simulations thesis\\delay5x250.data' );

time = delay5x10(:,1);
time1 = delay5x30(:,1);
time 2= delay5x50(:,1);
time 3= delay5x80(:,1);
time 4= delay5x100(:,1);
time 5= delay5x150(:,1);
time 6= delay5x250(:,1);

delay_1 = delay5x10(:,2);
delay_2 = delay5x30(:,2);
delay_3 = delay5x50(:,2);
delay_4 = delay5x80(:,2);
delay_5 = delay5x100(:,2);
delay_6= delay5x150(:,2);
delay_7= delay5x250(:,2);

figure
plot (time, delay_1, 'Color', 'red', 'LineWidth', 1.8, 'LineStyle', '-')
hold
plot (time1, delay_2, 'Color', 'blue', 'LineWidth', 1.8, 'LineStyle', '-')
plot (time2, delay_3, 'Color', 'green', 'LineWidth', 1.8, 'LineStyle', '-')
plot (time3, delay_4, 'Color', 'red', 'LineWidth', 1.8, 'LineStyle', ':')
plot (time4, delay_5, 'Color', 'blue', 'LineWidth', 1.8, 'LineStyle', ':')
plot (time5, delay_6, 'Color', 'green', 'LineWidth', 1.8, 'LineStyle', ':')
plot (time6, delay_7, 'Color', 'cyan', 'LineWidth', 1.8, 'LineStyle', '-')
axis([0 100 0 50]);
title('End-to-End Delay');
xlabel('Sensor Reporting rate(p/s)');
ylabel('Time (s)');
set(gca,'YGrid','on');
```

A.8 Full Thesis Material

All created thesis TCL files, awk files, simulation results, raw data(tracefiles) are included in “THESIS “ DVD, submitted to Supervisor.