

**TOPOLOGY CONTROL AND ENERGY CONSERVATION
IN
WIRELESS AD HOC SENSOR NETWORKS**

by

RAMASUBRAMANIAN K

Reg. No. 71203405012

of

Kumaraguru College of Technology, Coimbatore 641 006



P-1539

A PROJECT REPORT

Submitted to the

FACULTY OF INFORMATION AND COMMUNICATION ENGINEERING

*In partial fulfillment of the requirements
for the award of the degree*

of

MASTER OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

JUNE 2005

BONAFIDE CERTIFICATE

Certified that this project report entitled "TOPOLOGY CONTROL AND ENERGY CONSERVATION IN WIRELESS AD HOC SENSOR NETWORKS" is the bonafide work of **MR. K. RAMASUBRAMANIAN**, who carried out the research under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.



GUIDE

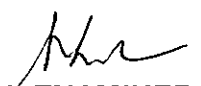


HEAD OF THE DEPARTMENT

The candidate with **University Register No. 71203405012** was examined by us in the project viva-voce examination held on 25/06/2005



INTERNAL EXAMINER



EXTERNAL EXAMINER

Abstract

Topology control for ad hoc networks aims to maintain a specified topology by controlling which links should be included in the network to achieve a set of network-wide or session-specific objectives such as reducing interference or probability of detection, reduction of energy consumption, increase of the effective network capacity and reducing end to end delay. The primary method of accomplishing topology control is by adjusting the transmission powers of the network nodes.

The topology of an ad hoc network has a significant impact on its performance in that a dense topology may induce high interference and low capacity, while a sparse topology is vulnerable to link failure and network partitioning. *Topology control* aims to maintain a topology that optimizes network performance while minimizing energy consumption. Existing topology control algorithms utilize either a purely centralized or a purely distributed approach. A centralized approach, although capable to achieve strong connectivity suffers from scalability problems. In contrast, a distributed approach, although scalable, lacks strong connectivity guarantees. This hybrid topology control achieves scalability using hybrid *clustering* technique while minimizing energy consumption. Also the thesis examines the performance as well as energy consumption issues of a wireless sensor network providing periodic data from a sensing field to a remote receiver. It distinguishes between two types of sensor organizations, one with a single layer of identical sensors (homogeneous) and one with an additional overlay of fewer but more powerful sensors (heterogeneous). The energy consumption and estimated lifetime based on a clustering mechanism with varying parameters related to the sensing field, e.g., size to the sink and distance are also considered. Quantification of the optimal number of clusters based on the proposed model and shows how to allocate the energy between different layers.

ஆய்வுச் சுருக்கம்

தானியங்கி மிண்ணணு சாதனங்களின் இட அமைப்பு (topology) இரண்டு வகைப்படுத்தப் படுகின்றன. அவைகளில் அடர்ந்த இட அமைப்பு (dense topology) குறுக்கீடுகளாலும், குறைந்த அளவு சாதனங்களாலும் பாதிக்கப்படுகின்றன. பாவலான இட அமைப்பு (sparse topology) துண்டிக்கப்பட்ட தொடர்புகளாலும், பிரிக்கப்பட்ட சாதனங்களின் தொகுப்புகளாலும் பாதிப்பிற்குள்ளாகின்றன. இட அமைப்பின் முக்கிய நோக்கமே, கூட்டமைப்பின் திறனை அதிகப்படுத்தியும், குறைவான மின்கல சக்தியை (energy conservation) உபயோகப்படுத்துவதுமே ஆகும்.

இதுவரை கண்டுள்ள முறைகளில் மையப்படுத்தப்பட்ட முறையினால் (centralized topology) துண்டிக்கப்படாத சேவையை மட்டுமே அளிக்க முடிகின்றது. இதனால் அதிக சாதனங்களை கட்டுப்படுத்த முடிவதில்லை. இரண்டாம் முறையான பாவலாக்கப்பட்ட இட அமைப்பில் (distributed topology) அதிக சாதனங்களை இணைக்க முடிகின்ற வேளையில், ஒரே தொகுப்பு (cluster) பல தொகுப்புகளாக பிரிவதை தடுக்க முடிவதில்லை. இந்த ஆய்வறிக்கையின் நோக்கமே தொகுக்கப்பட்ட கூட்டமைப்பில் அதிக சாதனங்களை இணைக்க முயல்வதும், குறைந்த மின்கல சக்தியை உபயோகித்து தகவல்களை தொலைவிலுள்ள மையப்படுத்தப்பட்ட தானியங்கி சேமிப்பானுக்கு (remote collector) அனுப்புவது ஆகும்.

இந்த ஆய்வறிக்கையில் இரு விதமான தானியங்கி சாதனங்கள் கையாளப்படுகின்றன. அவையாவன, அனைத்து சாதனங்களும் ஒரே விதமான, ஒரே அளவான மின்கல சக்தியுடையவை (homogenous) மற்றும் வெவ்வேறு விதமான அளவுகளும், வெவ்வேறு அளவான மின்கல சக்தியையும் (heterogeneous) உடையவை. உபயோகிக்கும் மின்கல சக்தியும், திட்டமிட்ட கால அளவும் பல்வேறு அளவீடுகளால் (parameters) மதிப்பிடப்பட்டன. குறைந்த அளவு சாதனங்களின் தொகுப்பு துல்லியமான அளவீடுகளால் மதிப்பிடப்பட்டன. மற்றும் சாதனங்களுக்கு செலவழியும் மின்கல சக்தியும் வெவ்வேறு நிலைகளில் நிரூபிக்கப்பட்ட சமன்பாடுகளால் (formula) அளிக்கப்படுவதும் இவ் ஆய்வறிக்கையில் செயல்படுத்தப்பட்டுள்ளன.

ACKNOWLEDGEMENTS

All that have started well will end well and at this instant of time I would like to thank the people who were directly and indirectly instrumental in initiating me to do this course.

I would like begin by thanking **Dr. K. K. Padmanabhan, Ph.D., Principal** for providing the necessary facilities to complete my thesis.

I take this opportunity to thank **Dr. S. Thangasamy, Ph.D., Head of the Department, Computer Science and Engineering**, for his precious suggestions.

I register my hearty appreciations to **Ms. L. S. Jayashree M.E., Ph.D., my thesis advisor** and mentor. I thank her for her wonderful lectures on *Computer Networks and Engineering Management* and *Mobile Computing* that inspired my interests in this field. I thank her for her support, encouragement and ideas. I thank her for the countless hours she has spent with me, discussing everything from research to academic choices.

I thank all project committee members for their comments and advice during the reviews. Special thanks to **Mr. R. Dinesh, M.S. (Wisconsin), Assistant Professor**, Department of Computer Science and Engineering, for arranging the brain storming project review sessions.

I also owe a huge debt of gratitude to my **parents, sisters** and **friends**. They have always been there whenever I needed them. They have provided me far more than I could ever give them.

TABLE OF CONTENTS

Contents	Page
Abstract	iii
List of Figures	viii
List of Abbreviations	ix
1. Introduction	
1.1. Wireless Ad hoc Networks overview	1
1.2. Mobile Ad hoc Networks	1
1.3. Wireless Sensor Networks	3
1.4. Topology Control Techniques for Wireless Ad hoc Sensor Networks	5
1.5. Energy Conservation Schemes for Ad hoc Networks	6
1.6. The current status of the problem taken up	6
1.7. Relevance and importance of the topic	7
2. Literature Survey	
2.1. Centralized Topology Control Algorithms	8
2.2. Distributed Topology Control Algorithms	9
2.3. Clustering Techniques	12

2.4. Energy Conservation Algorithms for Wireless Sensor Networks	15
3. Line of Attack	17
4. Details of the Methodology employed	
4.1 Clusters Formation and Cluster Heads Selection	20
4.2 Network Model and Assumptions	21
4.3 Energy Allocation	22
4.4 Collection of Data and Transmission and Changing Roles	23
5. Results obtained	24
6. Conclusions and Future outlook	28
Appendices	
Appendix 1	29
Appendix 2	44
References	47

LIST OF FIGURES

Figure	Caption	Page No.
1.	Mobile Adhoc Networks	
2.	Biological Systems with Sensors	
3.	Environmental Analysis	
4.	Centralized Topology Control Architecture	
5.	Distributed Topology Control Architecture	
6.	Proposed Sensor Network Architecture	
7.	Heterogeneous Network with random deployment sensors	
8.	Optimal clustering for	
	8.1 Alpha=2, L=50, D=100 and alpha=4, L=5, D=10	
	8.2 Alpha=2, L=500, D=1000	
	8.3 Alpha=2, L=5, D=1	
9.	Cluster Formation	
10.	Cluster Head Election	
11.	Active Cluster Head Election	
12.	Clusters Formation & Cluster Heads Election (both active & inactive)	
13.	Data Transmission from Head to Collector	

LIST OF ABBREVIATIONS

MANET	Mobile Ad Hoc NETWORK
CBRNE	Chemical, Biological, Radiological, Nuclear, and Explosive
MST	Minimum Spanning Tree
LINT	Local Information No Topology
RNG	Relative Neighborhood Graph
LILT	Local Information Link State Topology
NTC	Novel Topology Control Algorithms
min-R	Minimum Radius Graph
Dist-RNG	Distributed Relative Neighborhood Graph
DNTC	Distributed Novel Topology Control Algorithms
COMPOW	Common Power Level
LEACH	Low Energy Adaptive Clustering Hierarchy
ADB	Adaptive Dynamic Backbone
HID	Highest-degree Algorithm
LID	Lowest-id Algorithm
LCC	Least Clustering Change Algorithm
RCC	Random Competition Based Algorithm

CHAPTER 1

INTRODUCTION

1.1 WIRELESS AD HOC NETWORKS OVERVIEW

A wireless ad hoc network is a collection of autonomous nodes or terminals that communicate with each other by forming a multihop radio network and maintaining connectivity in a decentralized manner. Since the nodes communicate over wireless links, they have to contend with the effects of radio communication, such as noise, fading, and interference. In addition, the links typically have less bandwidth than in a wired network. Each node in a wireless ad hoc network functions as both a host and a router, and the control of the network is distributed among the nodes. The network topology is in general dynamic, because the connectivity among the nodes may vary with time due to node departures, new node arrivals, and the possibility of having mobile nodes. Hence, there is a need for efficient routing protocols to allow the nodes to communicate over multihop paths consisting of possibly several links in a way that does not use any more of the network "resources" than necessary. Yet, research in the area of ad hoc networking is receiving much attention from academia, industry, and government. Since these networks pose many complex issues, there are many open problems for research and opportunities for making significant contributions.

1.2 MOBILE AD HOC NETWORKS

In the next generation of wireless communication systems, there will be a need for the rapid deployment of independent mobile users. Significant examples include establishing survivable, efficient, dynamic communication for emergency/rescue operations, disaster relief efforts, and military networks. Such network scenarios cannot rely on centralized and organized connectivity, and can

be conceived as applications of **Mobile Ad Hoc NETWORKS (MANET)**. A MANET is an autonomous collection of mobile users that communicate over relatively bandwidth constrained wireless links. Since the nodes are mobile, the network topology may change rapidly and unpredictably over time. The network is decentralized, where all network activity including discovering the topology and delivering messages must be executed by the nodes themselves, i.e., routing functionality will be incorporated into mobile nodes.

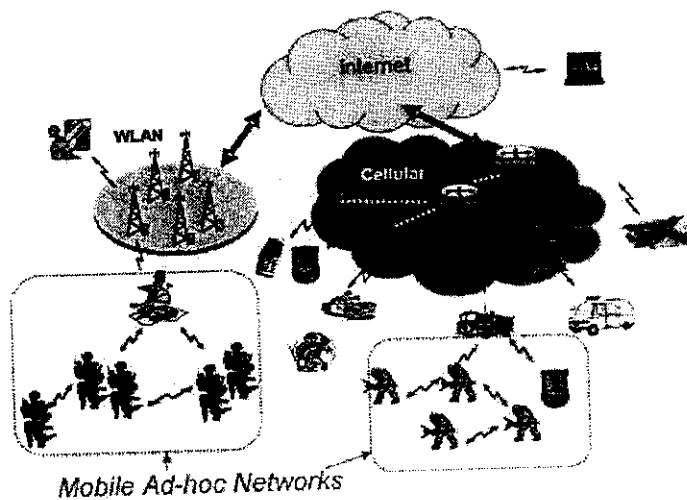


Fig 1. Mobile Ad Hoc Networks

The set of applications for MANETs is diverse, ranging from small, static networks that are constrained by power sources, to large-scale, mobile, highly dynamic networks. The design of network protocols for these networks is a complex issue. Regardless of the application, MANETs need efficient distributed algorithms to determine network organization, link scheduling, and routing. However, determining viable routing paths and delivering messages in a decentralized environment where network topology fluctuates is not a well-defined problem. While the shortest path (based on a given cost function) from a source to a destination in a static network is usually the optimal route, this idea is not easily extended to MANETs. Factors such as variable wireless link quality, propagation path loss, fading, multiuser interference, power expended, and topological changes, become relevant issues. The network should be able to

adaptively alter the routing paths to alleviate any of these effects. Moreover, in a military environment, preservation of security, latency, reliability, intentional jamming, and recovery from failure are significant concerns. Military networks are designed to maintain a low probability of intercept and/or a low probability of detection. Hence, nodes prefer to radiate as little power as necessary and transmit as infrequently as possible, thus decreasing the probability of detection or interception. A lapse in any of these requirements may degrade the performance and dependability of the network.

1.3 WIRELESS SENSOR NETWORKS

A wireless ad hoc sensor network consists of a number of sensors spread across a geographical area. Each sensor has wireless communication capability and some level of intelligence for signal processing and networking of the data. Some examples of wireless ad hoc sensor networks are the following:

1. Military sensor networks to detect and gain as much information as possible about enemy movements, explosions, and other phenomena of interest.
2. Sensor networks to detect and characterize Chemical, Biological, Radiological, Nuclear, and Explosive (CBRNE) attacks and material.

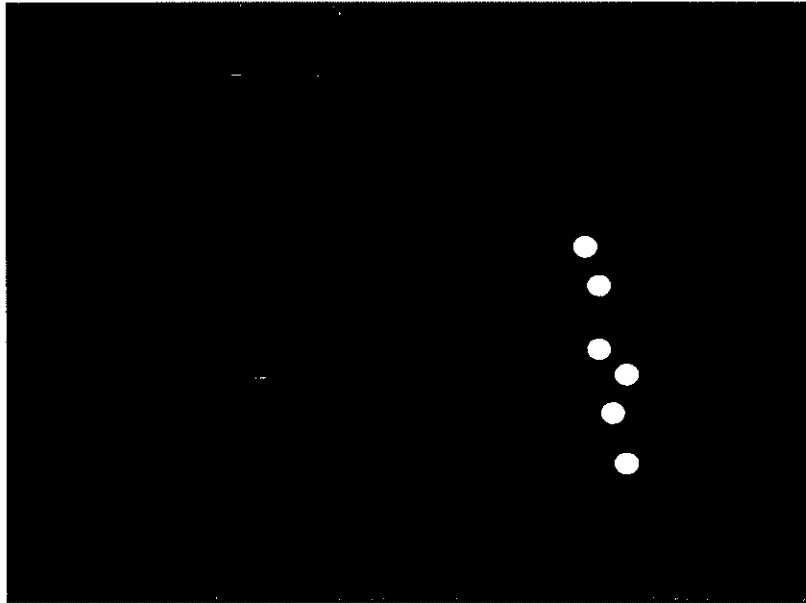


Fig 2. Biological Systems with Sensors

3. Sensor networks to detect and monitor environmental changes in plains, forests, oceans, etc.

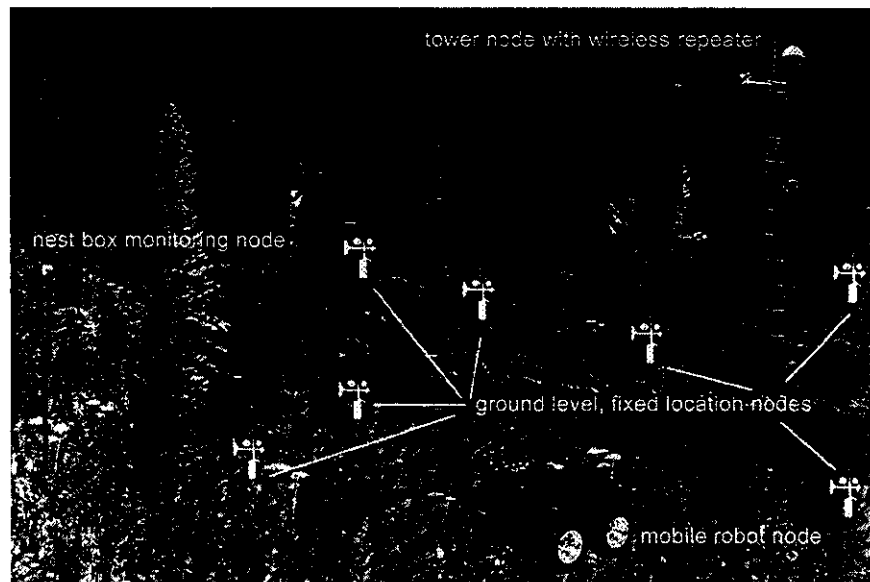


Fig 3. Environmental Analysis

4. Wireless traffic sensor networks to monitor vehicle traffic on highways or in congested parts of a city.
5. Wireless surveillance sensor networks for providing security in shopping malls, parking garages, and other facilities.
6. Wireless parking lot sensor networks to determine which spots are occupied and which are free.

The above list suggests that wireless ad hoc sensor networks offer certain capabilities and enhancements in operational efficiency in civilian applications as well as assist in the national effort to increase alertness to potential terrorist threats.

1.4 TOPOLOGY CONTROL TECHNIQUES FOR WIRELESS SENSOR AD HOC NETWORKS

Topology control in a sensor network balances load on sensor nodes and increases network scalability and lifetime. Topology control for ad hoc networks aims to maintain a specified topology by controlling which links should be included in the network to achieve a set of network-wide or session-specific objectives such as reducing interference or probability of detection, reducing energy consumption, increasing the effective network capacity, and reducing end-to-end delay. The primary method of accomplishing topology control is by adjusting the transmission powers of the network nodes. Several topology control algorithms based on transmission power adjustment have been proposed, where topology control is defined as the problem of assigning transmission powers to the nodes so that the resulting topology achieves certain connectivity properties and so that some function of the transmission powers is optimized. Centralized algorithms [4], [5], [6] rely on the assumption that the locations of all of the nodes are known by a central entity in order to calculate the transmission powers that result in a topology with strong connectivity. However, these algorithms are not scalable for large ad hoc networks where

excessive amounts of information would need to be collected by a central entity. Distributed algorithms [6], [7], [8], on the other hand, are generally scalable and adaptive to mobility due to the fact that each node relies on local information collected from nearby nodes to autonomously compute its appropriate transmission power.

1.5 ENERGY CONSERVATION SCHEMES FOR AD HOC NETWORKS

Reducing energy consumption is a key design objective of ad hoc networks. Lettieri et al [1] show that efficient energy design of wireless networks can be approached systematically by classifying low energy design technologies into three levels: hardware level, intra-node power management and inter-node power management. At the hardware level, the focus is on reducing the power consumption of individual circuits and components. At the intra-node power management level, the objective is achieved by putting certain components in a single device (e.g. CPU, LCD, disk, etc) to operate in low power or even sleep at a proper time. At the inter-node power management level, the layers of the network stack are studied. Energy efficient algorithms are embedded in the protocols of each layer. Energy is conserved by proper coordination of multiple nodes. One major objective of topology control as discussed in [2] is to conserve energy at each node. By Lettieri et al's classification, this is a inter-node power management scheme. Topology control is achieved by assigning a proper transmission power at each node. In [3], integrated dual approach was discussed to reduce power consumption for IEEE802.11 Wireless LAN. This paper dealt only the power reduction in idle mode.

1.6 THE CURRENT STATUS OF THE PROBLEM TAKEN UP

A constant update or cock-driven sensor network was analyzed with a heterogeneous organization. This research described how dynamic clusters are formed, presented a way of determining the optimal number of clusters for a given set of parameters, and showed numerically results. Only a limited number

of aspects of a sensor network have been considered here. Query-driven and event-driven type of sensor network are to be considered in future research. The possibility of several collectors located in different places should also be considered. Another important issue to be explored is a heterogeneous network model where the difference between the sensors is not only the difference in available energy, but also in their processing capabilities and thus the consideration of energy consumption in data processing (compression, fusion etc.).

1.7 RELEVANCE AND IMPORTANCE OF THE TOPIC

Ad hoc sensor networks do not rely on existing infrastructure and are self-organizing. They can be rapidly deployed to provide robust communication in a variety of hostile environments. This makes wireless ad hoc sensor networks appropriate for providing tactical communication for military, law enforcement and emergency response efforts. Once deployed, they should require minimal external support for their functioning. Wireless sensor networks pose many new challenges primarily because the sensor nodes are resource constrained. Energy is constrained by the limited battery power in sensor nodes. The form factor is an important node design consideration for easy operability and ad-hoc deployment of these nodes, which limit the amount of resources to be put in a node. Thus the protocols and applications designed for sensor networks should be highly efficient and optimized in terms of the resources they consume. Also communication is an important consideration because ad-hoc is an infrastructure less network. So the nodes can act both as router and node and also it decides which links should be included and which links should be excluded in the communication network. Topology control is used for data dissemination and aggregation.

CHAPTER 2

LITERATURE SURVEY

2.1 CENTRALIZED TOPOLOGY CONTROL ALGORITHMS

In this, a particular network node is responsible for evaluating an optimum network topology based on the locations of the other network nodes.

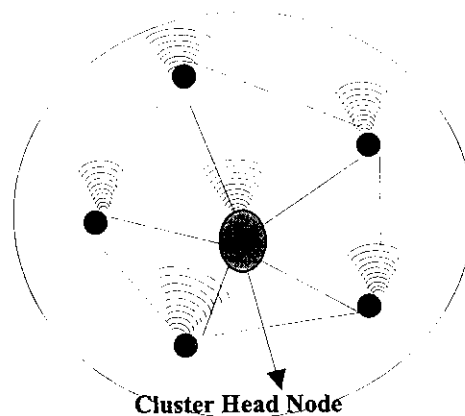


Fig 4. Centralized Topology Control Architecture

Chi-Fu Huang [9] presented a Brute-Force based Topology Formation which has minimum computational overheads. *C. Hou* [10] proposed two main algorithms, Kruskal's and Prim's algorithms for computing MST. MST is a sub graph of RNG. *Ramanathan* [11] forming the cluster by combining the nodes iteratively until the entire network is connected. Connect initialize's "N" clusters, one for each node. Node pairs are selected in an increasing order of their mutual distance. The transmission power of each node is increased until they are able to reach each other. *Hu L* [12], NTC algorithm modifies a Delaunay Triangulation (DT) graph to achieve the desired network capacity and network connectivity. The algorithm starts with a DT graph, which provides a reliable

backbone topology. The edges are sorted in the order of decreasing length. All edges longer than length R are removed from the graph.

Esther Jennings and Clayton [13] algorithm assumes that nodes have common transmission power. The algorithm searches for the smallest power to keep the network connected. The connectivity of the network is checked after every step increase in power. The search operation for checking the network connectivity incurs a cost of $O(N^2)$. The total cost of the algorithm is estimated to be $O(N^2P)$, where P is the number of power increments. The algorithm assumes that all network devices operate on a common power level, which is an unrealistic assumption. Choosing a common power level also increases the power redundancy in the network as it may introduce side-effect edges in the connected components. *Rosales-Hain* [14], Biconn-Augment adapts the connected network to a bi-connected network. Biconnectivity ensures that there is always at least two links connecting a node to every other node, hence the topology will remain connected even after a node goes down. In order to achieve biconnectivity the algorithm first tries to identify the bi-connected components using Depth First Search (DFS). The nodes are then selected in increasing order of their mutual distance and joined only if they are in different bi-connected components. This process is repeated until the network is bi-connected. A post processing phase similar to Connect ensures per-node power minimization. The algorithm has a computation cost of $O(N^2 \log N)$.

2.2 DISTRIBUTED TOPOLOGY CONTROL ALGORITHMS

Connectivity aware Distributed topology control algorithms try to adjust the neighbor count to maintain connectivity and stability in the network. Each node maintains neighbor information. In this, Local Information No Topology (LINT) [15] uses locally available information collected by

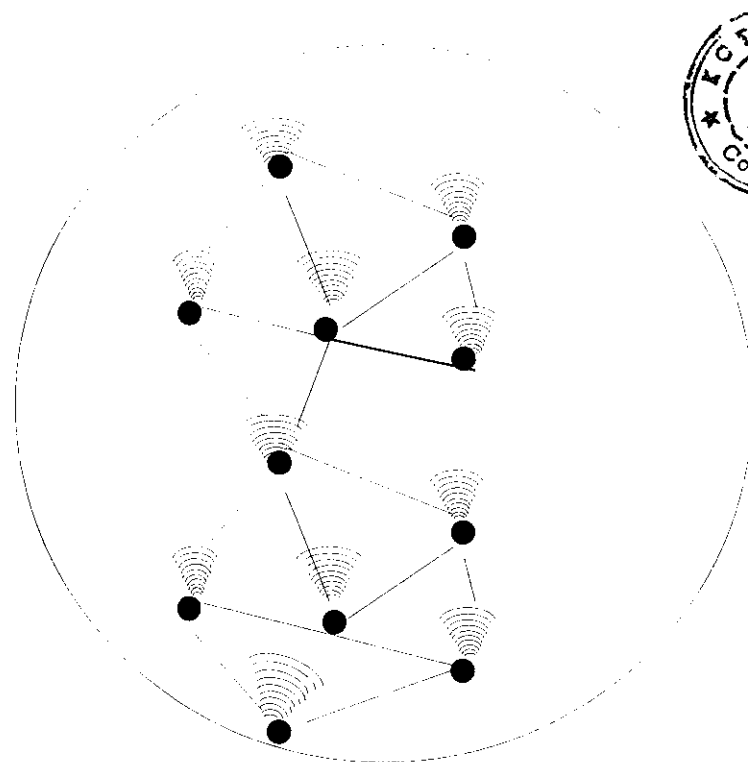


Fig 4. Centralized Topology Control Architecture

routing protocols to keep the degree of s bound. All network nodes are configured with three parameters, the desired node degree d_d , a high threshold of the node degree d_h and a low threshold of node degree d_l . A node periodically checks the number of its active s . If the degree is greater than d_d , the node reduces its operational power. If the degree is less than d_d the node increases its operational power. If neither is true then no action is taken. The increase/decrease in transmission power is bounded by maximum/minimum power settings of the radio. The minimum overhead of checking the count of an N node network is in the order of $O(N)$ as every node needs to check its count at least once. Local Information Link State Topology (LILT) [16] exploits the global topology information for recognizing and repairing network partitions. There are

two main parts of LILT, Reduction Protocol (NRP) and Addition Protocol (NAP). NRP reduces the transmission power to maintain the node degree around a certain configured value whereas NAP increases the transmission power to establish additional links. The overhead of checking the count of a N node network is in the order of $O(N)$. *Chi-Fu Huang* [17] executing Dist-RNG a node N_i grows its transmission power until the nearest N_j is found in the uncovered region. Once a node is found, an edge $(N_i; N_j)$ is added to RNG. If there are more than one node then a corresponding edge for each of them is added to the RNG. By using the new found N_j an angle θ_j is calculated. This angle θ_j defines a cone that spans the area covered by N_j . Θ donates the set of angles, which define cones that jointly span the covered regions. Initially Θ is set to zero. θ_j is merged with Θ and the whole process is repeated until the entire region (Θ) around N_i is spanned or the maximum power is reached. The computational overhead of the algorithm is in the order of $O(N \log N)$ for a N node network. *Hu L* [18], Dist-NTC all nodes broadcast their own existence and collect information within their maximum transmission range R . Furthermore, every node finds adjacent DTs within R . Each node keeps only a fixed number of shortest edges and informs other nodes about abandoned edges and rejects any edge abandoned by a . Network nodes that have less than specified edges are classified as 'active'. Every active node joins a 'distributed matching procedure' and finds the nearest active and sends a request packet to it and waits for a reply. Network links, which may arise due to the distributed matching procedure, are called non-basic links (L). If the receiver acknowledges the reply then an edge is added. Each active node rejects the request from any other node except the nearest active node. The process is repeated until the number of adjacent edges is equal to the fixed value or no more nodes (in range) are available for matching. Non-active nodes do not join the distributed matching procedure. An active node changes into a non-active one if it has enough links or cannot find an active node within its maximum transmission range. The overall communication complexity of the algorithm for a N node network is $O(NL)$.

Capacity aware Distributed topology control algorithm takes into account that the network nodes cause interference which impacts the communication of other nodes in the vicinity. *Jilei Liu and Baochun Li*, [19] distributed topology control algorithm, which, maintains a specific Contention Index (CI). CI is a product of node density (ρ) (number of nodes per unit area) and area size (A). In order to maintain global CI, all nodes try to keep the local CI bound to a specific value. The local estimate of contention index at the i^{th} node is evaluated from the number of one hop neighbors ($s(N_i)$). Each node looks up a particular optimization table to determine whether it is operating around an optimal value of CI. The optimal values of CI are evaluated beforehand through simulations and hard-coded in the network nodes. A node adjusts its transmission range to keep the CI value bound. MobileGrid uses CI values between 0.5 and 1.0 to maximize the network capacity. This approach is similar to LINT and hence checking CI is performed at least once which incurs a minimum overhead in the order of $O(N)$. *Narayanaswamy S.* [20], executed COMPOW which operates on the smallest power level to reach maximum network connectivity. COMPOW maintains a routing table at different power levels. Each routing table exchanges link state updates at different powers to generate the route information. The optimum power of a node is the smallest power level whose routing table has the same number of entries as that of a routing table at the maximum power level. COMPOW reduces the transmission power redundancy and interference by selecting the maximum operational power settings to reach the furthest node. In the worst case scenario the algorithm will incur a cost of $O(PN)$, where P are the number of power levels and N are the number of network nodes.

2.3 CLUSTERING TECHNIQUES

Clustering is a method by which nodes are hierarchically organized on their relative proximity to one another. The network is divided into clusters of nodes. Some nodes assume the role of cluster head, thus playing a more active role in keeping track of the topology and feasible routing paths. The role of cluster head

is a temporary role, which changes dynamically, as the topology or other factors affecting it to change. There are some existing mechanisms for cluster formation,

- (i) node-id based
- (ii) degree based,
- (iii) mobility based,

each focusing on a particular characteristic of the network. It is claimed that compared to conventional routing protocols, the cluster-based approach incurs lower overhead during topology updates and also has quicker convergence. The effectiveness of this approach also lies in the fact that existing routing protocols that can be directly applied to the network replacing the nodes by clusters.

Low Energy Adaptive Clustering Hierarchy (LEACH) [21] combines the ideas of energy-efficient cluster based routing and media access together with application specific data aggregation to achieve good performance in terms of system lifetime, latency, and application perceived quality LEACH includes a new, distributed cluster formation technique that enable self organization of large number of nodes, algorithms for adapting clusters and rotating cluster head positions to evenly distribute the energy load among all the nodes.

Adaptive Dynamic Backbone (ADB) [22] is a distributed clustering algorithm. This integrates the effectiveness of the flooding scheme and the efficiency of the tree-bases scheme for different environment conditions, within the same network. Highest-degree Algorithm(HID) [23] said each node broadcast beacon and receive acknowledgement to get the number of its neighbors. The node with maximum number of neighbors is chosen as a cluster head and any tie is broken by the unique node ids. The neighbors of a cluster head become members of that cluster and can no longer participate in the election process. Since no cluster heads are directly linked, only one cluster head is allowed per cluster. Any two nodes in a cluster are at most two-hops away since the cluster head is directly linked to each of its neighbors in the cluster. Basically, each node either becomes a cluster head or remains an ordinary node.

Lowest-id Algorithm(LID) [24] also as known as identifier based clustering. This algorithm assigns a unique id to each node and chooses the node with the minimum id as a cluster head. Thus, the ids of the neighbors of the cluster head will be higher than that of the cluster head. However, the cluster head will delegate its responsibility to the next node with the minimum id in its cluster. A node is called a gateway if it lies within the transmission range of two or more cluster heads. Gateway nodes are generally used for routing between clusters. Only gateway nodes can listen to the different nodes of the overlapping clusters that they lay.

Least Clustering Change Algorithm (LCC) [25] was presented by C. C. Chiang and M. Gerla. They start at the lowest-id cluster algorithm or highest-connectivity cluster algorithm to create initial clusters. When a non-cluster head node in cluster i move into a cluster j , no cluster head in cluster i and j will be changed(only cluster members are changed). When a non-cluster head node moves out its clusters and doesn't enter into any existing cluster, it becomes a new cluster head, forming a new cluster. When cluster head $C(i)$ from cluster i moves into the cluster j , it challenges the corresponding cluster head $C(j)$. Either $C(i)$ or $C(j)$ will give up its cluster head position according to lowest-id or highest-connectivity (or some other well defined priority scheme).

Previous research in clustering algorithms mainly focus on how to form clusters with a good shape such as minimum overlap of clusters, coverage etc. However, stability is also a serious problem to the real application, especially when clustering is used to support routing. For the hierarchical structure, stability of backbone nodes and local subnets are highly preferred.

The main idea of random competition based algorithm is that any node, which does not belong to any cluster, can initiate a cluster formation by broadcasting a packet to claim itself as a cluster head. The first node, which broadcast such

packet, will be elected as the cluster head and become members of this cluster. Cluster heads have to periodically broadcast a cluster head claim packet to maintain their clusters. Since there is a delay from when one node broadcasts its cluster head claim packet to when this packet is heard by its neighbors, several neighbor nodes may broadcast during this time period. To reduce such concurrent broadcasts, a random timer is introduced. Each node defers a random time before its cluster head claims. If it hears a cluster head claim during this random time, it then gives up this broadcast. In this scheme random timer is introduced to reduce conflicts when two nodes, which are within transmission range of each other broadcast simultaneously. Of course, the random timer cannot completely solve the concurrent broadcast problem. When the concurrent broadcasts happen, the node ID is used to solve the conflict and the node with lower ID will become the cluster head. When two cluster heads come into transmission range of each other, the one with lower ID will delegate the one with higher ID.

2.4 ENERGY CONSERVATION SCHEMES FOR AD HOC NETWORKS

Unbalanced energy consumption is an inherent problem in wireless sensor networks, and it is largely orthogonal to the general energy efficiency problem. For example, in a data gathering application, multihop wireless links are utilized to relay information to destination points called sinks. Inevitably, the nodes will be the first ones which runs out of power. Algorithms which allow "routing around" failed nodes will increase the load even more on the remaining active nodes close to the sink.

Jing Ai [26], proposed cluster based energy balancing scheme is intended to ameliorate the above energy unbalancing phenomena. He exploit the observation that in a heterogeneous sensor network there are nodes which are more powerful in terms of energy reserve and wireless communication ability. He transformed the flat communication infrastructure into a hierarchical one

where “strong” nodes act as cluster heads to gather information within the clusters and then communicate with the sink directly via single-hop link.

Vivek Mhatre [27], presented a cost based clustered sensor networks. He focused on the case where the base station is remotely located and the sensor nodes are not mobile.

Rajagopal Kannan [28] considered the problem of inter-cluster routing between cluster heads via intermediate sensor nodes in a hierarchical sensor network. He took both path length and path energy cost as metrics.

CHAPTER 3

LINE OF ATTACK

Topology control aims to maintain a topology that optimizes network performance while minimizing energy consumption. This hybrid topology control achieves both scalability and strong connectivity using hybrid *clustering* technique while minimizing energy consumption. Also the thesis examines the performance as well as energy consumption issues of a wireless sensor network providing periodic data from a sensing field to a remote receiver. It distinguishes between two types of sensor organizations, one with a single layer of identical sensors (homogeneous) and one with an additional overlay of fewer but more powerful sensors (heterogeneous). The energy consumption and estimated lifetime based on a clustering mechanism with varying parameters related to the sensing field, e.g., size and distance are also considered. Quantification of the optimal number of clusters based on the proposed model and shows how to allocate the energy between different layers.

Topology control for ad hoc networks aims to maintain a specified topology by controlling which links should be included in the network to achieve a set of network-wide or session-specific objectives such as reducing interference or probability of detection, reduction of energy consumption, increase of the effective network capacity and reducing end to end delay. The primary method of accomplishing topology control is by adjusting the transmission powers of the network nodes. This approach has the following steps:

- Clusters formation and to choose cluster heads in free space wireless model.
- Allocate energy between head and leaf nodes.
- Active cluster head collects data from leaf nodes and forward it to the remote receiver/collector. (one round)

- After certain rounds, active nodes become inactive and one of the inactive node become active node and continues to forward the data packets to the collector.

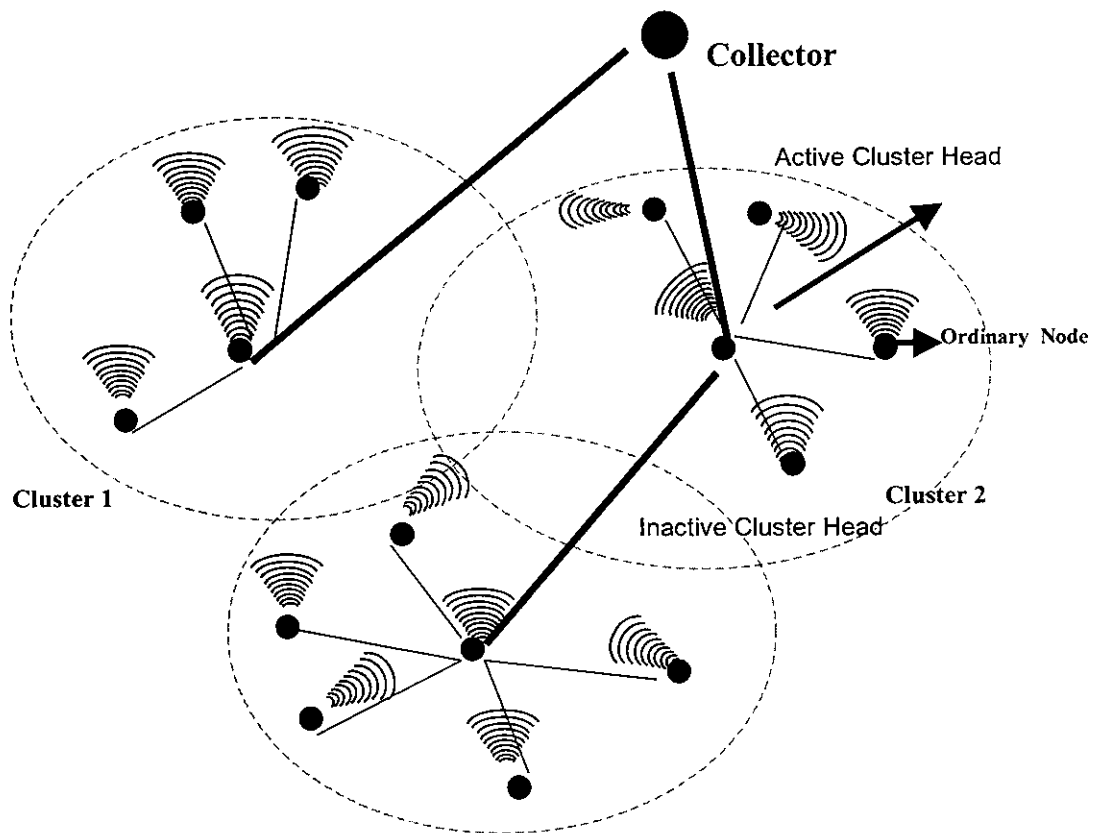


Fig 4. Proposed Sensor Network Architecture

To perform all the above, following are the assumptions:

- Collector/Receiver location is fixed.
- All nodes know the location of the collector.
- The sensing field are square and the nodes are deployed randomly.
- When deployed, head nodes have high power and leaf nodes have low power with the threshold value.
- Sensed data is collected in a periodic manner.

- Each node has a constant amount of raw data to send, including the overlay sensors (cluster head), if active. (b_s bits to send)
- Heads do not perform compression of the collected data nor data fusion.
- Nodes below a certain threshold (preset) are assumed to be dead.

CHAPTER 4

DETAILS OF THE METHODOLOGY EMPLOYED

4.1 CLUSTERS FORMATION AND CLUSTER HEADS SELECTION

In the clustering approach, sensors form clusters dynamically with sensors that are geographically proximate to each other. One of the sensors in the cluster will be elected as cluster head and will be responsible for relaying data from each sensor in the cluster to the remote receiver/collector. This approach localizes traffic and can potentially be more scalable. In addition, the cluster heads naturally become points where data fusion and data compression can occur considering the potential correlation among data from neighboring sensors. Since the cluster heads will inevitably consume more energy and thus die sooner than other sensors, methods of dynamically changing cluster heads are preferred so that the use of energy can be spread as evenly as possible among all sensors. This research focuses on the clustering approach and examines the use of a heterogeneous structure where most of the sensors carry low power, and some nodes carry high power than the other nodes. In this case, the field is first deployed with a number of overlay sensors, presumably more powerful but fewer in number and then deployed with normal low power sensors.

The key features of this approach are:

- Localized coordination and control for cluster set-up and operation.
- Randomized rotation of the cluster “base stations” or “cluster heads” and the corresponding clusters.
- Local compression to reduce global communication.

The use of clusters for transmitting data to the base station leverages the advantages of small transmit distances for most nodes, requiring only a few nodes to transmit far distances to the base station. This achieves a large reduction in the energy dissipation, as computation is much cheaper than communication.

4.2 NETWORK MODEL AND ASSUMPTIONS

Consider a square-sensing field with each side measuring L meters. The coordinates of the field are as shown in Figure 7, where crosses represent overlay sensors, and circles represent normal sensors.

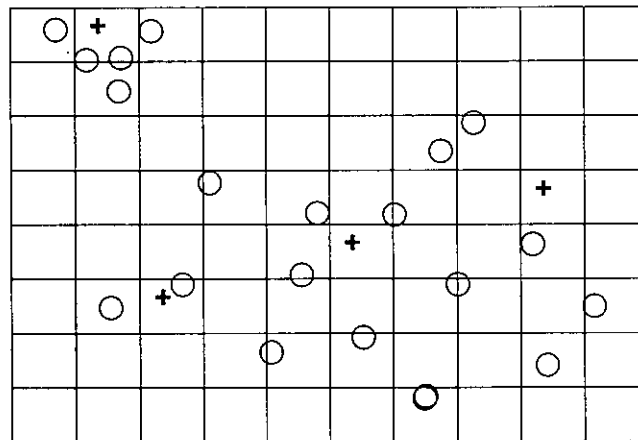


Fig 7. Heterogeneous Network with random deployment sensors

All data collected by the sensors is to be sent to a receiver/collector located outside the sensing field. Here the terms receiver and collector uses interchangeably in subsequent discussions. The collector is located at $(0, -D)$, and is thus D meters away from the sensing field. This location is assumed to be fixed. Also it is assumed that all sensors are aware of the location of the receiver via some type of pre-configuration or self-configuration. There is a total n normal sensors in the field. They are assumed to be uniformly distributed within the field. In addition to that, there $R \cdot q$, (where $R, q > 1$) overlay sensors in

the field, also randomly deployed. On average only q overlay sensors are active at any given time, i.e., on average there are q clusters. These overlay sensors will take turns (in a way described below) being cluster heads. The reason for such redundancy is that a very uneven topology due to randomness in deploying these overlay sensors could lead to rapid energy depletion of a particular overlay sensor and in turn, this can have an adverse effect on the lifetime of the network. If more overlay sensors are deployed than needed and then randomly choosing a subset to active periodically.

4.3 ENERGY ALLOCATION

Here it is assumed that some form of MAC is used within each cluster. Some form of MAC is also used between multiple cluster heads and the remote collector, but the communication between cluster heads and the remote collector takes place in a different channel than that between normal sensors.

The following energy model can be adopted:

$$\text{Energy spent in transmission} = e_d b d^a + e_t b; \quad (1)$$

$$\text{Energy spent in reception} = e_r b; \quad (2)$$

$$\text{Energy spent in sensing} = e_s b; \quad (3)$$

where,

- e_d is the energy dissipated per bit per m^2 and is chosen to be 100×10^{-12} joules.
- e_t is the energy spent by transmission circuitry per bit per m^2 and is chosen to be 50×10^{-9} joules.
- e_r is the energy spent by reception circuitry per bit per m^2 and is chosen to be 50×10^{-9} joules.

- e_s is the energy spent by sensing circuitry per bit per m^2 and is chosen to be 50×10^{-9} joules.
- b is number of bits to transmit or receive
- d is the distance from transmitter to receiver and α (alpha) is a constant ≥ 2 which depends on the attenuation the signal will suffer in that environment.

In this analysis, the common values of α (alpha) = 2 and α (alpha) = 4.

4.4 COLLECTION OF DATA AND TRANSMISSION AND CHANGING ROLLS

It is assumed that sensed data is collected in a periodic manner, and each such period is defined as a *round*. This round consists of the sensing of the data and the transmission of one packet containing the data sensed to the cluster head. The round also includes the relay of that packet and the packet of each sensor in the cluster to the collector. Furthermore, it is assumed that each sensor has a constant amount of raw data to send, including the overlay sensors if active. Thus every sensor, in every round, has b_s bits to send. A round starts with each overlay sensor dynamically deciding whether it will be a cluster head in the current round. If so it broadcasts its presence to the normal sensors and starts receiving data from the sensors that have decided to become part of its cluster. Normal sensors decide to which cluster they wish to belong based on the strength of the signal from the broadcast. It is assumed that the stronger the signal, the closer the head is and therefore the head with the strongest signal is chosen. If an overlay sensor decides not to be a cluster head for the current head for the current round, it goes to sleep for the duration of the round. Once the data from all the sensors within the cluster is gathered, it is relayed to the collector. This marks the end of a round the beginning of the next round. By specifying that an overlay sensor is active once and only once every R rounds, it is ensure that on average there are q clusters in the network.

CHAPTER 5

RESULTS OBTAINED

The following are the results obtained for the proposed topology and energy conservation scheme under different scenarios. The plots are obtained assuming the perfect scheduling, data packets of 1024 bits and control packets of 128 bits. For a different number of clusters the amount of energy carried by a normal sensor and an overlay sensor is determined with the help of formulae (1), (2) and (3). In these experiments the total number of sensors (both types) is fixed at 100. The expected number of active overlay sensors will be q , and $100-q$ is the number of normal sensors.

Let q vary from 1 to 100. In other words, the average number of active sensors (normal plus overlay) remains fixed, while the average number of cluster increases. Since an overlay sensor also has normal sensing capability, by doing so it is fixed to the amount of sensing data from one scenario to another. In addition, under such a setup, $q=100$ would correspond to direct transmission. Figures 8.1 through 8.3 show the expected number of rounds the network can last as a function of number of clusters. In Figure 8.1 there is a clear knee corresponding to q between 4 and 10, while the maximum is reached when $q = 100$, which represents direct transmission. This is because we have assumed perfect MAC scheduling, and that direct transmission does not involve cluster formation overhead that is incurred periodically otherwise. Intuitively it would like to maximize the lifetime of the network while minimizing the number of clusters in order to be scalable. Therefore the optimal average number of clusters is determined as the knee in the curve. The exact number for q varies slightly depending on other parameters such as L , D and α .

The change of α , or a scale-down in L and D does not change the overall shape of the curves. An increase in the dimension of the network also does not change the overall shape (Figure 8.2). Setting $L = 500$ and $D = 1000$ reduces

the number of rounds the network can last, but not the location of the knee. Note that in this last case having a larger number of clusters no longer means a longer lifetime. In a network of this size the distance from sensors to a cluster head increases, i.e., the network becomes more sparse. However the transmission range needed for broadcast during cluster formation by a cluster head increases even more. Therefore beyond a certain range the increased numbers of clusters result in higher energy consumption.

Reversing the ration between L and D results in a very different shape of this curve (Figure 8.3). In this case it is not so obvious what the best choice for q would be.

In the cases shown in Figures 8.1 and 8.2 the energy spent in propagation is the dominant factor in the total amount of energy consumed, and the amount of energy spent in transmission and reception circuitry is relatively minimal. However, in Figure 8.3 the distance between the receiver and field is small enough to allow the energy spent in the circuitry to become significant. In this case direct transmission becomes a valid choice, assuming that (near) perfect scheduling is possible. Therefore for q to be between 4 and 10 is only recommended for scenarios where the sensing field and the receiver are far away comparing to the size of the sensing field. When the field is larger comparing to the distance between the field and the receiver, direct transmission seems to a good idea and choose q as high as the receiver can handle. These results are obtained using the xgraph tool of NS2 simulator.

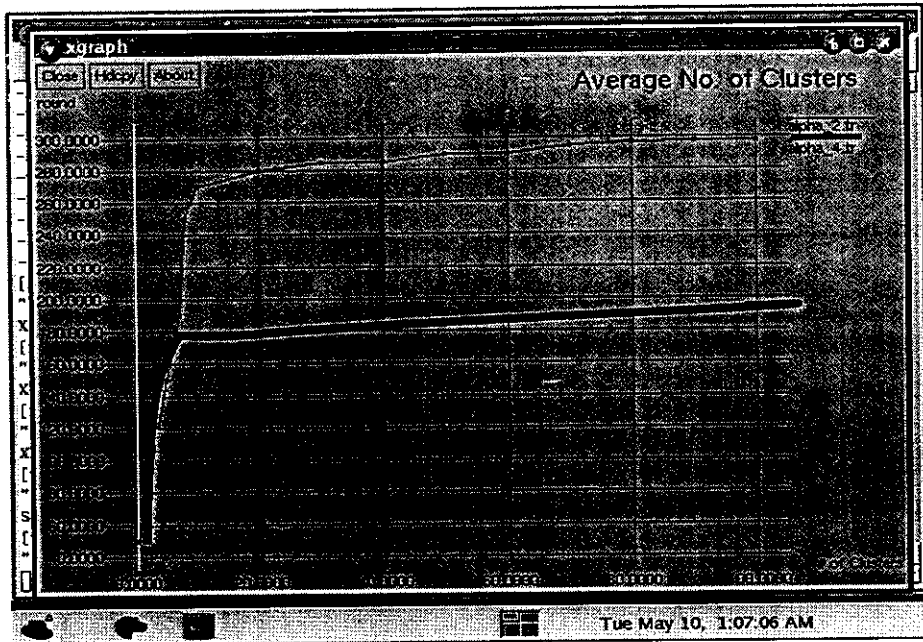


Figure 8.1. Optimal Clustering for $\text{Alpha}=2$, $L=50$, $D=100$ and
 $\text{alpha}=4$, $L=5$, $D=10$

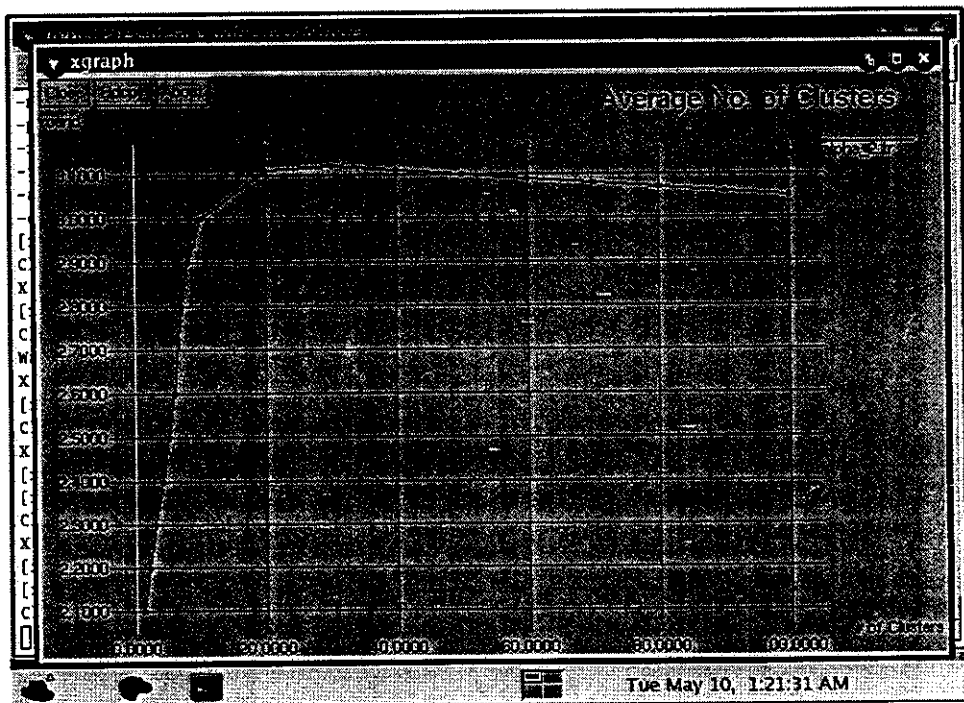


Fig 8.2. Optimal Clustering for $\text{Alpha}=2$, $L=500$, $D=1000$

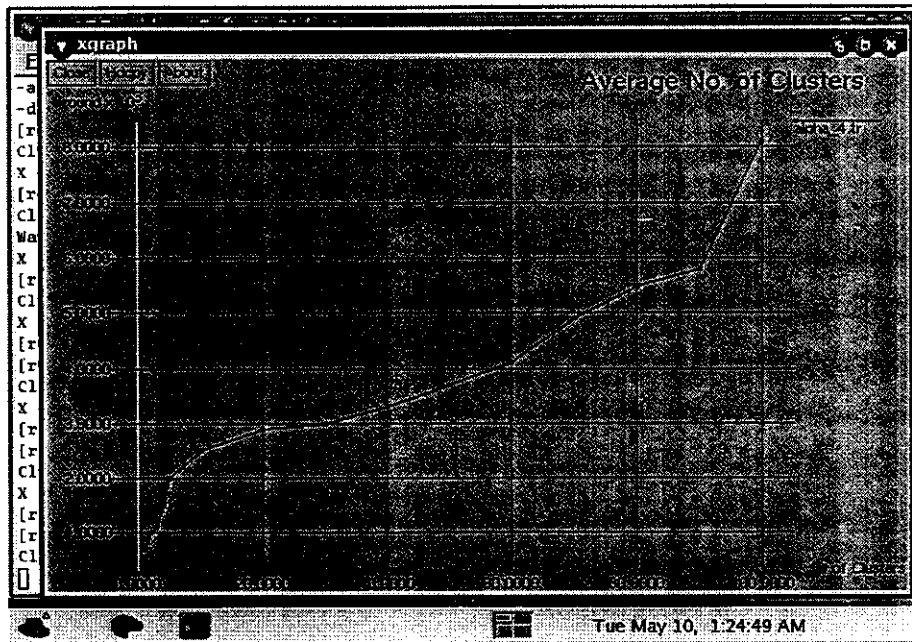


Fig 8.3. Optimal Clustering for Alpha=2, L=5 D=1

CHAPTER 6

CONCLUSION AND FUTURE OUTLOOK

This thesis work examines the performance as well as energy consumption issues of a wireless sensor network providing periodic data from a sensing field to a remote receiver. It distinguishes between two types of sensor organizations, one with a single layer of identical sensors (homogeneous) and one with an additional overlay of fewer but more powerful sensors (heterogeneous). The energy consumption and estimated lifetime based on a clustering mechanism with varying parameters related to the sensing field, e.g., size and distance were also investigated. Quantification of the optimal number of clusters based on the proposed model was done and showed how to allocate the energy between different nodes. Also shows how the dynamic clusters were formed, presenting a way of determining the optimal number of clusters for a given set of parameters.

Future Outlook:

Only a limited number of aspects of sensor networks have been considered here. Future work would explore similar issues in a query-driven and event-driven type of sensor networks. The possibility of several collectors located in different places should also be considered. Another important issue to be explored is a heterogeneous network model where the difference between the sensors is not only the difference in available energy, but also in their processing capabilities, and thus the consideration of energy consumption in data processing. In cases where delay and the resolution of the data are just as important, these performance measures should be considered jointly with energy efficiency.

APPENDICES

Appendix – 1: Sample Source Code

```

$ns_ node-config -adhocRouting $opt(adhocRouting) \
    -llType $opt(ll) \
    -macType $opt(mac) \
    -ifqType $opt(ifq) \
    -ifqLen $opt(ifqlen) \
    -antType $opt(ant) \
    -propType $opt(prop) \
    -phyType $opt(netif) \
    -channelType $opt(chan) \
    -topoInstance $wtopo \
    -energyModel $opt(EnergyModel) \
    -initialEnergy 260 \
    -rxPower 0.3 \
    -txPower 0.6 \
set opt(initialenergy)      5
set opt(energymodel)       EnergyModel
set opt(chan)              Channel/WirelessChannel
set opt(p_rx)              0.281
set opt(p_tx)              0.281
set opt(prop)              Propagation/TwoRayGround
set opt(netif)             Phy/WirelessPhy
set opt(mac)               Mac/802_11
set opt(ifq)               Queue/DropTail/PriQueue
set opt(ll)                LL
set opt(ant)               Antenna/OmniAntenna
set opt(x)                 500           ;# X dimension of the topography
set opt(y)                 500           ;# Y dimension of the topography
set opt(ifqlen)            64           ;# max packet in ifq

```

```

set opt(seed)      0.0
#set opt(nam)      alpha2.nam  ;# nam trace file
set opt(rp)        AODV
set opt(nn)        100          ;# how many nodes are simulated
set opt(tr)        ""          ;# trace file
set opt(cp)        ""
set opt(sc)        ""
set opt(stop)      900

# Other default settings
LL set mindelay_   50us
LL set delay_      25us
LL set bandwidth_  0          ;# not used
Agent/Null set sport_  0
Agent/Null set dport_  0
Agent/CBR set sport_  0
Agent/CBR set dport_  0
Agent/CBR set packetSize 512
Agent/TCPSink set sport_ 0
Agent/TCPSink set dport_ 0
Agent/TCP set sport_  0
Agent/TCP set dport_  0
Agent/TCP set packetSize_ 1460
Queue/DropTail/PriQueue set Prefer_Routing_Protocols 1
Mac/802_11 set basicRate_ 2Mb # set this to 0 if want to use bandwidth_ for
\n\ Mac/802_11 set dataRate_ 2Mb ;# both control and data pkts\n\
# unity gain, omni-directional antennas
# set up the antennas to be centered in the node and 1.5 meters above it
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
```

```
# Initialize the SharedMedia interface with parameters to make
# it work like the 914MHz Lucent WaveLAN DSSS radio interface
```

```
Phy/WirelessPhy set CPTresh_ 10.0
Phy/WirelessPhy set CSTresh_ 1.559e-11
Phy/WirelessPhy set RXThresh_ 3.652e-10
Phy/WirelessPhy set Rb_ 2*1e6
#Phy/WirelessPhy set Pt_ 0.2818
Phy/WirelessPhy set freq_ 914e+6
Phy/WirelessPhy set L_ 1.0
```

```
proc usage { argv0 } {
    puts "Usage: $argv0"
    puts "\tmandatory arguments:"
    puts "\t\t\t[-x MAXX\] \[-y MAXY\]"
    puts "\toptional arguments:"
    puts "\t\t\t[-cp conn pattern\] \[-sc scenario\] \[-nn nodes\]"
    puts "\t\t\t[-seed seed\] \[-stop sec\] \[-tr tracefile\]\n"
}
```

```
proc getopt {argc argv} {
    global opt
    lappend optlist cp nn seed sc stop tr x y
    for {set i 0} {$i < $argc} {incr i} {
        set arg [lindex $argv $i]
        if {[string range $arg 0 0] != "-"} continue
        set name [string range $arg 1 end]
        set opt($name) [lindex $argv [expr $i+1]]
    }
}
```



```
}
```

```
#####  
# Main Program  
#####  
getopt $argc $argv  
if { $opt(x) == 0 || $opt(y) == 0 } {  
    usage $argv0  
    exit 1  
}  
  
#  
# Initialize Global Variables  
#  
  
# create simulator instance  
set ns_ [new Simulator]  
  
# set wireless channel, radio-model and topography objects  
set wchan [new $opt(chan)]  
set wprop [new $opt(prop)]  
set wtopo [new Topography]  
  
# create trace object for ns and nam  
set tracefd [open $opt(tr) w]  
  
#set namtrace [open $opt(nam) w]  
# use new trace file format  
  
$ns_ use-newtrace
```

```
$ns_ trace-all $tracefd
$ns_ namtrace-all-wireless $namtrace $opt(x) $opt(y)

# define topology
$wtopo load_flatgrid $opt(x) $opt(y)

$wprop topography $wtopo

#
# Create God
#
set god_ [create-god $opt(nn)]

#
# define how node should be created
#

#global node setting
set chan_1_ [new $opt(chan)]
$ns_ node-config -adhocRouting $opt(rp) \
    -llType $opt(ll) \
    -macType $opt(mac) \
    -ifqType $opt(ifq) \
    -ifqLen $opt(ifqlen) \
    -antType $opt(ant) \
    -propType $opt(prop) \
    -phyType $opt(netif) \
    -channel $chan_1_ \
    -topoInstance $wtopo \
    -agentTrace ON \
```

```

-routerTrace ON \
-macTrace OFF
-energyModel $opt(energymodel) \
-rxPower $opt(p_rx) \
-txPower $opt(p_tx) \
-initialEnergy $opt(initialenergy) \

# Create the specified number of nodes [$opt(nn)] and "attach" them
# to the channel.

for {set i 0} {$i < $opt(nn)} {incr i} {
    set node_($i) [$ns_node]
    $node_($i) random-motion 0           ;# disable random motion
    $node_($i) topography $wtopo
}

#
# Define node movement model
#
puts "Loading connection pattern... $opt(cp)"
source $opt(cp)

#
# Define traffic model
#
puts "Loading scenario file... $opt(sc)"
source $opt(sc)

# Define node initial position in nam

for {set i 0} {$i < $opt(nn)} {incr i} {

```

20 defines the node size in nam, must adjust it according to your
scenario. The function must be called after mobility model is defined

```
$ns_ initial_node_pos $node_($i) 30
}
```

```
#
```

```
# Tell nodes when the simulation ends
```

```
#
```

```
for {set i 0} {$i < $opt(nn)} {incr i} {
```

```
    $ns_ at $opt(stop).000000001 "$node_($i) reset";
```

```
}
```

```
# tell nam the simulation stop time
```

```
$ns_ at $opt(stop) "$ns_ nam-end-wireless $opt(stop)"
```

```
$ns_ at $opt(stop).000000001 "puts \"running conta.sh...\" ; exec conta.sh
```

```
$opt(tr)" $ns_ at $opt(stop).000000001 "puts \"NS EXITING...\" ; $ns_ halt"
```

```
puts $tracefd "tracegraph"
```

```
puts $tracefd "M 0.0 nn $opt(nn) x $opt(x) y $opt(y) rp $opt(rp)"
```

```
puts $tracefd "M 0.0 sc $opt(sc) cp $opt(cp) seed $opt(seed)"
```

```
puts $tracefd "M 0.0 prop $opt(prop) ant $opt(ant)"
```

```
puts "Starting Simulation..."
```

```
$ns_ run
```

//Cluster Power Control

```

extern "C" {
#include <stdarg.h>
#include <float.h>
};

#include "clusterpow.h"
#include "priqueue.h"
#include <random.h>
#include <cmu-trace.h>
#include <address.h>
#include <mobilenode.h>

// jitter for all broadcast packets
#define CLUSTERPOW_BROADCAST_JITTER 0.01

// jitter used for events that should be effectively
#define CLUSTERPOW_ALMOST_NOW 0.1

// instantaneous but are jittered to prevent synchronization
// default TTL
#define IP_DEF_TTL 32

// Returns a random number between 0 and max
static inline double jitter(double max, int be_random_)
{
    return (be_random_ ? Random::uniform(max) : 0);
}

int CLUSTERPOW_Agent::diff_subnet(int dst)
{
    {
        char *dstnet = Address::instance().get_subnetaddr(dst);
        if (subnet_ != NULL) {
            if (dstnet != NULL) {
                if (strcmp(dstnet, subnet_) != 0) {
                    delete[]dstnet;
                    return 1;
                }
            }
            delete[]dstnet;
        }
    }
    //assert(dstnet == NULL);
    return 0;
}

```

```

static void mac_callback(Packet * p, void *arg)
{
    //printf("CLUSTERPOW: mac_callback\n");
    Packet::free(p);
}

void CLUSTERPOW_Agent::forwardPacket(Packet * p)
{
    hdr_ip *iph = HDR_IP(p);
    //Scheduler & s = Scheduler::instance ();
    hdr_cmn *hdrc = HDR_CMN(p);
    int dst;
    ftable_ent *pfte;

    // set direction of pkt to -1 , i.e downward
    hdrc->direction() = hdr_cmn::DOWN;

    // if the destination is outside mobilenode's domain
    // forward it to base_stn node
    // Note: pkt is not buffered if route to base_stn is unknown

    dst = Address::instance().get_nodeaddr(iph->daddr());

#ifdef UNDEFINED
    if (diff_subnet(iph->daddr())) {
        cout << "In diff_subnet() " << myaddr_ << endl ;
        pfte = ftable_->GetEntry(dst);
        if (pfte && pfte->metric != BIG)
            goto send; }

        else {
            //drop pkt with warning
            fprintf(stderr,
                "warning: Route to base_stn not known: dropping pkt\n");
            Packet::free(p);
            return;
        }
    }
#endif

    pfte = ftable_->GetEntry(dst);

    if (pfte && pfte->metric != BIG) {

```

```

        //printf("(%d)-have route for dst\n",myaddr_);
        goto send;
    } else {
        //printf("(%d)-no route, queue pkt\n",myaddr_);
        drop(p, DROP_RTR_QFULL);
        return;
    }

send:
    hrc->addr_type_ = NS_AF_INET;
    hrc->xmit_failure_ = mac_callback;
    hrc->xmit_failure_data_ = this;
    if (pfte->metric > 1)
        hrc->next_hop_ = pfte->hop;
    else
        hrc->next_hop_ = dst;

    // set the txpower too
    hrc->txpower_ = pfte->txpower;

//cout << "Clusterpow forward(): ptype " << hrc->ptype() << "\t" << iph->saddr()
<< " -> " << dst << endl;

    assert(!HDR_CMN(p)->xmit_failure_ ||
        HDR_CMN(p)->xmit_failure_ == mac_callback);
    target_->recv(p, (Handler *) 0);
    return;
}

void CLUSTERPOW_Agent::sendOutBCastPkt(Packet * p)
{
    Scheduler & s = Scheduler::instance();
    // send out bcast pkt with jitter to avoid sync
    s.schedule(target_, p,
        jitter(CLUSTERPOW_BROADCAST_JITTER, be_random_));
}

void CLUSTERPOW_Agent::recv(Packet * p, Handler *)
{
    hdr_ip *iph = HDR_IP(p);
    hdr_cmh *cmh = HDR_CMN(p);
    int src = Address::instance().get_nodeaddr(iph->saddr());
    int dst = cmh->next_hop();
}

```

```

//cout << "Clusterpow rcv(): ptype " << cmh->ptype() << "\t" << iph->saddr() << "
-> " << dst << endl;

/*
 * Must be a packet I'm originating...
 */
if (src == myaddr_ && cmh->num_forwards() == 0) {
    /*
     * Add the IP Header
     */
    cmh->size() += IP_HDR_LEN;
    iph->ttl_ = IP_DEF_TTL;
}
/*
 * I received a packet that I sent. Probably
 * a routing loop.
 */
else if (src == myaddr_) {
    drop(p, DROP_RTR_ROUTE_LOOP);
    return;
}
/*
 * Packet I'm forwarding...
 */
else {
    /*
     * Check the TTL. If it is zero, then discard.
     */
    if (--iph->ttl_ == 0) {
        drop(p, DROP_RTR_TTL);
        return;
    }
}

if ((src != myaddr_) && (iph->dport() == ROUTER_PORT)) {
    // I should never receive updates
    // my peers never send packets
    //processUpdate(p);
} else if ((u_int32_t) dst == IP_BROADCAST &&
    (iph->dport() != ROUTER_PORT)) {
    if (src == myaddr_) {
        // handle brdcast pkt
//printf("%d broadcasting on port %d, and I am %d\n", src, iph->dport(),
myaddr_);
        sendOutBCastPkt(p);
    }
}

```



```

    } else {
        // // hand it over to the port-demux
        //printf("CLUSTERPOW at node %d: port_dmux_\n", myaddr_);
        port_dmux_>recv(p, (Handler *) 0);
    }
} else {
    forwardPacket(p);
}
}

```

```

static class CLUSTERPOWClass:public TclClass {
public:
    CLUSTERPOWClass():TclClass("Agent/CLUSTERPOW") {
    } TclObject *create(int, const char *const *) {
        return (new CLUSTERPOW_Agent());
    }
}

```

```
class_clusterpow;
```

```

CLUSTERPOW_Agent::CLUSTERPOW_Agent():Agent(PT_MESSAGE),
ll_queue(0), myaddr_(0), subnet_(0), node_(0),
port_dmux_(0), be_random_(1),
use_mac_(0)
{
    ftable_ = new ForwardingTable();
    bind("use_mac_", &use_mac_);
    bind("be_random_", &be_random_);
    //DEBUG
    address = 0;
}

```

```

void CLUSTERPOW_Agent::startUp()
{
    subnet_ = Address::instance().get_subnetaddr(myaddr_);
    address = Address::instance().print_nodeaddr(myaddr_);
    //printf("CLUSTERPOW agent starting: myaddress: %d ->
%s\n",myaddr_.address);

```

```

    ftable_ent fte;
    bzero(&fte, sizeof(fte));

```

```

    fte.dst = myaddr_;
    fte.hop = myaddr_;

```

```

fte.metric = 0;
    fte.txpower = 0;

fable_>AddEntry(fte);
}

int CLUSTERPOW_Agent::command(int argc, const char *const *argv)
{
    if (argc == 2) {
        if (strcmp(argv[1], "start-clusterpow") == 0) {
            startUp();
            return (TCL_OK);
        } else if (strcmp(argv[1], "dumprtab") == 0) {
            Packet *p2 = allocpkt();
            hdr_ip *iph2 = HDR_IP(p2);
            ftable_ent *pfte;

            printf("\nForwarding Table Dump %d[%d]\n-----\n",
                iph2->saddr(), iph2->sport());
            printf("time\t\t dst\t nhop\t metric\t txpower\n");

            /*
             * Freeing a routing layer packet --> don't need to
             * call drop here.
             */
            Packet::free(p2);

            int count = 0;
            for (fable_>InitLoop(); (pfte = fable_>NextLoop()); {
                if(pfte->metric != BIG) {

                    count++;
                    printf("%f\t %d\t %d\t %d\t %d\n", Scheduler::instance().clock(), pfte->dst, pfte-
                    >hop, pfte->metric, pfte->txpower);
                }
            }
            printf ("##### Total entries: %d \n", count);

            return (TCL_OK);

        } else if (strcasecmp(argv[1], "ll-queue") == 0) {
            if (!(ll_queue = (PriQueue *) TclObject::lookup(argv[2]))) {
                fprintf(stderr, "CLUSTERPOW_Agent: ll-queue lookup of %s failed\n",
                    argv[2]);
            }
        }
    }
}

```

```

        return TCL_ERROR;
    }

    return TCL_OK;
}

} else if (argc == 3) {
    if (strcasecmp(argv[1], "addr") == 0) {
        int temp;
        temp = Address::instance().str2addr(argv[2]);
        myaddr_ = temp;
        return TCL_OK;
    }
    TclObject *obj;
    if ((obj = TclObject::lookup(argv[2])) == 0) {
        fprintf(stderr, "%s: %s lookup of %s failed\n", __FILE__,
            argv[1], argv[2]);
        return TCL_ERROR;
    }
    if (strcasecmp(argv[1], "tracetarget") == 0) {
        tracetarget = (Trace *) obj;
        return TCL_OK;
    } else if (strcasecmp(argv[1], "node") == 0) {
        node_ = (MobileNode *) obj;
        return TCL_OK;
    } else if (strcasecmp(argv[1], "port-dmux") == 0) {
        port_dmux_ = (NsObject *) obj;
        return TCL_OK;
    } else if (strcasecmp(argv[1], "del-route") == 0) {
        nsaddr_t rdst= atoi(argv[2]);
        printf("Clusterpow deleting route\n");
        return TCL_OK;
    }
} else if (argc == 6) {
    if (strcasecmp(argv[1], "add-route") == 0) {
        ftable_ent new_fte, *old_fte ;
        bzero(&new_fte, sizeof(new_fte));
        bzero(&old_fte, sizeof(old_fte));

        new_fte.dst = atoi(argv[2]);
        new_fte.hop = atoi(argv[3]);
        new_fte.metric = atoi(argv[4]);
        new_fte.txpower = atoi(argv[5]);

        old_fte = ftable_->GetEntry(new_fte.dst);
        if (old_fte && old_fte->metric != BIG) {

```

```
        if(old_fte->txpower > new_fte.txpower) {
            ftable_->AddEntry(new_fte);
            //printf("Node %d Replacing route to: %d %d %d %d\n", myaddr_,
new_fte.dst, new_fte.hop, new_fte.metric, new_fte.txpower);
        }
        } else {
            ftable_->AddEntry(new_fte);
            //printf("Node %d Adding new route: %d %d %d %d\n", myaddr_, new_fte.dst,
new_fte.hop, new_fte.metric, new_fte.txpower);
        }

        return TCL_OK;
    }
}

return (Agent::command(argc, argv));
}
```

Appendix – 2: Screen Shots

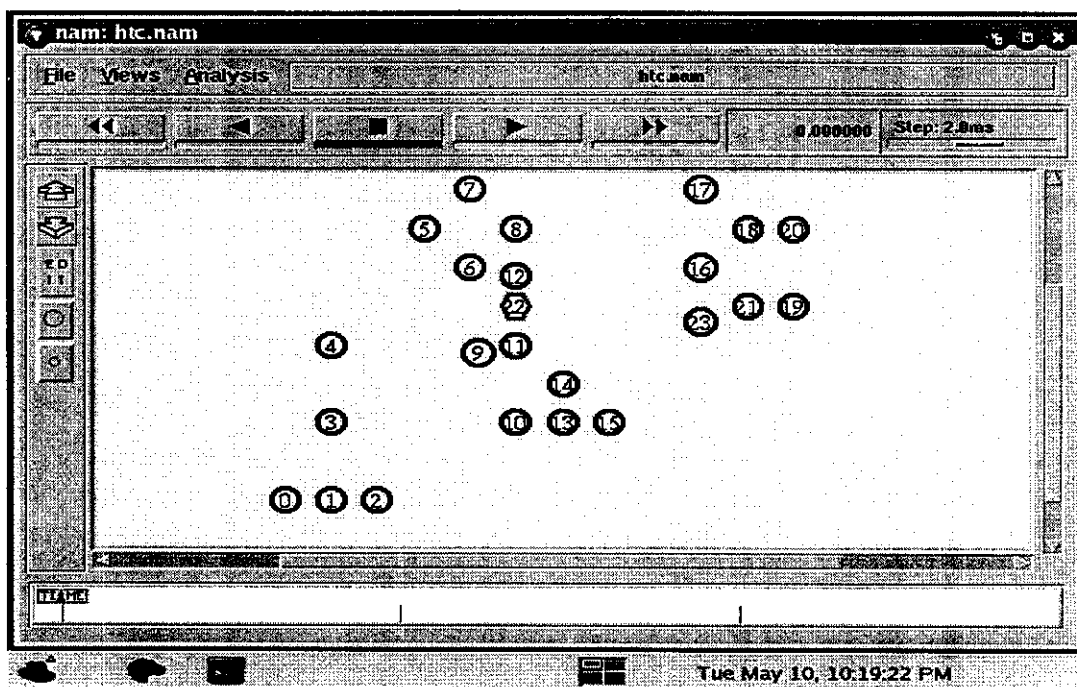


Fig 9. Cluster Formation

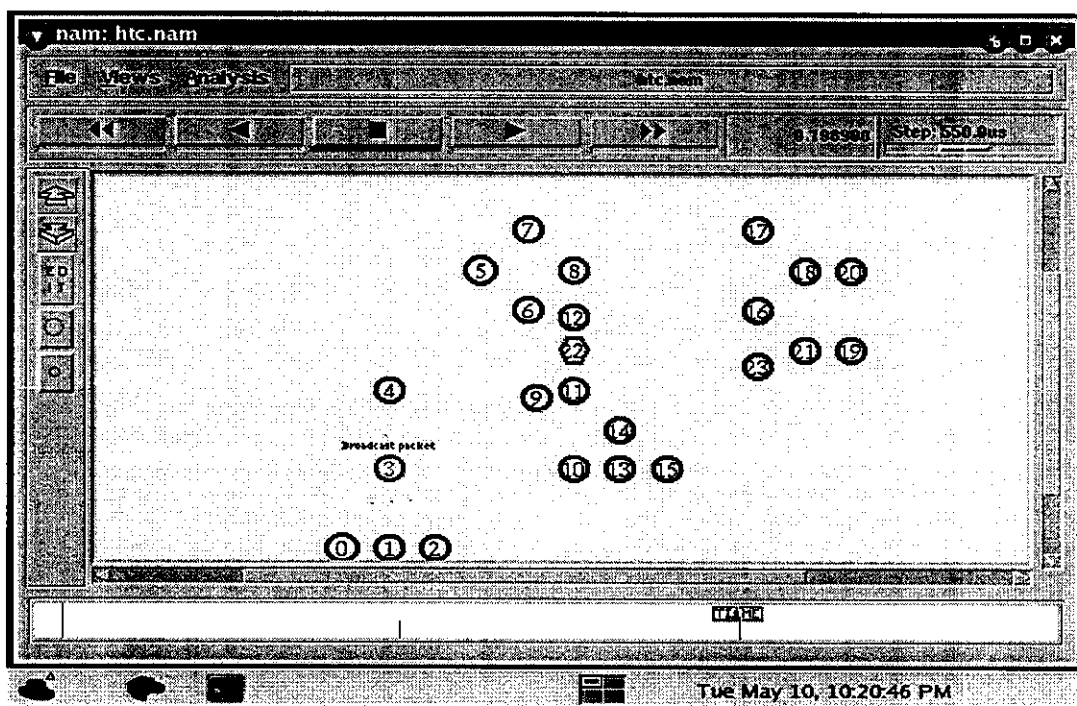


Fig 10. Cluster Heads Election

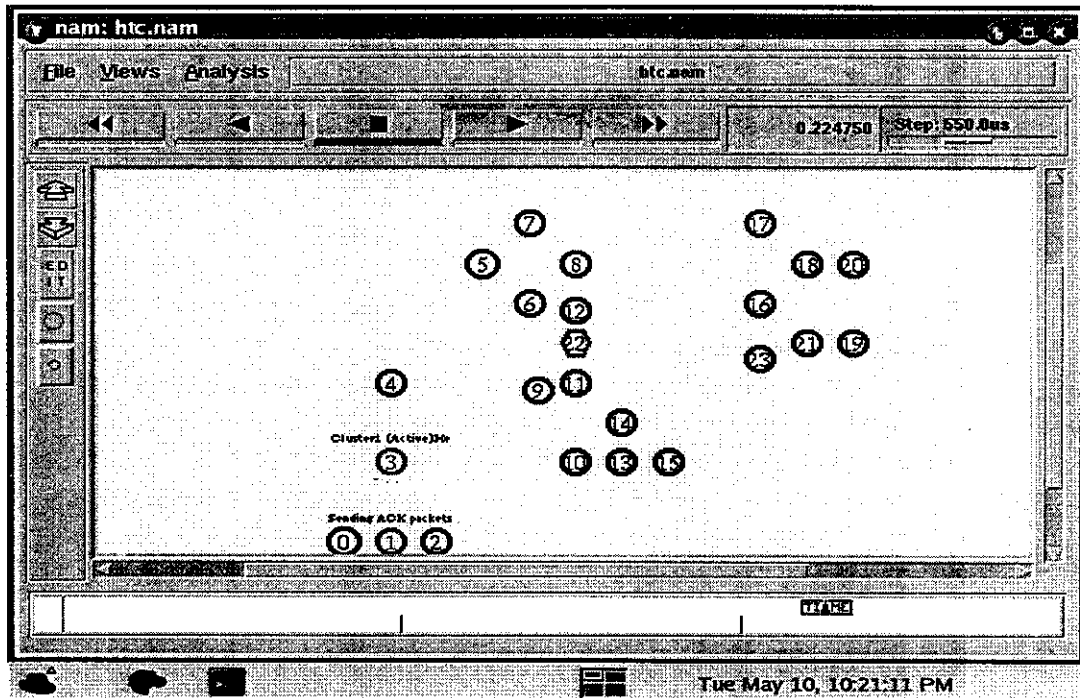


Fig 11. Active Cluster Head Election

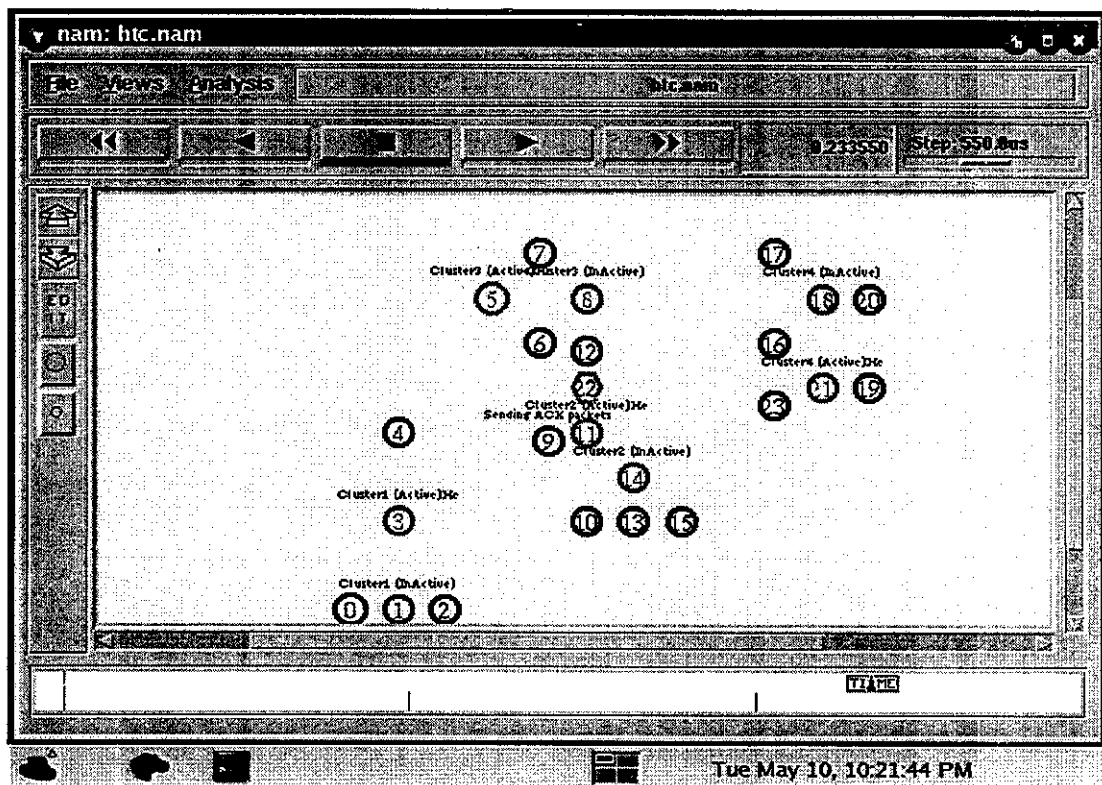


Fig 12. Clusters Formation &
Cluster Heads Election (both active & inactive)

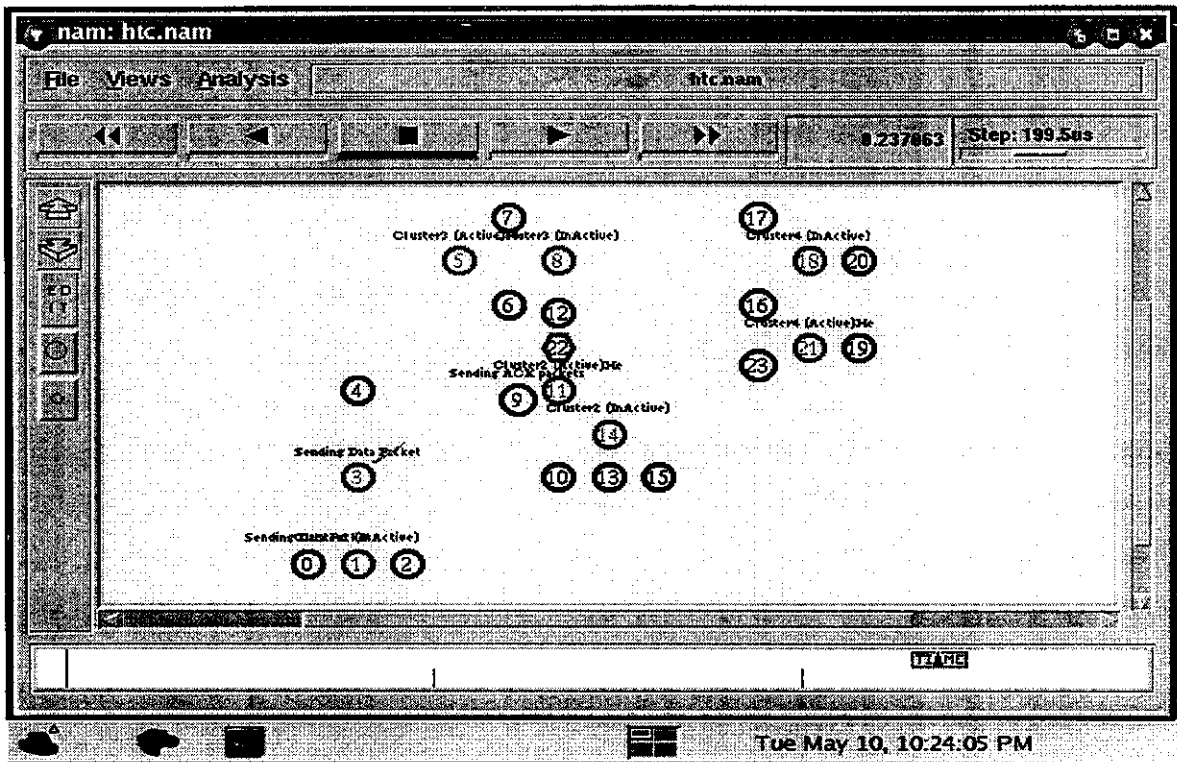


Fig 13. Data Transmission from Head to Collector

REFERENCES

- [1] P. Lettieri, C. Schurgers and M. Srivastava, "Adaptive Link Layer Strategies for Energy Efficient Wireless Networking", *Wireless Networks*, 5(5), October 1999, pp. 339-355.
- [2] Rui Liu, "Topology Control for Ad Hoc Networks", Computer and Information Sciences, University of Delaware, Spring 2004.
- [3] Anmol Sheth, Richard Han, "Adaptive Power Control and Selective Radio Activation For Low-Power Infrastructure-Mode 802.11 LANs", ICDCS 2003.
- [4] Errol L. Lloyd, Rui Liu, Madhav V. Marathe, Ram Ramanathan, and S. S. Ravi, "Algorithmic Aspects of Topology Control Problems for Ad hoc Networks", in the IEEE MOBIHOC, EPFL Lausanne, Switzerland, June 2002.
- [5] Ram Ramanathan and Regina Rosales-Hain, "Topology Control of Multihop Wireless Networks using Transmit Power Adjustment", in Infocom 2000, Tel-Aviv, Israel, 2000, pp. 404.413.
- [6] Li Li, Joseph Y. Halpern, Paramvir Bahl, Yi-Min Wang, and Roger Wattenhofer, "Analysis of a Cone-Based Distributed Topology Control Algorithms for Wireless Multi-hop Networks", in ACM Symposium on Principle of Distributed Computing (PODC), Newport, Rhode Island, August 2001.
- [7] Roger Wattenhofer, Li Li, Paramvir Bahl, and Yi-Min Wang, "Distributed Topology Control for Power Efficient Operation in Multihop Wireless Ad Hoc Networks", in Infocom 2001, Anchorage, Alaska, April 2001.

- [8] Volkan Rodoplu and Teresa H. Meng, "Minimum energy mobile wireless networks", *IEEE Transactions Selected Areas in Communications*, vol. 17, no. 8, August 1999.
- [9] Zhuochuan Huang, Chien-Chung Shen, Chavalit Srisathapornphat, and Chaiporn Jaikaeo, "Topology Control for Ad hoc Networks with Directional Antennas", in *The 11th International Conference on computer Communications and Networks (ICCCN02)*, Miami, Florida, October 14-16 2002.
- [10] Lefteris M. Kirousis, Evangelos Kranakis, Danny Krizanc, and Andrzej Pelc, "Power Consumption in Packet Radio Networks," in *14th Annual Symposium on Theoretical Aspects of Computer Science, STACS 97*, Lubeck, Germany, February 1997, vol. 1200, pp. 363. 374, Springer-Verlag.
- [11] Errol L. Lloyd, Rui Liu, Madhav V. Marathe and Ram Ramanathan, "Cluster Based Topology Control for Ad hoc Networks", in *the IEEE Mobile Computing*, March 2003.
- [12] Ram Ramanathan and Regina Rosales-Hain, "Topology Control of Multihop Wireless Networks using Transmit Power Adjustment", in *Infocom 2000*, Tel-Aviv, Israel, 2000, pp. 404.413.
- [13] Li Li, Joseph Y. Halpern, Paramvir Bahl, Yi-Min Wang, and Roger Wattenhofer, "Analysis of a Cone-Based Distributed Topology Control Algorithms for Wireless Multi-hop Networks", in *ACM Symposium on Principle of Distributed Computing (PODC)*, Newport, Rhode Island, August 2001.
- [14] Roger Wattenhofer, Li Li, Paramvir Bahl, and Yi-Min Wang, "Distributed Topology Control for Power Efficient Operation in Multihop Wireless Ad Hoc Networks", in *Infocom 2001*, Anchorage, Alaska, April 2001.

[15] Volkan Rodoplu and Teresa H. Meng, "Minimum energy mobile wireless networks", *IEEE Transactions Selected Areas in Communications*, vol. 17, no. 8, August 1999.

[16] Swetha Narayanaswamy, Vikas Kawadia, Ramavarapu S.Sreenivas, and P. R. Kumar, "Power Control in Ad-Hoc Networks: Theory, Architecture, Algorithm and Implementation of the COMPOW Protocol", in *Proceedings of European Wireless*, Florence, Italy, Feb. 2002, pp. 156.62.

[17] Mainak Chatterjee, Sajal K. Das, and Damla Turgut, "WCA: A Weighted Clustering Algorithm for Mobile Ad hoc Networks", *accepted for publication in the Journal of Cluster Computing in the special issue on Mobile Ad hoc Networking*, 2001.

[18] Chaiporn Jaikao and Chien-Chung Shen, "Adaptive Backbone-based Multicast for Ad hoc Networks", in *IEEE International Conference on Communications (ICC 2002)*, New York City, NY, April 2002.

[19] Chi-Fu Huang, Yu-Chee Tseng, Shih-Lin Wu, and Jang-Ping Sheu, "Distributed topology control algorithm for multihop wireless networks", in *Neural Networks, 2002. IJCNN '02. Proceedings of the 2002 International Joint Conference on , Vol.1, Pages: 355-360*, 2002.

[20] Kenneth J. Supowit, "The relative hood graph with application to minimum spanning trees", in *Journal of the ACM*, vol. 30, no. 3, pp 428-448, 1983.

[21] Ning Li, Jennifer C. Hou, and Lui Sha, "Design and Analysis of an MSTBased Topology Control Algorithm", in *INFOCOM 2003*.

[22] Hu L, "Topology control for multihop packet radio networks", in *Communications, IEEE Transactions on , Vol.41, Iss.10, Pages: 1474- 1481*, 1993.

[23] Jilei Liu and Baochun Li, "Mobilegrid: capacity-aware topology control in mobile ad hoc networks", in *Computer Communications and Networks, 2002. Proceedings. Eleventh International Conference on, Vol., Pages: 570- 574, 2002.*

[24] Chaiporn Jaikaeo, Chien-Chung Shen, "Adaptive Backbone-Based Multicast for Ad Hoc Networks", Proc. IEEE Int'l Conf. Comm. (ICC 2002), 28 Apr – 2 May, 2002.

[25] Gaurav Srivastava, Paul Boustead, Joe F.Chicharo, "A comparison of Topology Control Algorithms for Ad hoc networks", in *INFOCOM 2003.*

[26] Jing Ai, Damla Turgut and Ladislau Boloni, "A Cluster-based Energy Balancing Scheme in Heterogeneous Wireless Sensor Networks", *INFOCOMM 2002*, pp 1583 – 1596, New York, June 2002.

[27] Vivek Mhatre, Catherine Rosenberg, "Homogeneous Vs Hetrogeneous Clustered Sensor Networks", In the proceedings of SIGCOMM 2001 Conference on Communications Architectures, Protocols and Applications, August 2001.

[28] Rajagopal Kannan, Lydia Ray, Ram Kalidindi, S.S.Iyengar, "Max-Min Length-Energy-Constrained Routing in Wireless Sensor Network", Proc. Of 2nd IEEE Wksp Mobile Comp. Sys. And Applications, Feb 1999.

