ADAPTIVE CLUSTERING AND TRANSMISSION RANGE ADJUSTMENT FOR TOPOLOGY CONTROL IN WIRELESS SENSOR NETWORKS

A PhD Dissertation presented to
the Faculty of the Graduate School
at the University of Missouri-Columbia

In Partial Fulfillment
of the Requirements for the Degree

Doctor of Philosophy

---

by

BOLIAN YIN

Dr. Hongchi Shi, Dissertation Supervisor

DECEMBER 2006

The undersigned, appointed by the dean of the Graduate School, have examined the dissertation entitled

ADAPTIVE CLUSTERING AND TRANSMISSION RANGE ADJUSTMENT FOR TOPOLOGY CONTROL IN WIRELESS SENSOR NETWORKS

presented by Bolian Yin,

a candidate for the degree of doctor of philosophy,

and hereby certify that, in their opinion, it is worthy of acceptance.

Professor Hongchi Shi

Professor Yi Shang

Professor Gordon K. Springer

Professor Haibin Lu

Professor Dominic K. Ho

To my parents and my beloved wife Meng ...

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to Dr. Hongchi Shi, my dissertation advisor. His foresight and experiences in research have been the great support for me in the past several years. My thankfulness to Dr. Shi is beyond the academic research. I have benefited substantially from talking with him about the career and many other things.

I especially appreciate the constructive advices from Dr. Yi Shang. His academic insight has been a great help in my research.

I am really thankful to my committee members, Dr. Gordon K. Springer, Dr. Haibin Lu, and Dr. Dominic K. Ho, for their precious time and valuable comments on my research.

Many thanks to my labmates, Ahmed Abdol-Monem Ahmed, Xiaoli Li, Yuanliang Liu, Peng Zhuang, Malik Al Jarad, Hui Peng, and Qi Qi, for their help and friendship.

At last, special thanks to my parents and my wife. Without their love and support I could not have come this far.

Table of Contents

List of Tables

List of Figures

Abstract

A wireless sensor network (WSN) is characterized by a limited energy supply and a large number of nodes. Topology control (TC) as one of the main ways to control energy consumption in WSNs has been the focus of a considerable body of research. Network topology control is about the management of network topology to support network-wide requirements. Topology control algorithms can be divided into *transmission-power-based* algorithms and *duty-cycle-based* algorithms according to their energy saving approaches.

The contributions of the dissertation include a two-level topology control strategy, a distributed connected dominating set construction algorithm (DSP-CDS), an energy consumption analysis model to solve the optimal transmission range problem in clustered WSNs, and a distributed traffic-adaptive clustering algorithm (RDSP-CDS) for non-uniform traffic networks.

The *duty-cycle-based* and *transmission-power-based* approaches have different network conditions for them to perform well. By dynamically integrating the two approaches, I have developed a two-level topology control strategy to achieve further energy saving.

In topology control, clustering as a very promising energy saving technique can be used with either a *transmission-power-based* algorithm or a *duty-cycle-based* algorithm. Connected dominating set (CDS) is a special cluster structure where the cluster heads form a connected network without using gateways. I have designed a

distributed algorithm, DSP-CDS, for constructing CDS. DSP-CDS converges quickly in a single phase.

Motivated by the effects of clustering and transmission range on energy consumption, I have developed an energy consumption model for clustered WSNs and use it to solve the optimal transmission range problem. This model provides us an insight into the energy consumption behavior in clustered wireless sensor networks and the relationship among major factors.

Observing that traffic load often has unpredictable changes after deployment and has great impact on the optimal transmission range, I have designed a traffic-adaptive clustering algorithm, RDSP-CDS, by utilizing the energy consumption analysis. RDSP-CDS incorporates adaptive transmission range adjustment into the DSP-CDS algorithm. RDSP-CDS is suitable for dynamic network topologies due to transmission range changes, node mobility, and/or node failure.

Chapter 1

Introduction

1.1 Wireless Sensor Networks

1.1.1 Vision and challenges

Advances in wireless communication, micro-electro-mechanical system (MEMS), and digital electronics have enabled a new generation of distributed computing environment: wireless sensor networks (WSNs).

Wireless sensor networks enable us to interact with the real world to a degree that was never possible before [17] [22] [23]. A wireless sensor network consists of a large number of low-power, inexpensive, unobtrusive sensor nodes and is usually deployed in environments of interest or embedded into ordinary objects for monitoring habitats, tracking objects, and processing collected information. It can provide us real time data in a hostile environment at high fidelity over years and enable us to respond promptly.

A wireless sensor network is subject to a unique set of constraints due to limited resources and hostile environments and presents significant technical challenges. A wireless sensor node has to be low-power, inexpensive, and small-sized. A sensor node is usually battery-powered and equipped with a low-data-rate, short-range wireless

radio transceiver. Energy consumption is one of the most important factors in sensor node hardware and software design. A wireless sensor network has to be scalable, robust, essentially unattended, and self-organized, and it has to have a long lifetime. A wireless sensor network may be composed of hundreds or even thousands of nodes. The network protocols have to deal with collaboration, redundancy, data fusion, and node mobility issues in the network, and instability of wireless link, noisy environment, malicious attack, and uncertainty issues in the real world. In addition, the whole system should be cost-effective to develop, deploy, utilize, and maintain.

Figure 1.1: MicaZ and MicaDot UC Berkeley motes are commonly used in the research community.

Figure 1.2: TelOS mote is for even lower power consumption.

1.1.2 Hardware, software, and simulation environments

Researchers at Berkeley developed a hardware platform consisting of processor/radio boards commonly referred to as Motes. The UC Berkeley motes are the most commonly used sensor nodes in the research community. The Mica family motes [30] [18] (Figure 1.1) are commercial mote products. A Mica mote has a processor, a two-way ISM band radio transceiver, and a logger memory. In addition, the board offers enhanced processor capabilities, including a boot-loader that allows for over-air re-programming of Mote code. TelOS [54] (Figure 1.2) is another type of UC Berkeley motes for even lower power consumption. It supports USB and IEEE 802.15.4 for easy

use. Both Mica mote and TelOS mote use TinyOS [41], an open-source operating system designed specially for wireless sensor networks. TinyOS has a component-based architecture and uses nesC as its programming language. The nesC language has a C-like syntax and is primarily intended for programming embedded systems such as sensor networks.

Miniaturization of wireless sensor nodes makes it easier to integrate senor node into everyday objects besides the research platforms. Yamashita et al. [77] present an ultra-small, lower-power sensor node module called ZN1 (Figure 1.3 and Figure 1.4). It is equipped with temperature, vibration, and light sensors and integrates ZigBee RF. The size of the module is 15×15mm. Niedermayer et al. [49] propose a miniaturized platform based on 3D-packaging technologies to shrink a wireless sensor node with the same architecture from 26mm to 6mm per side. They expect to have the 5mm per side wireless sensor node using wafer level packaging and silicon thin film technologies.

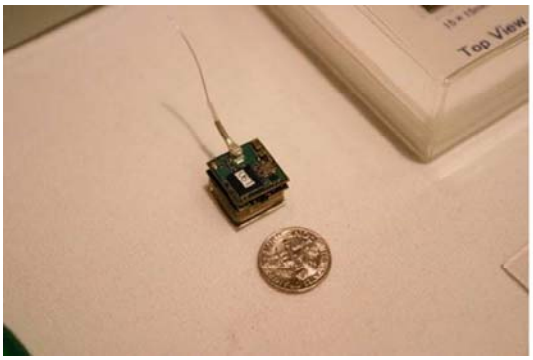


Figure 1.3: A ZN1 module (compared with a dime) is an ultra-small sensor node (Photo by Peng Zhuang).

Figure 1.4: A watch embedded with a ZN1 module and the base station communicate through ZigBee RF (Photo by Peng Zhuang).

TOSSIM [40] is a discrete event simulator for TinyOS sensor networks. It compiles directly from TinyOS code and runs natively on a PC. TOSSIM can simulate thousands of nodes simultaneously. Every mote in simulation runs the same TinyOS program. TOSSIM allows users to debug, test, and analyze algorithms in a controlled

and repeatable environment.

Marionette [74] is a tool suite for application development for WSNs. Marionette allows the developer to call functions and to read or write variables at run-time without requiring special code in the applications. It is specially useful for debugging and rapid prototyping of new applications.

NS-2 [68] is a discrete event simulator and is perhaps the most widely used network simulator. It uses the object-oriented design and is written in C++ and OTcl. With the wireless and mobility extension from the Rice Monarch Project [66], NS-2 is able to simulate mobile nodes connected by wireless network interfaces, including the ability to simulate multi-hop wireless sensor networks.

MATLAB [65] is a software environment for technical computing. For wireless sensor networks, MATLAB is convenient and powerful for mathematical modeling and customized algorithm development. It is especially powerful at performing advanced analysis, visualizing data, and developing algorithms.

1.1.3 Applications

WSNs are application oriented. Most WSN applications still remain in the research community, but they are expected to be adopted soon in a wide variety of fields.

Environmental monitoring

This is one of the driving applications of WSNs. Wireless sensor networks are deployed to monitor habitats and environmental changes over time. Long network lifetime is the most important requirement in these applications. WSNs have significant advantages on habitat and environmental monitoring, and large scale field experiments have been conducted [64] [67].

Battlefield monitoring

A wireless sensor network can be rapidly deployed to monitor the intended area on a battlefield. The network should be robust and efficiently provide information such as passing troops and vehicles.

Industrial monitoring

Wireless sensor nodes may be used to monitor and control the manufacturing process. Sensor nodes should provide accurate information in severe environments such as high temperature, loud noise, and dirty air. Quick response is often required, and actions may need to be triggered by sensor nodes without human intervention.

Intelligent home, office, and automobile

Sensor nodes are embedded in ordinary objects to collect context-sensitive data. They form a wireless network to exchange and process data and provide a safe, intelligent, and convenient living, working, or driving atmosphere.

1.2 Topology Control in Wireless Sensor Networks

Network topology control is about management of network topology to support network-wide requirements. In this definition, we consider all mechanisms that may change the network topology in the view of upper layer users (e.g., routing) as topology control. This is different from some other definitions. For example, Santi [59] only considers transmission power adjustment of nodes as topology control techniques. Another aspect of topology control is that it aims at network-wide goals, for example, extending network lifetime, guaranteeing network throughput, minimizing average delay, etc. Local optimum is not the aim of topology control by this definition.

What distinguishes the topology control problem in WSNs from that in traditional wired and wireless networks is that the network topology algorithms have to be energy-efficient, application-oriented, and executed in a distributed manner. Active sensor nodes and the wireless connections among them define the network topology. Due to the limited battery power supply in nodes, topology control algorithms should reduce energy consumption as much as the application allows. A topology control algorithm has to be application-oriented, and an algorithm that is energy-efficient in one application may not be in another. In addition, different applications have different network topology requirements for broadcasting, unicasting, and convergecasting. Due to the lack of infrastructure and the *ad hoc* characteristics in wireless sensor networks, the network topology should be computed and maintained in a distributed manner, use only local information, and be able to accommodate network changes.

1.2.1 Topology control and quality of service

Topology control and quality of service (QoS) are closely related. On one hand, QoS has to be satisfied on network connectivity, sensing coverage, communication delay, network throughput, and network responsiveness. On the other hand, a topology control algorithm should reduce energy consumption to extend the network lifetime. If we take the energy consumption as the cost of service (CoS), topology control is to reduce the CoS and maintain the QoS at some reasonable level. More specifically, topology control in wireless sensor networks is to control the wireless transmission devices and states of each node so that the total network energy consumption is minimized while the network topology is maintained for application needs.

Network connectivity

Network connectivity, perhaps, is the most fundamental requirement in a wireless sensor network. In a multi-hop network, a node must be connected to other nodes to

report collected data. Sometimes, a node needs to have at least n neighbors to satisfy routing and fault tolerant requirements, and the network should maintain n-degree connectivity. The transmission power changes of nodes by topology control change the neighborhood of a node and hence change the connectivity of the network.

Sensing coverage

One main purpose of a wireless sensor network is to "sense" an space of interest, and coverage is one of the basic requirements of an application. Full coverage is often not necessary, and some percentage of uncovered space is either not important or can be estimated from known information. However, some applications may need n-degree coverage so that any point in the space needs to be covered by at least n sensor nodes. A sleeping node may not be able to sense or at least not be able to report the sensing data in a timely manner. The duty-cycle changes of nodes by topology control put some nodes into the sleeping state and therefore change the coverage of the network.

Communication delay

End-to-end communication delay can be measured by the time it takes to successfully send and receive a packet between two nodes. Low duty cycle of nodes along the path increases the communication delay due to buffering or dropping packets. Transmission power adjustment of nodes affects the path length (hop counts) and hence affects the communication delay.

Network throughput

An application may have a throughput requirement on the network. For example, from a node A to node B there must exist some routes to transfer some amount of data within some amount of time. Regardless of what topology is generated, a topology control algorithm should guarantee the minimum throughput requirement

of the application.

Network responsiveness

In a time-sensitive application like security surveillance, detection of events must be reported within limited delay. Detection latency may be increased because of sensor nodes working in the low duty cycle, communication delay, or both. The topology control algorithm should guarantee the responsiveness requirement when scheduling duty cycles of nodes.

1.2.2 Topology control and other aspects of WSNs

Topology control has a close relationship with deployment, media access, and routing in the network.

Topology control and deployment

Sensor nodes may be deployed randomly or in a structured manner. The network may have a uniform or nonuniform node density. Nodes in the network can be of the same type to form a homogenous network, or have various capabilities to form a heterogenous network. Topology control in a homogenous network with a uniform structure is simpler because of the *a priori* knowledge about the relative location and capability of other nodes in the network. Without this *a priori* knowledge, topology control is more challenging in post-deployment configurations, and more complex protocols are needed to collaborate the work among nodes.

Topology control and MAC

Topology control is so closely related to media access control (MAC) that a simple topology control algorithm could be directly built into MAC. A topology control algorithm depends on the mechanisms provided by MAC to schedule the activities

of nodes. The abilities of MAC (for example, whether the duty-cycle is supported in MAC) and the overhead of switching from sleep to active mode affect the topology control decision on whether or when a energy saving method is applied.

Topology control and routing

Topology control directly provides the underlying network structure for routing protocols. The choice of a particular topology control algorithm has a strong impact on the choice of routing protocols, and vice versa. Furthermore, routing protocols should be robust enough to cope with constant topology changes in wireless sensor networks.

1.3 My Research Work on Topology Control

I establish a new model for better understanding the energy consumption behavior in clustered WSNs, especially the relationship among energy consumption, transmission power, and clustering. I develop two new energy-efficient topology control algorithms by utilizing both clustering and adjusting transmission power. I also develop an efficient clustering algorithm in *ad hoc* networks to construct a connected dominating set in a single phase.

The two-level topology control strategy is designed to utilize both transmission power and duty cycle for further energy consumption (Chapter 3). Network clustering is one of the efficient approaches to saving energy and keeping network functionality by keeping cluster heads active and turning other nodes into the state of low duty cycle. DSP-CDS is an efficient distributed clustering algorithm (Chapter 4). Unlike existing algorithms which need two or more phases, DSP-CDS converges quickly in a single phase and generates a CDS of comparably small size.

To understand various factors that affect the performance of topology control

algorithms, an energy consumption model (Chapter 5) is established for clustered wireless sensor networks and is used to solve the optimal transmission range problem. Using this model, the total energy consumption can be estimated beforehand based on the traffic pattern, energy model, and network deployment parameters. This model provides an insight into the energy consumption behavior in clustered wireless sensor networks and the relationship among major factors.

I also design a distributed traffic-adaptive topology control algorithm, RDSP-CDS (Chapter 6), based on the energy consumption model established in Chapter 5 and the DSP-CDS algorithm in Chapter 4. This adaptive algorithm makes transmission power and clustering decisions using information collected about the traffic load.

1.4 Dissertation Organization

This dissertation is organized as follows. Chapter 2 is the literature review. Chapter 3 presents the two-level topology control strategy for energy saving. Chapter 4 is the distributed single-phase algorithm for constructing connected dominating set. Chapter 5 establishes the energy consumption model in clustered *ad hoc* networks. Chapter 6 presents the traffic load adaptive clustering algorithm, RDSP-CDS. Chapter 7 concludes the dissertation.

Chapter 2

Topology Control Algorithms in Wireless Sensor Networks

Topology control (TC) as one of the most important energy-saving approaches in WSN has been the focus of a considerable body of research, including theoretical study and algorithm design. Theoretical study in topology control concerns the optimum result of the network topology instead of how to achieve it. Due to my research interest, this survey focuses mainly on the topology control algorithms, which is the process of building and maintaining a desired network topology from the original network. Theoretical studies are only briefly mentioned in Section 2.2.

2.1 A Taxonomy of Topology Control Algorithms

A wireless sensor node may change the network topology by two mechanisms: transmission power and duty cycle, which are also the two mechanisms that can be utilized by a topology control algorithm for energy saving. Based on the informal definition of TC in Chapter 1.2, the topology control algorithms using the two approaches are referred to as *transmission-power-based* algorithms and *duty-cycle-based* algorithms.

Clustering is a technique used in networks to divide nodes into two categories,

cluster heads and normal nodes. Each normal node can directly communicate with at least one cluster head, and the cluster head usually takes more responsibility. In wireless sensor networks, clustering can be either used with a transmission-power-based algorithm (as in *LEACH* [29]) to allow the normal nodes to use just enough transmission power to reach a cluster head or used with a duty-cycle-based algorithm (as in *Span* [11]) to turn normal nodes into a low duty cycle, both for energy saving. This dissertation is especially interested in the connected dominating set (CDS) based clustering algorithms for their desired connectivity feature among cluster heads. Section 2.5 is dedicated to CDS clustering algorithms.

Adaptive topology control algorithms update the transmission powers or duty cycles of nodes based on some continually changing conditions. They will be introduced in Section 6.2

Given a wireless transmitter and a wireless receiver, the largest distance or range between them to successfully transmit a signal is decided by the transmission power. In order to maintain the lowest acceptable received signal power over a given range, the transmission power can be decided under a regular radio propagation model. In this dissertation, the transmission power and the transmission range are used interchangeably.

2.2 Theoretical Study

Gupta and Kumar [27] have studied the critical transmission range for connectivity problem: What is the minimum common transmission range of nodes to ensure network connectivity? They have proved that if n nodes are uniformly placed in the area of a disk unit, and the transmission range is set to $r = \sqrt{\frac{\log n + c(n)}{\pi n}}$, then the resulting network is asymptotically connected with probability of one if and only if $c(n) \to +\infty$. Using a short transmission range can reduce communication interfer-

ence, but it increases the path length and is not necessarily a good choice in terms of energy saving.

The transmission-power-based topology control approach can be stated as the transmission range assignment problem: assigning a transmission range to each node in the network so that the resulting network is strongly connected and the energy cost is minimized. Kirousis et al. [36] have shown that the problem is solvable in polynomial time for one-dimensional networks and is NP-hard in three-dimensional networks. Clementi et al. [16] have shown that the problem is NP-hard in two-dimensional networks as well. In a later paper, Blough et al. [6] have considered the symmetric transmission range assignment problem, which adds a symmetric constraint to the transmission range assignment problem so that the resulting network contains only bidirectional links. The authors have shown that the symmetric transmission range assignment problem remains NP-hard.

The most common way to use the duty-cycle-based approach for energy saving and maintain the network coverage at the same time is to arrange nodes into clusters. What is the minimum number of clusters needed given a fixed transmission range of nodes? This is the same as the problem of covering with unit disks. Given a set of points in the Euclidean plane, find the a minimal set of unit disks to cover all the points. This problem is known to be NP-complete [15]. A related problem is to cover a plane with unit disks, and Kershner [35] has shown that the most efficient arrangement of disks to cover the plane is the hexagonal lattice arrangement shown in Figure 2.1. To cover a square of side length σ, Verblunsky [69] has given a lower bound on the number of unit disks: $3^{\frac{3}{2}}N(\sigma) > \sigma^2 + \sigma$, where $N(\sigma)$ is the number of unit disks.

In clustering methods, cluster heads form a dominating set of the underlying graph of the original network. For a graph G, a set S of vertices is a dominating set (DS) of G if every vertex of G is either in S or adjacent to a member of S. A dominating set S

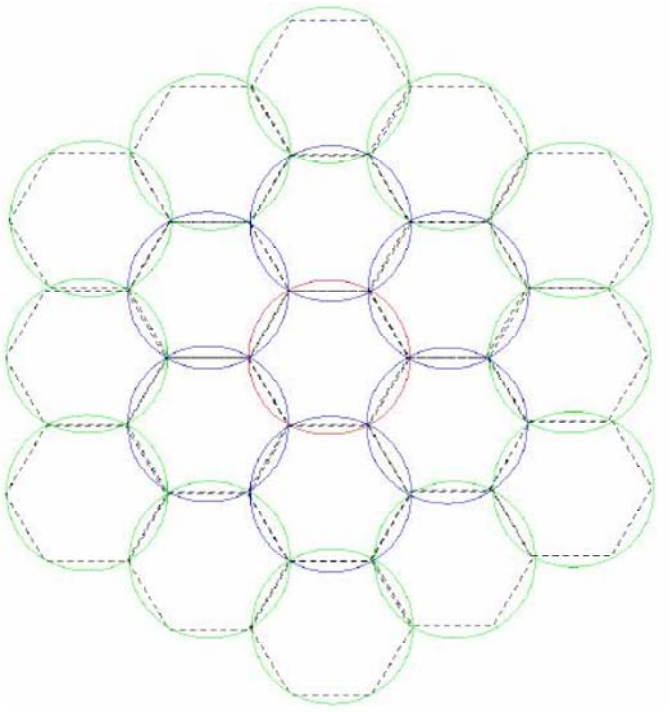


Figure 2.1: The hexagonal lattice is the most efficient arrangement of disks to cover the plane (Figure courtesy of [53]).

is a connected dominating set (CDS) if the subgraph of G induced by S is connected. A node in the dominating set is called *dominator*. A small size CDS is desirable in many applications. A CDS of minimum size is called minimum connected dominating set, or MCDS. Finding an MCDS in a general graph is an NP-complete problem [25]. A CDS is a good choice for network backbones. Energy consumption may be reduced by turning nodes not in the CDS into low duty cycle, and the network functionality can be conducted (at a lower level) by the active nodes in the CDS.

2.3 Transmission-Power-Based TC Algorithms

Radio propagation models have been used to predict the average received signal strength at a given distance from the transmitter. In the free space radio propagation model, the received power falls off with distance raised to the αth power, where α is the path loss exponent usually from 2 to 6 depending on the environment [56]. The *transmission-power-based* approach concerns how to adjust transmission power of nodes to reduce energy consumption while maintaining the network connectivity and topology.

The algorithm by Hou and Li [31] uses the lowest possible transmission power to reach the nearest neighbor in the forwarding direction to reduce collision and improve network performance. The energy consumption issue is not considered in their work.

Rodoplu and Meng [58] describe a distributed algorithm to compute the minimum total power consumption path for every node in the network to reach a sink node. Each node is able to dynamically adjust its transmission power to reach the next hop node on the shortest path (with energy consumption as cost) to the sink. Their algorithm requires accurate position information of nodes. The links between nodes in their topology are not symmetric, which is not desired for a network protocol design.

LINT and *LILT* [55] use two distributed heuristics to adaptively adjust node

transmission power in response to topology change and attempt to maintain a connected topology using the minimum power. *LINT* uses locally available neighborhood information collected by a routing algorithm and attempts to keep the number of neighbors of each node bounded. *LILT* additionally exploits the global topology information with some routing algorithms.

LEACH [29] is a cluster-based algorithm. Every node has data to transmit to a base station. Cluster heads are one hop away from the base station, and other nodes forward their data to the cluster heads. The operation of *LEACH* is broken up into rounds, and cluster heads are re-elected in each round to achieve even energy consumption among the nodes. A cluster head creates a TDMA schedule to communicate with nodes in its cluster. In order to save energy, a non-cluster-head node turns off its radio until its allocated time slot. *LEACH* uses the strength of received signal as the indicator of distance between nodes and does not require node position information. However, it requires synchronization in a cluster due to the use of TDMA. In a later paper, Hussain and Matin [33] have proposed a centralized off-line algorithm that can compute a better selection of cluster heads than *LEACH*.

Wattenhofer and Li [73] [42] have proposed a distributed cone-based topology control algorithm, *CBTC*. In the first step, each node uses the lowest transmission power to include at least one node in every cone of α degrees. Then, the algorithm improves the graph generated from the first step by removing asymmetric edges and redundant nodes. They have shown that $\alpha \geq \frac{5\pi}{6}$ is needed to ensure a connected network. The *CBTC* algorithm is a distributed localized algorithm and can generate a network with symmetric links. It does not need network synchronization and node position information. However, it needs the directional information of nodes, which requires additional hardware (e.g., more than one directional antenna).

LMA and *LMN* [37] are two distributed algorithms that reduce the transmission power by controlling the number of neighbors. In *LMA*, each node uses a transmission

power to ensure that the number of neighbors is in the interval

$$[NodeMinThresh, NodeMaxThresh],$$

where $NodeMinThresh$ and $NodeMaxThresh$ are the minimum and maximum thresholds on the number of neighbors. The *LMN* algorithm is similar to *LMA*, except that a node controls the mean number of neighbors from its neighbors. The algorithms are very simple and only the ID of each node, but the resultant network is not symmetric.

LMST [44] is a Minimum Spanning Tree (MST) based topology control algorithm. Each node builds its local minimum spanning tree and only keeps one-hop on-tree nodes as neighbors in the final topology. The authors have shown that the algorithm preserves the network connectivity and the degree of each node is bounded by 6. Like *CBTC*, it needs additional processing to obtain a symmetric resulting network. Unlike *CBTC*, it obtains the connected symmetric network by simply removing all uni-directional links. The *LMST* algorithm is distributed and localized, but it requires node position information. The authors [43] have later proposed two localized transmission-power-based topology control algorithms (DRNG and DLSS) to extend their work to heterogenous networks where nodes may have different maximal transmission power.

The algorithm developed by Liu and Li [47] is for heterogenous networks as well. Each node establishes its local vicinity topology using the broadcast messages from its neighbors. The local vicinity topology of node i is represented with a weighted, directed graph $\overrightarrow{G_i} = (V_i, \overrightarrow{E_i})$, where V_i is the collection of node i's vicinity nodes and $\overrightarrow{E_i}$ is the collection of node i's vicinity edges. A node is node i's vicinity node if it can receive the broadcast messages from node i when node i uses its maximum transmission power. For any two nodes $j, k \in V_i$, link $\overrightarrow{L_{jk}}$ is called node i's vicinity edge, if $P_j^{max} \geqslant P_{jk}$. The weight of $\overrightarrow{L_{jk}}$ is P_{jk}. Given the weighted, directed graph $\overrightarrow{G_i}$,

node i executes a single-source shortest path (SSSP) algorithm to obtain an SSSP tree with node i as the root. Node i is assigned a transmission power required to reach the farthest one-hop node in the SSSP tree.

Recently, the transmission-power-based topology control algorithms are proposed based on the cooperative communication. Cooperative communication [50] [60] is a novel model to re-think the link model in mobile *ad hoc* networks, which allows the single-antenna network nodes to form a virtual multiple-input multiple-output (MIMO) system and reap some of the benefits of MIMO systems, like transmit diversity. Cardei et al. [7] have introduced the topology control with cooperative communication (TCC) problem, which employs cooperative communication in topology control to maintain a specific topology with reduced energy consumption. The TCC problem is about how to assign the transmission powers of nodes so that the power level in all nodes is minimized based on a cooperative communication model.

2.4 Duty-Cycle-Based TC Algorithms

A radio device uses different powers for transmitting, receiving, idling, and sleeping. Transmitting power is greater than the others and can be different for radio coverage requirements. Receiving and idling usually consume similar powers. Studies [61] on real hardware have shown that the power ratio between Tx/Rx/Idle and Sleeping power consumption for 802.11 wireless LAN cards can be 10:1. For the lower power radio, that ratio can be on the order of 100:1. For example, Cerpa and Estrin [8] have reported the values of 36:9:9:0.015 (in mW) for Tx:Rx:Idle:Sleep about the RFM Tx-1000 [57].

In typical wireless sensor networks, a node spends a great amount of time in the idle mode (the state of not receiving or transmitting packets), and the energy consumption in the idle mode can not be neglected. Energy could be saved by turning

off the radio when not in use. The *duty-cycle-based* approach concerns how to schedule the duty cycle of nodes to reduce energy consumption while maintaining the basic application functionality.

Nodes in the network can be synchronized to sleep and wake up to provide service only a fraction of time. But, we are more interested in using the clustering method with duty cycle, in which the network functions all the time. Cluster heads form a dominating set of the network and work in the active mode. Other nodes operate in the low-duty-cycle mode to reduce energy consumption. Usually nodes rotate to be cluster heads to extend the entire network lifetime.

Span [11] is a distributed algorithm for power saving in *ad hoc* wireless networks. The nodes make their local decisions on whether to enter the power-saving mode or the active mode. The active nodes, called coordinators, have a higher priority in forwarding packets and form a backbone subnetwork to connect all the sleeping nodes. Periodically, a non-coordinator node makes a decision on whether it should become a coordinator, and a coordinator node makes a decision on whether it should become a non-coordinator. *Span* works in *ad hoc* networks in which all nodes use the same transmission power for communication. In addition to saving energy, *Span* aims at throughput conservation as well. But the algorithm does not guarantee a connected active subnetwork.

SBPM [32] is a sentry-based power management scheme for applications such as intruder detection and tracking. *SBPM* makes use of sentries to define a coarse network of nodes that are necessary to perform the task and turn on non-sentry nodes when necessary. Besides energy saving, *SBPM* emphasizes keeping network coverage, and non-sentry nodes can switch to full-power mode when refined sensing is needed.

GAF and *CEC* [76] are also duty-cycled-based topology control algorithms. *GAF* determines redundant nodes and controls the node duty cycle to extend the network operational lifetime while maintaining the network connectivity, independent of the

involvement of *ad hoc* routing algorithms. *CEC* maintains local connectivity with low overhead and is thus able to dynamically adapt to a changing network.

In clustering methods, cluster heads have special responsibilities in maintaining the topology and information, and re-election of cluster heads in mobile networks are usually expensive. QoS-guaranteed algorithms replying on the clustering topology is more sensitive to the stability of the cluster structures. Alkahtani and Mouftah [1] have proposed the Smooth and Efficient Re-Clustering (SERC) protocol to enhance the stability of a clustered network. In SERC, a secondary cluster head is identified by the primary cluster head and known to all the cluster members so that the cluster leadership can be transferred more smoothly when the primary cluster head quits. Chatterjee et al. [10] have proposed a node-weighted clustering algorithm, WCA. WCA computes a heuristic weight for each node during cluster head election, and the weight includes factors like neighborhood topology, node mobility, transmission power, and battery level. Clustering stability can be achieved by assigning proper weights to these factors.

Stability of the cluster structure is often not the ultimate goal in an application. Actually, clustering itself is often a convenient stability support for upper layer protocols. Chiu et al. [13] have proposed a GCR routing protocol which utilizes the underlying clusters to repair the local routing path so that stability of connections is improved and some QoS properties can be supported.

Although clustering is one of the most promising techniques in wireless sensor networks for energy saving, it is not always a better choice than the non-clustered network. Vlajic and Xia [70] have discussed the conditions under which the clustered organization outperforms the non-clustered organization.

2.5 CDS Clustering Algorithms

The existing algorithms for constructing connected dominating set (CDS) usually use one of the following approaches. One approach is to grow a tree from a root. The set S initially contains one node and then repeatedly expands itself to include some neighbors of the nodes in S until S is a CDS. Another approach is to form separate small CDS-like pieces in the first phase (clustering phase) and construct a CDS by connecting the clusters together in the second phase (connecting phase). The third approach is to construct a crude CDS with a large number of nodes in the first phase and prune some redundant nodes without disconnecting the dominators in the second phase. In this paper, algorithms using the three approaches are referred to as *Type-I*, *Type-II*, and *Type-III*, respectively. An algorithm using any of the three approaches can be either centralized or distributed.

Distributed CDS construction algorithms in literature usually have two or more phases, because some extra care must be taken to make sure the dominating set is connected. A distributed *Type-I* algorithm starts the dominating set from a root which is elected in the first phase, and only nodes neighboring the current dominators are considered as candidates each round in the second phase. Therefore, connectivity of the dominating set can be guaranteed. A distributed *Type-II* algorithm usually needs to maintain a global structure (e.g., a spanning tree), by which the connecting phase can make sure the resultant CDS is connected. The clustering phase in a *Type-II* algorithm can be seen as the dominating set problem, for which many distributed algorithms have been proposed in literature [39] [38] [34] [45]. A *Type-III* algorithm does generate a CDS in the first phase, but the crude CDS is far less desired. For example, the CDS generated in the first phase by Wu and Li's algorithm [75] contains too many redundant nodes.

Guha and Khuller's algorithms I and II [26] are centralized algorithms of *Type-I* and *Type-II*, respectively. Algorithm I is a greedy algorithm, and it grows the

connected dominating set from a vertex with the maximum degree. In each round, it selects into the dominating set one node or a pair of nodes that are neighbors to the current dominators and that have the largest *yield*. The *yield* of a node (or a pair of nodes) is defined as the total number of neighbors of the node (or the pair of nodes) that are not dominated. The authors have proved that algorithm I generates a CDS of size at most $2(1 + H(\Delta)) \cdot \mid OPT_{DS} \mid$, where Δ is the maximum degree, H is the harmonic function, and OPT_{DS} is the optimal dominating set. Algorithm II runs in two phases. Initially, all nodes are white. In each step of the first phase, a node that can reduce the maximum number of *piece*s is marked black, and its white neighbors are marked gray. A *piece* is defined as a white node or a black connected component. In each step of the second phase, pieces are recursively connected by choosing a chain of two vertices. Finally, all black nodes form a CDS of the network.

Das and Bharghavan's algorithms I and II [19] are distributed *Type-II* and *Type-I* algorithms, respectively. Algorithm I first finds a spanning forest of the network in the first phase and then connects the fragments by using a distributed minimum spanning tree algorithm in the second phase. Algorithm II first starts a segment from somewhere in the network and then grows the segment into a dominating set by adding extensions. An extension to the segment can be either one- or two-edge path consisting of one node in the fragment and one or two nodes not in the fragment. In mobile environments, the algorithms treat multiple-node movement as separate single-node movement (categorized as movement of a non-CDS node, a leaf CDS node, and an interior CDS node) and update the CDS accordingly.

Wu and Li [75] have proposed a *Type-III* algorithm. In the first phase, every node v exchanges its neighbor set, $N(v)$, with all of its neighbors and becomes a dominator if it has two unconnected neighbors. Thus, each node uses the information of its two-hop neighborhood to make a decision. The authors prove that all dominators form a CDS in the first phase. In the second phase, some dominators are removed from the CDS

using two rules. The first rule removes a dominator v if $N(v) \cup v$ is covered by another dominator u and $id(v) < id(u)$. The second rule removes a dominator v if $N(v) \cup v$ is covered by two other dominators u and w and $id(v) = min(id(v), id(u), id(w))$. To deal with the mobility issues, the authors summarize the topology changes as three types: node turns on, node turns off, and node moves. In their discussion, a mobile node sends out special signals to notify its neighbors about its changes, and the neighbors take actions accordingly upon receiving those signals to maintain the CDS.

Alzoubi et al. [2] have proposed a *Type-II* algorithm, which uses node *id* to select dominators in the clustering phase and connects them by paths between disconnected dominators in the connecting phase. Chen and Liestman's algorithms I and II [12] are similar to Guha and Khuller's algorithms II and I, respectively, except that Chen and Liestman use them to construct the weakly-connected dominating set. Chen and Liestman also provide a distributed version of their two algorithms, algorithms III and IV, respectively. In both distributed algorithms, information is passed over to a special arbitrator node to decide the dominators. In algorithm III, a rooted spanning tree is constructed beforehand, and its root is the arbitrator. In algorithm IV, the arbitrator is the node from which the CDS grows.

The algorithm by Wan et al. [71] is a distributed *Type-I* algorithm, and it has three phases. The algorithm constructs a rooted spanning tree using the algorithm proposed by Cidon and Mokryn [14] in the first phase, then a maximal independent set (MIS) based on the spanning tree in the second phase, and a dominating tree based on the MIS in the third phase. The internal nodes of the dominating tree form a CDS. Wan et al. prove that their algorithm has an approximation factor of at most 8 and achieves $O(n)$ time complexity and $O(n \log n)$ message complexity, which is message optimal. However, their paper does not discuss the maintenance of CDS in a mobile environment.

Zhou et al. [83] have proposed a MAC layer timer-based algorithm, MT-CDS, which is a distributed *Type-I* algorithm. In the first phase, the node with the smallest ID is elected as the initiator in a distributed way, and the initiator becomes the first node in the dominating set. In the second phase, nodes in the dominating set periodically broadcast their status messages. Upon receiving a status message, a neighboring node starts a timer, which, upon expiring, turns the node into a dominator. A node updates its timer to reflect the changes in its neighborhood, and a larger number of uncovered neighbors leads to a shorter timer. A node with no uncovered neighbors never has the timer expired. Thus, nodes with more uncovered neighbors will become dominators each round. The authors show that MT-CDS can maintain a generated CDS under network topology changes. Those changes include withdrawing of the initiator (which causes a new round of initiator election), withdrawing of a dominator, and joining of a new node.

Bao and Garcia-Luna-Aceves [5] use a priority method to first select a minimal dominating set (MDS) and then form a CDS by adding more gateway nodes into the MDS. Their priority method considers several factors, e.g. node ID, energy level, and mobility, which is different from the above algorithms that usually use one factor in dominator election. Their algorithm is a *Type-II* algorithm.

Chapter 3

A Two-Level Topology Control Strategy

3.1 Introduction

The *transmission-power-based* and *duty-cycle-based* approaches discussed in Chapter 2.1 have different network conditions for them to perform well, and node density is a major factor. Intuitively, when the node density is high (compared with application needs), there are many redundant nodes, and the *duty-cycle-based* approach has better performance. When the node density is low, on the other hand, most nodes have to be active, and the *transmission-power-based* approach seems better. The two-level topology control strategy is proposed to integrate the *duty-cycle-based* and *transmission-power-based* approaches for further energy saving in wireless sensor networks. First, a *duty-cycle-based* algorithm is applied on the whole network to generate an active subnetwork. Then, each node executes a *transmission-power-based* algorithm to adjust its transmission power according to its local knowledge about the active subnetwork. In this two-level strategy, the *duty-cycle-based* algorithm and the *transmission-power-based* algorithm interact dynamically. The transmission range

changes of nodes affect how to form the active subnetwork, while topology changes in the active subnetwork affect how to determine nodes' transmission ranges at the same time.

Due to the distributed nature of wireless sensor networks, topology control algorithms are desired to be distributed. In the implementation, *Span* [11], which is a *duty-cycle-based* algorithm, is used as the basis of the two-level topology control strategy, while a transmission range adjusting algorithm [47] is used as one of the instances of the *transmission-power-based* algorithms. They will be introduced next.

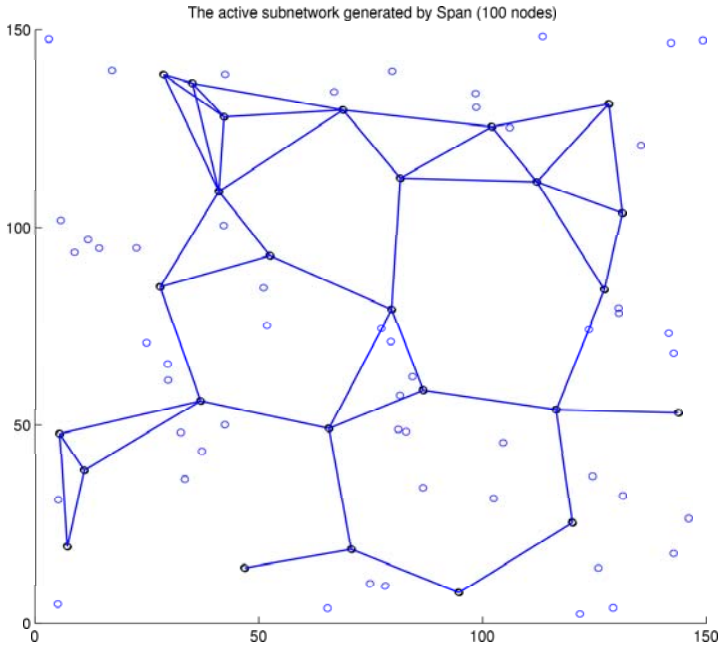


Figure 3.1: Coordinators in *Span* form an active subnetwork.

3.1.1 *Span*, a duty-cycle-based algorithm

Span [11] is a distributed algorithm for power saving in *ad hoc* wireless networks. The nodes make their local decisions on whether to enter the power-saving mode or the active mode. The active nodes, called coordinators, have a higher priority

in forwarding packets and form a backbone subnetwork to connect all the sleeping nodes. Figure 3.1 is a typical subnetwork generated by *Span*. Black points are coordinators, and the lines show the connections between coordinators. Each non-coordinator (white point) has at least one coordinator as its neighbor.

The *Span* implementation is based on IEEE 802.11 Power-Saving Mode (PSM). In IEEE 802.11 PSM, nodes in power-saving (PS) mode are synchronized to sleep and wake up periodically. During sleep, nodes switch off their wireless transceivers. Frames to the sleeping nodes are buffered at the senders. When awake, senders notify receivers with *ad hoc* traffic indication messages (ATIMs) about the buffered frames, and the receivers that have received ATIMs should stay awake until the frames are delivered.

In *Span*, a non-coordinator node periodically makes a decision on whether it should become a coordinator using the following *coordinator eligibility rule*. This ensures that the entire network is covered with enough coordinators.

Span coordinator eligibility rule: *A non-coordinator node should become a coordinator if it discovers, using only information gathered from local broadcast messages, that two of its neighbors cannot reach each other directly or via one or two coordinators.*

In order to avoid announcement contention, *Span* adds to every coordinator announcement a back-off delay calculated from the following expression:

$$delay = \left(\left(1 - \frac{E_r}{E_m}\right) + \left(1 - \frac{C_i}{\binom{N_i}{2}} \right) + R \right) \times N_i \times T, \quad (3.1)$$

where E_r is the remaining energy in node i, E_m is the maximum/initial energy, N_i

is the number of neighbors for node i, C_i is the number of additional pairs of nodes among these neighbors that would be connected if i were to become a coordinator, R is a random number in the range $(0, 1]$, and T is the round-trip delay for a small packet over the wireless link.

Periodically, a coordinator node makes a decision on whether it should become a non-coordinator using the *Span coordinator withdrawal rule.*

Span coordinator withdrawal rule: *A coordinator node should withdraw as a coordinator if it discovers, using only information gathered from local broadcast messages, that:* 1. *every pair of its neighbors can reach each other directly or via one or two other coordinators, or* 2. *after some period of time, every pair of neighbor nodes can reach each other via one or two neighbors.*

The second part in this rule is to rotate the coordinators among all nodes. In order to prevent a temporary loss of the network connection, a withdrawn coordinator works as a tentative coordinator for some time (called grace period) and can still be used to forward packets.

The *Span* rules do not guarantee a connected active subnetwork. However, a simple technique is used to wake up additional nodes if necessary during packet delivery.

Span works in *ad hoc* networks in which all nodes use the same transmission power for communication.

3.1.2 *NSP*, a transmission-power-based algorithm

The algorithm developed by Liu and Li [47] is for wireless sensor networks where each node can adjust its transmission power and the maximum transmission power of each node may be different. It is referred to as Neighborhood Shortest Path (*NSP*) algorithm in this dissertation.

When each node uses its maximum transmission power, the network topology is assumed to be a strongly connected graph: $\overrightarrow{G} = (V, \overrightarrow{L})$. For node i, P_i is its

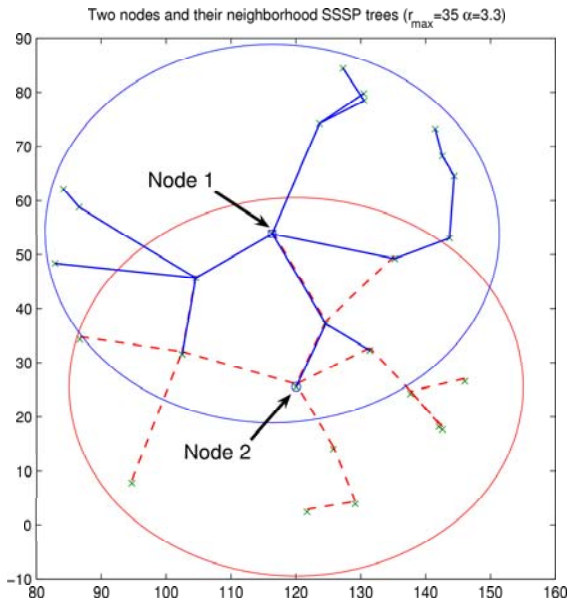


Figure 3.2: The neighborhood SSSP trees of node 1 and node 2 overlap in the overlapping area of their radio coverage circles ($r_{max} = 35$).

transmission power, P_i^{max} is its maximum transmission power, and P_{ij} is the power required for node i to reach node j with $P_{ij} = P_{ji}$. There exists a link $\overrightarrow{L_{ij}}$ if and only if $P_i^{max} \geq P_{ij}$.

Each node establishes its local vicinity topology using the broadcast messages from its neighbors. The local vicinity topology of node i is represented by a weighted, directed graph $\overrightarrow{G_i} = (V_i, \overrightarrow{E_i})$, where V_i is the collection of node i's vicinity nodes. A node is node i's vicinity node if it can receive the broadcast messages from node i when node i uses its maximum transmission power. $\overrightarrow{E_i}$ is the collection of node i's vicinity edges. For any two nodes $j, k \in V_i$, link $\overrightarrow{L_{jk}}$ is called node i's vicinity edge, if $P_j^{max} \geq P_{jk}$. The weight of $\overrightarrow{L_{jk}}$ is P_{jk}.

Given the weighted, directional graph $\overrightarrow{G_i}$, node i executes a single-source shortest path (SSSP) algorithm to obtain an SSSP tree with node i as the root. Node i is assigned a transmission power required to reach the farthest one-hop node in the SSSP tree. Figure 3.2 illustrates two nodes and their neighborhood SSSP trees, where all

nodes in the network have the same $r_{max} = 35$. The neighborhood SSSP tree of node 1 (in the upper circle, connected by solid lines) and that of node 2 (in the lower circle, connected by dashed lines) overlap in the overlapping area of their radio coverage circles. When node 1 and node 2 adjust their transmission powers, they can still reach each other by the intermediate nodes. The network keeps connected when all nodes adjust their transmission powers according to their neighborhood SSSP trees.

3.2 Two-Level Topology Control

This chapter considers the following type of wireless sensor networks: All nodes are randomly deployed in a rectangular area, each node is equipped with an omni-directional antenna with adjustable transmission range, and all the nodes in the network have the same maximum transmission range.

The concepts of *maximum neighbors* and *effective neighbors* are introduced to help distinguish neighbors with different distances to the interested node with regard to the current transmission range in a transmission range adjustable wireless network.

Assume that all the nodes in a wireless sensor network N have the same maximum transmission range r_{max}. Let D_{ij} denote the Euclidean distance from node i to node j with $D_{ij} = D_{ji}$. The current transmission range of node i is denoted by r_i and is used for data packet delivery at the current time. Node j is node i's *maximum neighbor* if $D_{ij} \leq r_{max}$. Node j is node i's *effective neighbor* if $D_{ij} \leq r_i$ and $D_{ji} \leq r_j$.

Because all the nodes in N have the same maximum transmission range r_{max} and a stricter condition is used for effective neighbors, the links between node i and its effective and maximum neighbors are guaranteed to be bidirectional.

The two-level topology control strategy integrates the *duty-cycle-based* approach and the *transmission-power-based* approach for further energy saving in wireless sensor networks. First, a *duty-cycle-based* algorithm is applied on the whole network to

generate an active subnetwork. Then, each node executes a *transmission-power-based* algorithm to adjust its transmission power according to its local knowledge about the active subnetwork. Both the *duty-cycle-based* and the *transmission-power-based* algorithms should be adapted to fit into the two-level topology control strategy. In addition to considering energy consumption and topology changes, the *duty-cycle-based* algorithm should be adaptive to the transmission range changes of a node. The transmission range, which is dynamically changed by the *transmission-power-based* algorithm, affects which node is selected into the active subnetwork. The active subnetwork is the communication backbone and delivers nearly all the packets in the network. The *transmission-power-based* algorithm computes a node's transmission range only based on its maximum coordinator neighbors instead of all maximum neighbors. Because most neighbors of a node usually work in the power-saving mode, the transmission range computed based on them may not include (enough) coordinators to forward packets effectively. Therefore, the transmission range used by a node is decided by its local knowledge about the active subnetwork, which is dynamically changed by the *duty-cycle-based* algorithm.

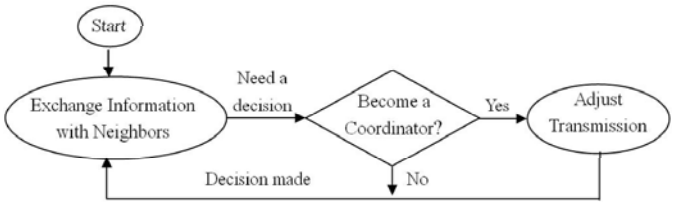


Figure 3.3: A node using the two-level topology control works in rounds.

3.2.1 Collecting local information

From the point of view of each node, the two-level topology control strategy can be shown by Figure 3.3. Each node uses *Hello* messages to collect information about its maximum neighbors. Each node periodically broadcasts *Hello* messages using r_{max}.

A *Hello* message contains the node's position, states (whether it is a coordinator or not), current neighbors (including both coordinator neighbors and non-coordinator neighbors), and current transmission range. A *Hello* message broadcast can also be triggered by the transmission range changes in a node in order to maintain the bidirectional links with its effective neighbors.

Each node updates its local knowledge about its maximum neighbors with each received *Hello* message. An update of the current transmission range is needed in three cases:

1. A non-coordinator becomes a coordinator;
2. a coordinator becomes a non-coordinator; and
3. a coordinator changes its transmission range.

In any of the above cases, a node applies the *transmission-power-based* algorithm on its maximum coordinator neighbors to compute the new transmission range. If the new value differs from the current one, the node updates its current transmission range and broadcasts this change to all of its maximum neighbors.

3.2.2 Forming active subnetwork

In a wireless sensor network, some of nodes called coordinators form an active subnetwork as the communication backbone. The coordinator election rule decides when and which node should become a coordinator.

In *Span*, the coordinator election rule considers topology changes made by a candidate coordinator and its remaining energy. Given a transmission range adjustable network, it is reasonable to take the transmission range into account. Generally, a coordinator with a smaller transmission range is preferred if all the other conditions are kept the same. For the convenient incorporation of the transmission range into Equation (3.1), a new term from the interval $(0, 1]$ is desired. The following one is

used in the implementation

$$\frac{r}{r_{max}},$$

where r is the current transmission range of the node and r_{max} is the maximum transmission range.

Plugging the new term into the *Span* back-off delay Equation (3.1), we obtain a new back-off delay for the transmission range adjustable network:

$$delay = \left(\left(1 - \frac{E_r}{E_m}\right) + \left(1 - \frac{C_i}{\binom{N_i}{2}} \right) + \frac{r}{r_{max}} + R \right) \times N_i \times T. \quad (3.2)$$

3.2.3 Adjusting transmission range

Two transmission range adjusting algorithms are implemented: FNC and NSP.

FNC (Farthest Neighbor Coordinator)

A node uses a transmission range just enough to cover the farthest coordinator in its maximum neighbors. This algorithm attempts to have the minimum transmission range to keep the backbone unchanged.

NSP (Neighborhood Shortest Path)

A node sets its transmission range only as long as the distance to the farthest one-hop coordinator in its local SSSP tree. This algorithm attempts to have the minimum transmission range to reach all its one-hop coordinators.

In *NSP*, the SSSP tree is built with the same method as described by Liu and

Li [47] with two changes. First, the neighbors of a node in [47] form a directed graph, while we have an undirected graph, since all the nodes are assumed to have the same maximum transmission range r_{max}. Secondly, all the maximum neighbors in [47] are vertices in the neighborhood graph, while only maximum coordinator neighbors are used in our implementation.

FNC is very simple and achieves acceptable performance. *NSP* generally achieves better performance over *FNC* at the price of more computing cost and memory usage to maintain the SSSP tree.

3.2.4 Routing with reduced transmission range

Transmission range changes affect the routing algorithms. Usually, the next-hop node is found in the effective neighbors. In order to avoid the route loss due to the reduction in transmission range, the maximum neighbors are used as the last resort if no route is found in the effective neighbors. Therefore, the packet receiving rate is no worse than in *Span*.

If *Span-FNC* is used as the transmission range adjusting algorithm, the route failure in the effective neighbors should be rare, given a well-connected original network, because the coordinators are always given a higher priority in routing and *Span-FNC* effective neighbors include all maximum neighbor coordinators. From the simulation, over 95% of packet forwarding is done by coordinators.

In *Span-NSP*, if routing fails in the effective neighbors and a coordinator c is found as the next-hop node in the maximum neighbors, there is a better way to improve routing by replacing c with a coordinator in the effective neighbors c' (if c' exists): Go upstream from c in the NSP tree of the node until an ancestor c' in the effective neighbors is found. With the construction method of the NSP tree, there must be a cheaper route through c' to c than that of using c directly.

3.2.5 Complexity Analysis

Communication

In the two-level topology control strategy, the communication cost lies only in the local information collection stage (Section 3.2.1). The actual cost is determined by the broadcast rate and the size of *Hello* messages. The broadcast rate is decided by the designer according to the application needs. Generally, better quality of service (QoS) is achieved with a higher broadcast rate at the price of a higher communication cost. A *Hello* message is transmitted as the payload of a MAC frame, and the total number of bits of all *Hello* messages transmitted by a node is $O(n \cdot m)$, where n is the maximum number of neighbors of a node, and m is the number of bits to hold the unique node ID. A *Hello* message uses 32 bits for the position of a node, 2 bits for the state, $m = 10$ bits for the unique node ID and each neighbor node's ID. This setting works for networks of over one thousand nodes. If the maximum number of neighbors of a node is $n = 16$ (which is very large), the maximum size of a hello message will be about 26 bytes.

Space

Each node uses its local memory to store two-hop neighbor information from the latest *Hello* messages. The total space needed is $O(n^2 \cdot m)$. Using the previous example, each node requires at most 416 bytes for that purpose.

Time

Only light computation is involved in the two-level topology control strategy. Almost all computation during the active subnetwork formation (Section 3.2.2) is covered by Equation (3.2). The computation during transmission range adjustment (Section 3.2.3) depends on which algorithm is used. If *FNC* is used, the computation will

be trivial. If *NSP* is used, a node will take time to compute an SSSP tree from the current neighbor coordinators. However, the rate of the re-computing is not high, and the time to obtain a new range is not critical in the algorithm.

The communication, space, and time complexities of the two-level topology control strategy hardly impose much burden on today's wireless sensor hardware. For example, current MICAz MPR2400 Mote is equipped with an Atmel ATMega128L microcontroller (which has up to 16 MIPS throughput at 16 MHz and 128K bytes in-system programming flash memory [3]) and 2.4 GHz RF transceiver with a data rate of 250 kbps [18].

3.3 Simulation

This section describes the energy model, configurations, and results of simulation. The implementation, including the implementation of *Span-FNC* and *Span-NSP*, is based on *Span* and runs in the *ns*2 network simulator.

3.3.1 Energy models

Different assumptions about energy models affect the simulation results significantly. Table 3.1 gives the energy model in *Span* where a fixed transmission range is used in the whole network.

Table 3.1: *Span* Energy Model

Tx	Rx	Idle	Sleeping
1400mW	1000mW	830mW	130mW

In the simulation, an energy model that reflects transmission range changes is required. This chapter considers the low-energy radio (which is typical for wireless sensor networks) and uses an energy model similar to the one used in [29] to compute

the transmission power. The idle and sleeping powers are estimated proportionally from the receiving power, and the proportions are the same with the ones in Table 3.1. Table 3.2 is such an energy model, in which the transmission power is adjustable according to the transmission range needed, r is the distance over which the packet is transmitted, and α is the path loss exponent in the radio propagation model.

Table 3.2: Tx Power Adjustable Energy Model

Tx	Rx	Idle	Sleeping
$(100 + 0.2r^{\alpha})mW$	$100mW$	$83mW$	$13mW$

3.3.2 Simulation environment and parameters

For convenient comparison with *Span*, network configurations similar to [11] are used.

Nodes are uniformly deployed in a rectangular area. Two groups of source/sink nodes (in the same number) locate in the narrow columns at both sides of the rectangular area. The nodes in different groups are paired to send and receive packets. Some of the nodes are elected as coordinators to work in the active mode, and they have a higher priority in terms of forwarding data packets. Non-coordinator nodes work in the power-saving mode and are synchronized to wake up for a while periodically.

Data packets are constant bit rate (CBR) flows: A sender sends packets of a fixed size at a fixed rate. Simulations are executed in networks with different node densities. Node density is defined as the average number of maximum neighbors of a node (source and sink nodes are excluded). For example, a network over a $150m \times 150m$ area consisting of 100 nodes with $r_{max} = 30m$ has the node density of 12.56 nodes per neighborhood.

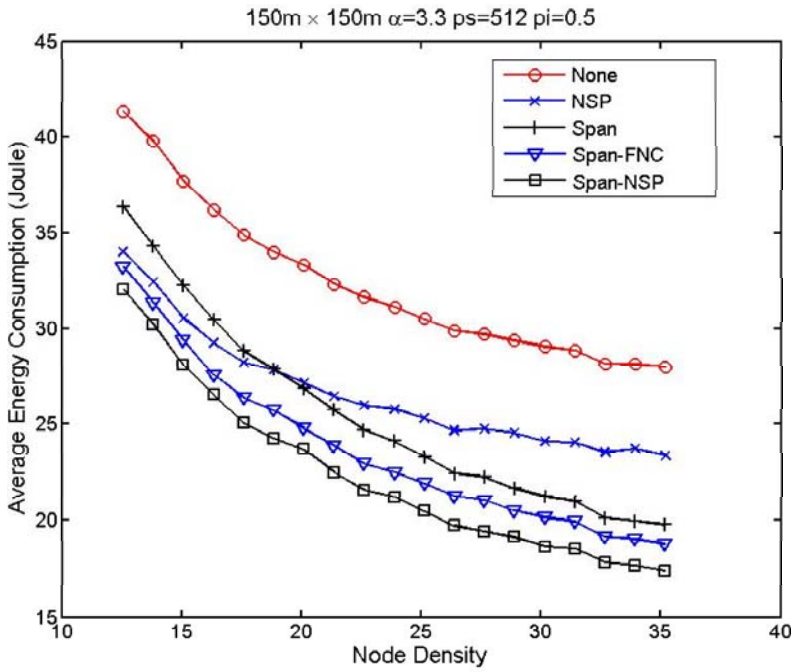


Figure 3.4: The two-level topology control strategy (*Span-FNC* and *Span-NSP*) achieves further energy saving ($\alpha = 3.3$).

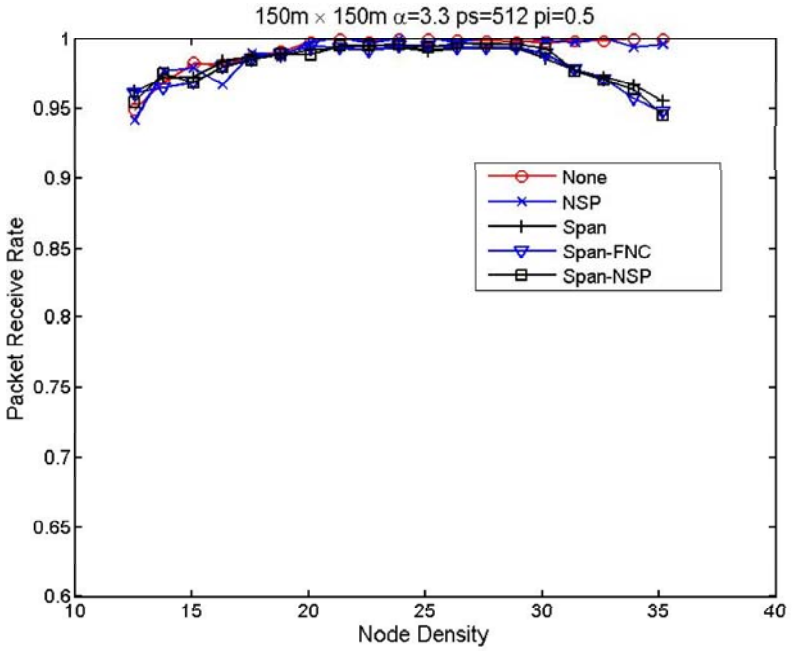


Figure 3.5: The packet delivery rates of all algorithms are comparable when the node density is low, and it is sacrificed in algorithms using the *duty-cycle-based* approach when the node density is high ($\alpha = 3.3$).

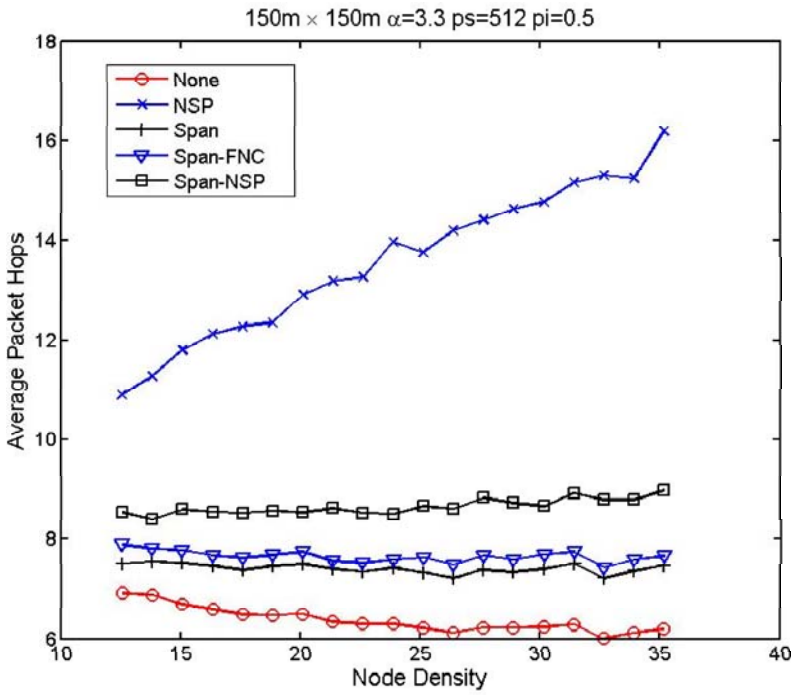


Figure 3.6: As the node density increases, the average packet hop number of *Span-NSP* has a moderate increase, and that of *Span-FNC* is almost unchanged. ($\alpha = 3.3$)

3.3.3 Simulation results

Figures 3.4, 3.5, and 3.6 are the simulation results when $\alpha = 3.3$ in the energy model (Table 3.2). The packet size is 512 bytes (ps=512), and the packet interval is 0.5 second (pi=0.5). *None* is for the network without using any topology control mechanism. *Span* and *NSP* each use one topology control approach. *Span-FNC* and *Span-NSP* are for the two-level topology control strategy.

Figure 3.4 shows that *NSP* can save more energy than *Span* when the node density is less than 20, while *Span* is better when the node density is greater than 20. *Span-FNC* and *Span-NSP* always have better performance than both *Span* and *NSP* in terms of energy saving. For example, *Span-NSP* saves 6% to 30% energy over *NSP* and saves 13% to 15% energy over *Span*. *Span-NSP* always saves about 2% more energy compared with *Span-FNC*. Energy is mainly consumed by the packet delivery and idle nodes in wireless sensor networks. *NSP* reduces the energy consumption of

packet delivery, which dominates the energy consumption in networks with relatively lower node densities. *Span* reduces the average idle time (by increasing the sleeping time), which dominates the energy consumption in networks with relatively higher node densities. The two-level topology control strategy (*Span-FNC* and *Span-NSP*) exploits both techniques and achieves further energy saving for a wide range of node densities.

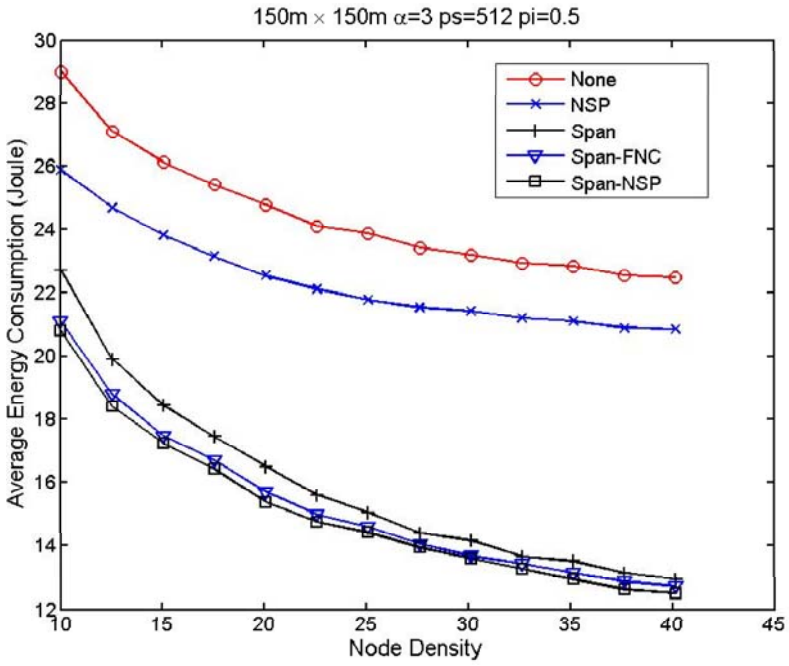


Figure 3.7: The *duty-cycle-based* algorithms are preferable when the path loss exponent is $\alpha = 3$.

Figure 3.5 shows that the packet delivery rates of all algorithms are comparable when the node density is below 30. When the node density is over 30, however, the packet delivery rates are sacrificed in algorithms (*Span*, *Span-FNC*, and *Span-NSP*) that use the *duty-cycle-based* approach. A *duty-cycle-based* algorithm puts some nodes into the power-saving mode, which may cause congestion in the coordinators, hence dropping packets. When the traffic is not very heavy and the node density is not very high, however, the receiving rate can be maintained at some reasonable level. In Figure 3.5, the receiving rate is over 95% with node density below 35. Wireless

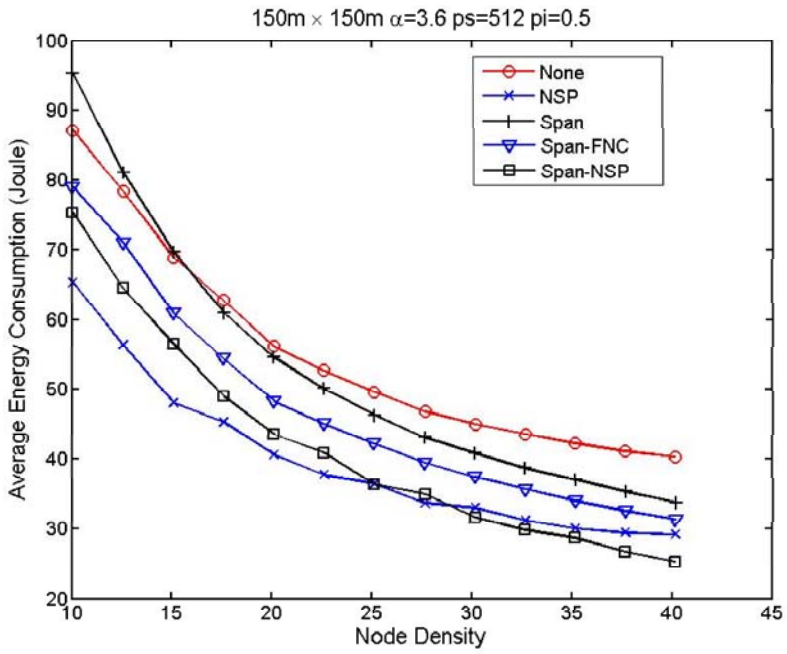


Figure 3.8: The *transmission-power-based* algorithms are preferable when the path loss exponent is $\alpha = 3.6$ and the node density is small.

channel contention is another factor that may cause the delivery rate to decrease, but in a network where traffic is not very heavy, it is not a factor as important as the average transmission range or the number of active neighbors.

Figure 3.6 shows that the average packet hop number increases in *NSP* and *Span-NSP* as the node density increases. Compared with *NSP*, *Span-NSP* has a moderate increase due to a relatively small number of scattered coordinators. *Span-FNC* and *Span* maintain almost the same average hop number over a wide range of node densities, because they always try to use the farthest neighbor coordinators. *None* can have smaller hop numbers in denser networks, because it is easier to find a nearer next-hop node to the destination.

As stated, the energy model has a great impact on the simulation results. Figures 3.7 and 3.8 show the energy consumption results when $\alpha = 3$ and $\alpha = 3.6$ in the energy model. The general trend is that *duty-cycle-based* algorithms are preferable to *transmission-power-based* algorithms when the path loss exponent α is small or the

node density is large, while *transmission-power-based* algorithms are preferable when the path loss exponent is large or the node density is small.

3.4 Conclusion

The two-level topology control strategy demonstrates a way to integrate the existing *duty-cycle-based* and *transmission-power-based* topology control approaches in wireless sensor networks for further energy saving. The algorithms based on the *duty-cycle-based* and *transmission-power-based* approaches are adapted to fit into the two-level approach and dynamically affect each other. While both the *duty-cycle-based* and the *transmission-power-based* approaches have their preferred network conditions under which one outperforms the other, the simulation result shows that the two-level topology control strategy can achieve better performance than either of them in terms of energy saving in networks with a wide range of node densities.

There are many factors in the wireless sensor networks that may affect the performance of the topology control algorithms. Topology changes in the network, energy remaining in nodes, and effective transmission ranges of nodes are foci of the two-level approach. Future work may also consider the balance of workload and rate of energy consumption for finer tuning between the *duty-cycle-based* and *transmission-power-based* approaches.

The work presented in this chapter has been published [78, 80].

Chapter 4

DSP-CDS: A Distributed Clustering Algorithm

4.1 Introduction

Clustering is an important approach to solving capacity and scalability problems in wireless *ad hoc* networks where no physical infrastructure is available. Connected dominating set (CDS) as a clustering method is widely used in wireless *ad hoc* networks for topology control [48], routing [19] [75], broadcasting [62], etc. For a graph G, a set S of vertices is a dominating set of G if every vertex of G is either in S or adjacent to a member of S. A dominating set S is a connected dominating set if the subgraph of G induced by S is connected. A node in the dominating set is called *dominator*. A small size CDS is desirable in many applications. A CDS of minimum size is called minimum connected dominating set, or MCDS. Although finding an MCDS in a general graph is an NP-complete problem [25], many approximate MCDS algorithms have been proposed.

Distributed CDS construction algorithms in literature usually have two or more phases, as discussed in Section 2.5. An algorithm with multiple separated phases

has two drawbacks. One is the long delay in later phases when the network size is large, because each later phase must wait until the previous phase finishes in the whole network and the convergence time of each phase increases as the network size increases. The other is the poor ability to adapt to the dynamic network topology, which is typical in mobile *ad hoc* networks (MANETs) and wireless sensor networks (WSNs). If the root in a *Type-I* algorithm withdraws from the network or the global structure in a *Type-II* algorithm is damaged due to node movements, the connectivity of the dominating set can not be maintained without recalculations in all phases. When a recalculation happens, the current structure of CDS is not valid any more. Applications depending on the CDS must wait until the recalculation finishes in the entire network.

This chapter presents the DSP-CDS algorithm, a novel distributed single-phase CDS construction algorithm for *ad hoc* networks. The algorithm converges quickly in a single phase and generates a CDS of competitively small size. In DSP-CDS, each node only uses the information of its one-hop neighbors to make a local decision on whether to join the dominating set, and dominators are selected simultaneously across the network. This is like the clustering phase in a distributed *Type-II* algorithm, but DSP-CDS does not need a separate connecting phase. Clusters (called pieces) in DSP-CDS grow until they are connected to each other, and the merged larger pieces continue to grow until a CDS is formed in the network. Each node bases its decision on a key variable, *strength*, which guarantees that the dominating set is connected when the algorithm converges. The rules for computing *strength* can be changed to accommodate different application needs, for example, to include the ability of balancing energy consumption among neighboring nodes. DSP-CDS adapts well to dynamic network topologies caused by node mobility, node failure, and deployment of new nodes. Upon topology changes, the nodes adjust their *strength*s and trigger local updates to the current CDS structure.

4.2 Distributed Single-Phase CDS Construction

4.2.1 Terminology

In the DSP-CDS algorithm, each node in the network has a unique identifier, called *nid*. A node can be in one of the three *state*s: white, gray, and black. A dominator is in the black state. A node adjacent to a dominator, if it is not a dominator itself, is in the gray state. All the other nodes (neither black nor gray) are white. A *piece* is a connected sub-network during the CDS construction. Connected black nodes and their gray neighbors form one piece. A white node itself is a piece. A piece has a unique ID, called *pid*, known to all nodes in the piece. A piece has a special node called *master*. The master of a trivial piece is the white node itself, and the master of a non-trivial piece is one of the black nodes. The piece master decides the *pid* of the piece. The *strength* of a node indicates the ability of the node to connect with different pieces.

4.2.2 Design of DSP-CDS

DSP-CDS is a distributed algorithm. It is designed to achieve three goals. First, it generates a CDS fast in a single phase. Secondly, it generates the CDS of competitively small size compared with other distributed algorithms. Thirdly, it efficiently maintains the CDS under network topology changes.

The single-phase method

It is desirable to have dominators selected simultaneously across the network so that the CDS can be generated faster. The question is how do we guarantee that the dominators are connected without resorting to a root election phase as in *Type-I* algorithms or a connecting phase as in *Type-II* algorithms and that the constructed CDS has a reasonably small size. In DSP-CDS, the answer lies in *strength*, a concept

introduced to elect dominators.

The definition of *strength* is the key in DSP-CDS. It must satisfy three criteria:

1. A node with a greater *strength* indicates a better choice as a dominator;

2. The *strength* of a node decreases and reaches 0 at some point during the execution of the algorithm; and

3. The dominating set is connected when all nodes reach the *strength* of 0.

Various factors can be incorporated into the definition of *strength*. For example, an application may consider communication capability or energy so that the selected dominators have higher throughput or more remaining energy.

DSP-CDS is a single-phase method. It starts in a state where all nodes in the network are white, and the number of pieces is equal to the number of nodes. The algorithm progresses as the nodes with greater *strength* values become black and the number of pieces in the network decreases. Given a well-defined *strength* and a connected original network, the algorithm converges to a state where all the black nodes in the network form a CDS and all nodes reach the *strength* of 0. If the network is not connected, the algorithm forms a CDS for every connected component in the network.

Dynamic topology consideration

Dynamic topology is one of the salient characteristics of mobile *ad hoc* networks where nodes are free to move. The network topology may also change unpredictably due to node failure, running out of power, or adding new nodes into the network. Dynamic topology has a significant impact on CDS algorithms.

Two actions of a node lead to the network topology change: withdrawing and joining. Withdrawing refers to the functional termination of a node in the network,

and it happens when a node fails, runs out of power, or exits from the network. Joining refers to the functional start of a node in the network, and it happens when a new node is added or a node recovers from a failure. Moving of a node can be treated as two separate actions of withdrawing and joining, if the node can be assumed to stop receiving and transmitting messages when in motion. To cover a broader range of situations, this paper assumes no special notification sent from the withdrawing or joining node. Relying on such notification, even if possible, imposes high expectation on the ability of nodes. The neighbors of a changing node must rely on other mechanisms to detect the changes. In DSP-CDS, periodically broadcast messages are used for this purpose (See Section 4.2.3 for details).

The changing neighborhood resulted from a node withdrawing or joining affects the generated CDS. Generally, there are two methods to handle it: recalculating and updating. With the recalculating method, a distributed CDS algorithm starts at a fixed interval or is triggered by some event (e.g. when disconnection of the dominating set is detected), and a new CDS is generated from scratch. With the updating method, the CDS is maintained by updating a portion of the existing dominating set according to the topology changes. A practical strategy may use the updating method most of the time and use the recalculating method when necessary. This chapter only discusses the updating method.

4.2.3 Implementation of DSP-CDS

In DSP-CDS, the execution of each node is divided into rounds, as illustrated in Figure 4.1. In each round, a node exchanges status messages with its neighbors, updates its *strength*, and decides if it should become a dominator. If the new *strength* is 0, the node will not change its state (white, gray, or black). If the new *strength* is greater than 0, the node decides its new state based on its current status (including the value of *strength*) and the local knowledge about its neighbors.

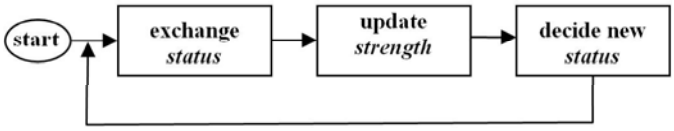


Figure 4.1: A node using DSP-CDS works in rounds.

PseudoCode–1 gives the computation of strength, and PseudoCode–2 gives the implementation of the DSP-CDS algorithm.

Strength

In the DSP-CDS implementation, the *strength* of a node i is computed using the following rules:

Rule-1: If node i is black, its *strength* is 0;

Rule-2: If node i is gray or white, its *strength* is the sum of the points contributed by its neighbors;

Rule-3: A black or white neighbor contributes 2 points, and a gray neighbor contributes 1 point, with the following exceptions:

. A neighbor with the same *pid* as node i contributes 0 point to the strength of node i;

. Among the neighbors sharing the same *pid* only the one that can contribute the greatest points contributes to the strength of node i.

This definition of strength considers only the topology information. A node a has greater strength if it can reduce more pieces in the network after becoming black. A black node has the zero strength and does not make state decision. A gray neighbor contributes fewer points than a black or white neighbor, because connecting to only gray nodes in another piece will not combine the two pieces into one. Applications

are free to make changes to this definition with different point distribution among neighbors, or with consideration of other factors such as energy and communication capabilities, as long as the three criteria in Section 4.2.2 are satisfied.

PseudoCode. 1 Strength computation in DSP-CDS

```
// Compute the strength of a node based on local knowledge about neighbors
compute_strength {
    if (no neighbor)
        return InvalidStrength; // Strength for an isolated node

    pidSet = {pid};
    new_strength = 0;

    // Compute the strength using information of neighbors if the node is
    // not black
    if (state != black) {
        // Count points of black or white nodes in the first round
        for each black or white neighbor node i {
            if i.pid is not in pidSet {
                pidSet = pidSet + {i.pid};
                new_strength += 2;
            }
        }
        // count points of gray nodes in the second round
        for each gray neighbor node i {
            if i.pid is not in pidSet {
                pidSet = pidSet + {i.pid};
```

```
                    new_strength += 1;
                }
            }
        }
        return new_strength;
    }
```

An isolated node has no neighbors, so it has the invalid strength and will never change its state (PseudoCode–1). The invalid strength is also used as the initial strength value of a node (PseudoCode–2), when no valid strength is ever computed for it.

State decision

In DSP-CDS, each node decides when to do the work in each round. Two time intervals, T_1 and T_2, are introduced to control the frequency of **STATUS** broadcast and the frequency of state decision, respectively. T_1 and T_2 are parameters of the algorithm, and they should satisfy the following constraint in order for the algorithm to make proper decisions:

$$T_2 \geq 2T_1 + 2D_{max}, \tag{4.1}$$

where D_{max} is the estimated maximum single-hop delay in the network.

Here, a proper decision of node i means that a new state decision in node i should base on the feedback from its neighbors who have already updated their status according to the previous state decision made in node i. This is illustrated in Figure 4.2.

The basic process of state update is as follows. A node in the network sends a **STATUS** message to its neighbors at the interval of T_1. A **STATUS** message

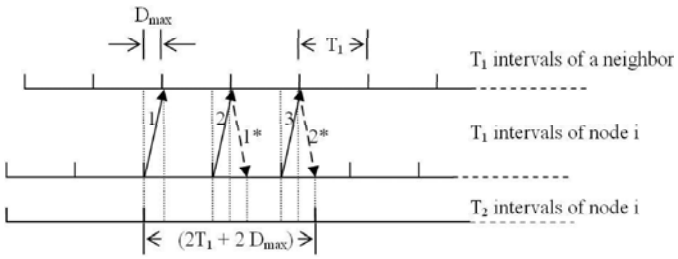


Figure 4.2: DSP-CDS uses the constraint $T_2 \geq 2T_1 + 2D_{max}$ in order to make proper decisions.

contains the current information about the sender: *nid*, *pid*, *state*, and *strength*. A node with a nonzero strength makes a decision on whether to become black at the interval of T_2. If its strength is the greatest among its neighbors, the node becomes black. The *pid* and *nid* values are used for tie breaking. A node with zero strength does nothing at this time. Upon receiving a **STATUS** message, a node updates its local knowledge about the sender. If the sender is not black, nothing additional happens in the receiver. If the sender is black, a black or gray node receiving a message with a greater *pid* will update its *pid*; while a white receiver will become gray and update its *pid* to the one in the message. See PseudoCode–2 for details.

PseudoCode. 2 Pseudo code of each node program

```
// Define a constant
define InvalidStrength = −1;
// Initialize local variables
pid = nid; // Initially, each node has a unique nid
state = white;
strength = InvalidStrength; // Initial strength value
parent = nil;

// T1 and T2 are input variables of the algorithm.
```

```
10 reset timer t1 = T1;
   reset timer t2 = T2;
   retst timer beaconTimer = BeaconExpiration;
   reset timer pidTimer = PidExpiration;

15 upon expiration of t1 {
       strength = compute_strength(); // Compute a new strength
       broadcast STATUS(nid, pid, state, strength);
       reset timer t1 = T1;
   }
20
   upon expiration of t2 {
       if (state == black || strength == 0) return;
       strength = compute_strength(); // Compute a new strength
       // Get the maximum strength among neighbors
25     (s, p, n) = get_max_neighbor();
       if ((strength, pid, nid) > (s, p, n))  state = black;
       reset timer t2 = T2;
   }

30 upon expiration of beaconTimer {
       if (state == black and pid == nid) {
           broadcast BEACON(nid, pid);
       }
       retst timer beaconTimer = BeaconExpiration;
35 }
```

```
upon expiration of pidTimer {
    if ((pid != nid) and (state == gray or state == black)) {
        pid = nid;
        parent = nil;
        reset timer t1 = T1;
        reset timer t2 = T2;
        if (state == gray) {
            state = white;
            strength = InvalidStrength;
        }
    }
}

upon expiration of neighborTimer(nid) {
    delete neighbor(nid)
}

upon receving a STATUS message S {
    save neighbor(S.nid) = (S.pid, S.state, S.strength);
    reset timer neighborTimer(sener.nid) = NeighborExp;
    if ((S.state == black) and (S.pid > pid || state == white)) {
        pid = S.pid;
        parent = S.nid;
        reset timer pidTimer = PidExp;
        if (state == white) state = gray;
    }
}
```

```
65 upon receving a BEACON message {
       // only consider the BEACON from the parent
       if (parent == BEACON.nid) {
           reset timer pidTimer = PidExpiration;
           if (state == black) {
70             // A black node needs to relay the beacon
               broadcast BEACON(ni, pid);
           }
       }
   }
75
   // Get the maximum (strength, pid, nid) amongst neighbors using
   // local information
   get_max_neighbor {
       (s, pid, nid) = (0, 0, 0);
80     for each neighbor node i {
         if (i.strength, i.pid, i.nid) > (s, pid, nid) {
           (s, pid, nid) = (i.strength, i.pid, i.nid);
         }
       }
85     return (s, pid, nid);
   }
```

Note: $(A_1, A_2, A_3) > (a_1, a_2, a_3)$ is true when

$A_k > a_k (k \in [1, 3])$ and $A_i = a_i (i = 1..k-1)$

Handling node mobility

STATUS messages are also used to deal with network topology changes. A node broadcasts **STATUS** messages at fixed interval (determined by T_1) to indicate its aliveness in addition to reporting its status. A node maintains in its local storage the information about its known neighbors and updates it with each received **STATUS** message. If a node does not receive a **STATUS** message from a neighbor for a period of time (defined by the timer *neighborTimer*), it regards the neighbor as a withdrawn node and deletes the associated entry from its local storage. If a node receives a **STATUS** message from an unknown node, it will allocate a new neighbor entry for it.

BEACON messages are utilized to maintain the connectivity of the dominating set, if still possible, in case that some dominators withdraw. **BEACON** messages are generated in the master dominator of each piece at a fixed interval (indicated by *beaconTimer*) and propagated to all nodes in the piece. If a gray node does not receive a **BEACON** message for a period of time (indicated by *pidTimer*), it will become a white node (a new piece) and clear its strength and *pid* accordingly. If a non-master black node does not receive a **BEACON** message for a period of time (also indicated by *pidTimer*), it will withdraw from the piece and start a new piece with itself as the master dominator.

A node remembers the black node as its *parent* from which it gets the *pid*, and it will later only accept **BEACON** messages from the parent to avoid excessive broadcast of **BEACON** messages among black nodes. Therefore, in any piece the connected dominators form a rooted tree with the piece master as the root, non-master black nodes as intermediate nodes, and gray nodes as leaves. The **BEACON** messages flood from root to leaves through the tree in normal situations.

Two additional constraints apply on the timers:

$$neighborTimer > T_1 \tag{4.2}$$

$$pidTimer > beaconTimer. \tag{4.3}$$

Table 4.1: Summary of Parameters in DSP-CDS

Parameter	Meaning
T_1	frequency of **STATUS** messages
T_2	frequency of the state decisions
neighborTimer	expiration threshold of the neighbor status
beaconTimer	frequency of **BEACON** messages
PidTimer	expiration threshold of a connected piece

A withdrawing or joining node causes the strength and/or pid changes in its neighbors, which in turn triggers new state decisions in those neighbors to adapt to the changed topology. The departure of a white node or a gray node does not have any effect on the existing dominating set. A neighbor node will simply delete its entry for the withdrawn node and adjust its strength accordingly. The departure of a black node may leave some gray nodes unattended. If that happens, those gray nodes will withdraw from the current piece and become white nodes. The departure of a black node may also leave other black nodes in the same piece disconnected. If that happens, those black nodes will form several new pieces. In addition to the above routines, the departure of a piece master will always cause one or more new masters to be elected. Adding a new node into the network is relatively easier to handle, and the new node will join a current CDS as a gray node or cause election of new dominators.

Node withdrawing is less frequent to happen on black nodes because of the small percentage of them in the network. Therefore, a black node does not go back white

and start all over again, as a gray node does, in order to preserve as much as possible the existing CDS structure and achieve a quicker recovery from the changed topology.

Setting of parameters

DSP-CDS has five parameters summarized in Table 4.1. Setting of these timers are left to the application, as long as the constraints of inequalities 4.1, 4.2, and 4.3 are satisfied.

The *beaconTimer* should reflect the node withdrawing frequency in a network. The more frequent the withdrawing happens, the smaller *beaconTimer* should be. To compensate for the loss of **BEACON** messages in an unreliable network, the constraint of inequality 4.3 should be stricter, and *pidTimer* should be set to several times larger than *beaconTimer*. Inequality 4.2 may be changed similarly. In this case, the chance of message loss is much smaller, and a **STATUS** message is only sent over one-hop distance.

The equality in constraint of inequality 4.1 assumes a reliable network. A longer T_2 than merely $T_2 = 2T_1 + 2D_{max}$ can compensate the loss of **STATUS** messages due to radio interference. In some cases, the state decision is made based on out-of-date information, but the gap is small and does not hurt the performance.

Other implementation issues

In the above discussion and the pseudo code, **STATUS** and **BEACON** messages are stated as two separate messages for the presentation purpose. In the real implementation, however, information in a **BEACON** message can be combined into a **STATUS** message, because the value of *beaconTimer* is usually several times of the value of T_1. If DSP-CDS sits in the MAC layer, the **STATUS** message itself can be in turn combined into the existing heartbeat message in the network, e.g. the beacon frame in IEEE 802.11x networks. If DSP-CDS is used in the application layer, the

application needs to maintain all the messages.

In a static topology network, there is no need to broadcast **STATUS** messages at a fixed interval. A great amount of messages can be saved by letting a node only broadcast the **STATUS** message when there is some change in its status (strength, pid, or state). See Section 4.3.3 for the message saving simulations in the DSP-CDS algorithm for the static topology networks.

4.2.4 Correctness of DSP-CDS

The proof assumes a reliable symmetric network. That is, there is no message loss and all communication channels are bidirectional. The proof is only for static topology networks, and the situation in dynamic networks will be explained later. The algorithm starts with the **initial state** where all nodes are white, have the strength of 0, and have the *pid*s equal to their *nid*s.

Theorem 4.2.1 *The DSP-CDS algorithm reaches a* ***stable state****, where all nodes have the strength of* 0 *and no node will change the state any more.*

Proof Initially, every node is white and has an invalid strength. After some neighbors are discovered and before any of them changes its state, the strength of a node will be a positive value, two times the number of its discovered neighbors, because all the neighbors are white and have unique *pid*s. Assuming there are m ($m > 0$) nodes that have positive strength values at time t, we know there are at most $m - 1$ nodes that have positive strength values at time $t + 2T_2$, since at least one of the m nodes will become black and reach the strength of 0 in the next $2T_2$ period, and the strength of a node stops changing when it drops to 0 in a static topology network. Therefore, all nodes have the strength of 0 no later than $t + 2mT_2$, and no nodes change their states after that.

Lemma 4.2.2 *If the original network is connected, there is no white node in the stable state.*

Proof (Proof by Contradiction) Assume there exists a white node in the stable state. The white node's *pid* must be equal to its *nid*, because it never receives a **STATUS** message from a black neighbor and never changes its *pid*. Since the *pid* of the white node is unique in the network and the original network is connected, the white node must have at least one neighbor with a different *pid*. Therefore, the white node has a positive strength value, which is in contradiction to the definition of stable state.

Lemma 4.2.3 *If the original network is connected, the black nodes form a connected subgraph in the stable state.*

Proof (Proof by Contradiction) Assume the algorithm forms more than one piece in the stable state. According to the algorithm, the *pid* does not propagate over the border of a piece, so any two nodes that are not in the same piece do not have the same *pid*. Since the network is connected, there must exist at least one node that has a neighbor with a different *pid*. That node will have a positive *strength*, which is in contradiction to the definition of stable state.

Lemma 4.2.4 *A gray node has at least one black neighbor.*

Proof A white node becomes gray if and only if it receives a **STATUS** message from a black neighbor, and black neighbors stop changing its state after becoming black in a static topology network. Therefore, a gray node has at least one black neighbor.

Theorem 4.2.5 *If the original network is connected, the black nodes form a CDS of the network in the stable state.*

Proof In the stable state, a node is either black or adjacent to a black node according to Lemma 4.2.2 and Lemma 4.2.4, and the black nodes form a connected subgraph

according to Lemma 4.2.3. Therefore, the black nodes form a CDS of the network in the stable state.

Theorem 4.2.6 *The DSP-CDS algorithm reaches a* ***final state*** *where all nodes have the same* pid *in a connected network.*

Proof After reaching the stable state, the state of all nodes will not change any more in a static topology network, but the *pid*s may still propagate through the CDS. If the size of the CDS is d, it takes at most $(d+1)T_1$ time for the final *pid* to reach all the nodes in the network, and all the nodes will have the same *pid*.

With a dynamic network topology, some nodes may withdraw from or join into the network during or after the CDS construction. Node mobility may disturb the structure of CDS, but the neighbors of those withdrawing or joining nodes will update their strength to reflect the changes. Therefore, if the network keeps static for a limited period of time after a change, the DSP-CDS algorithm will converge to the final state and generate a valid CDS as in a static topology network.

4.2.5 Execution process of DSP-CDS

Figure 4.3 through Figure 4.8 illustrate the execution process of DSP-CDS on a medium size network. The network size is 70×70. There is a total of 49 nodes arranged into a 7×7 grid with small random perturbation of grid points. Nodes IDs are randomly assigned, and each node has the same radio range $R = 18$ so that its neighbors are only on the nearby layer. This layout makes the network easy to read but have sufficient complexity. Each figure is a network snapshot taken at the interval of $T2$ and roughly shows the result after each round of execution. The numbers to the upper-right corner of each node are values of the node's *nid*, *pid*, and *strength*, respectively.

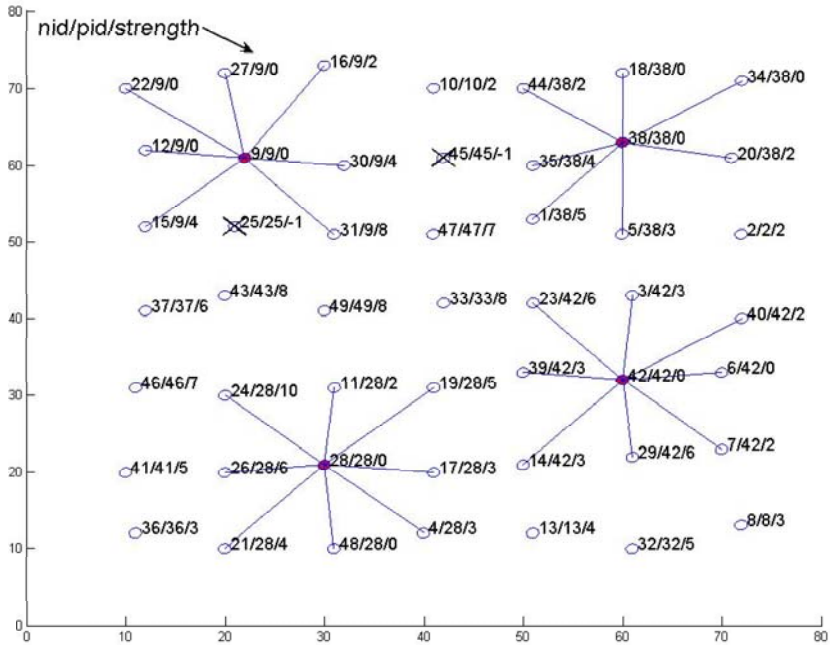


Figure 4.3: Round #1: Nodes $9, 28, 38$ and 42 become dominators, while nodes 25 and 45 are off. The numbers are *nid/pid/strength*.

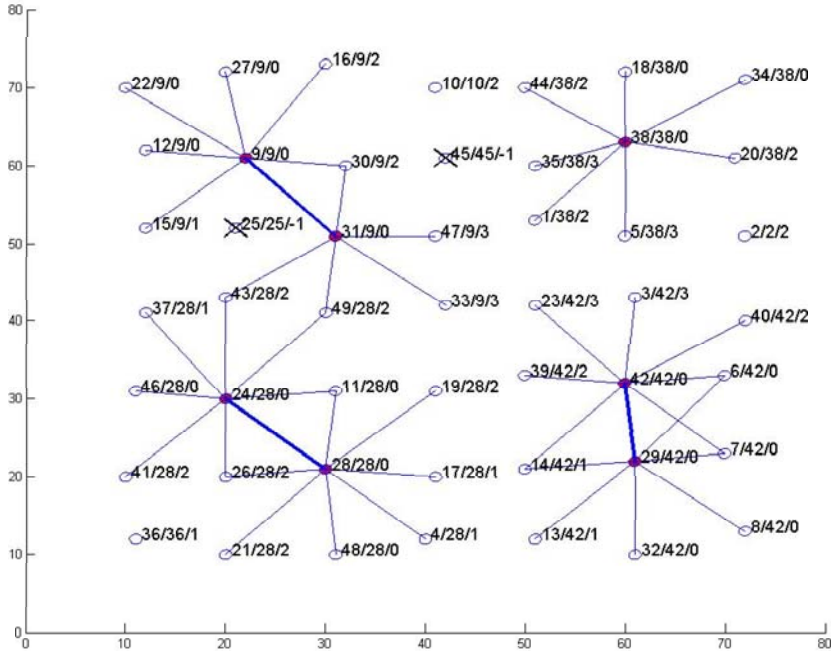


Figure 4.4: Round #2: Nodes $31, 24$ and 29 become dominators. The connected dominators are linked by thick lines.

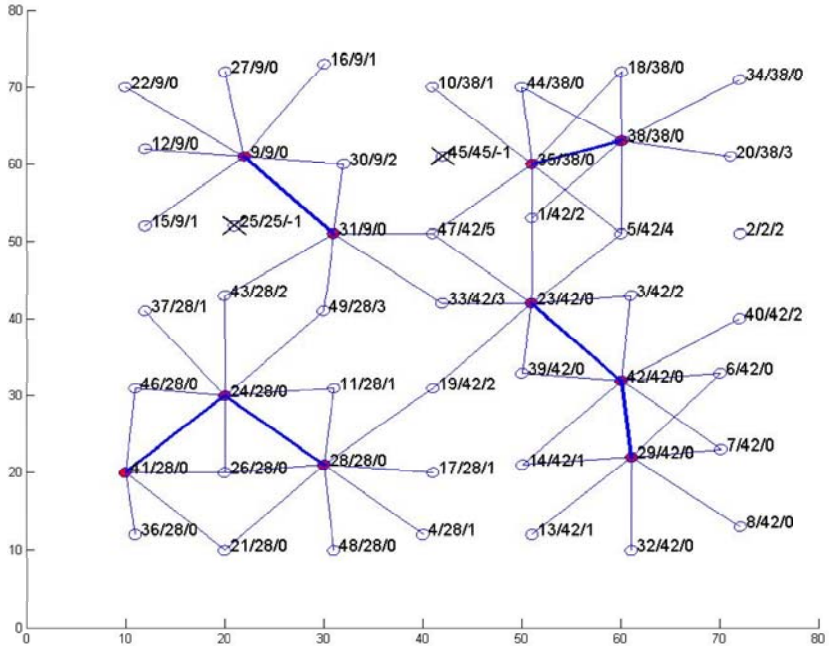


Figure 4.5: Round #3: More dominators are elected, forming 4 larger pieces. All nodes in the same piece share the same *pid*.

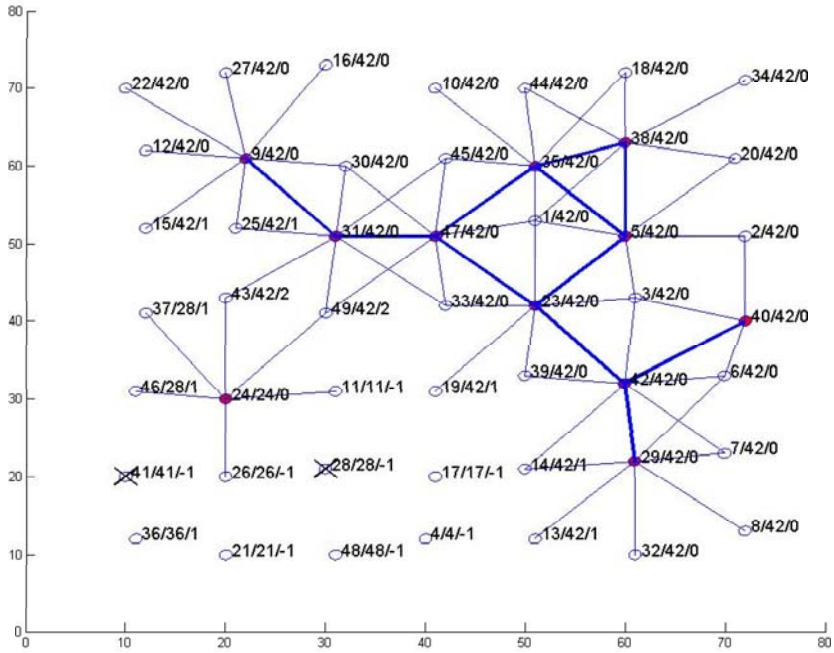


Figure 4.6: Round #4: Nodes 25 and 45 join the network, while nodes 28 and 41 withdraw from the network.

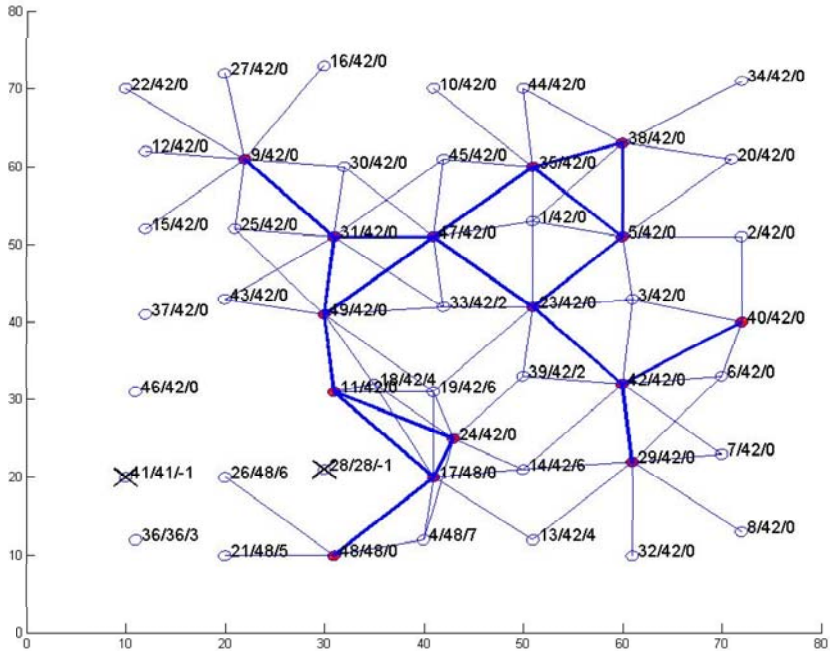


Figure 4.7: Round #5: Nodes 18 and 24 move to new locations. Node 24 was a dominator, and the gray nodes that were dominated only by node 24 become white again.

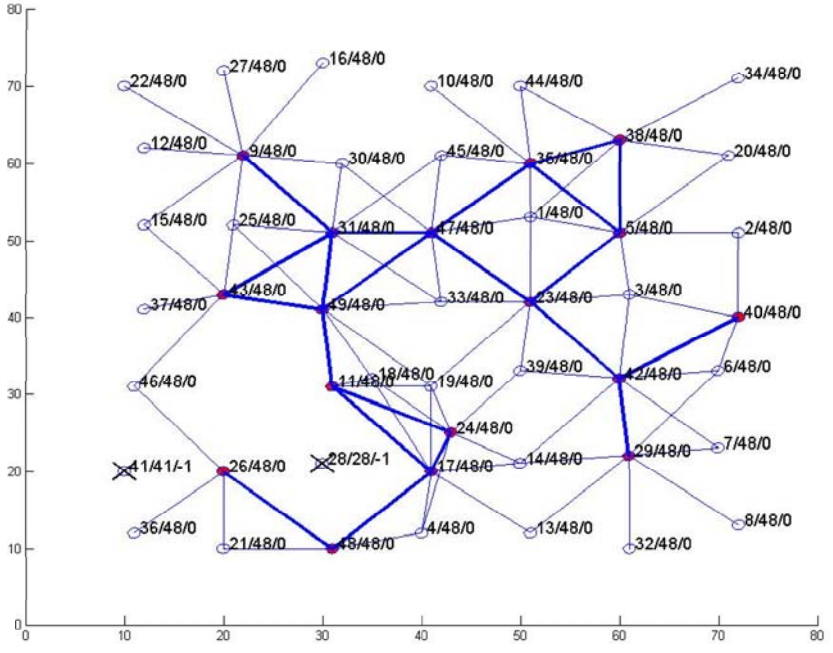


Figure 4.8: Round #7: Final CDS. All nodes are in one piece and share the same *pid*. (Round #6 is omitted.)

Generally, node withdrawing or joining occurring during the process of CDS construction has greater impact on the performance of the algorithm than after it. In these figures, some nodes change their states to simulate such impact. Nodes 25 and 45 are turned off (with a cross on them) at first and turned on between rounds 3 and 4 to simulate joining of nodes. Nodes 28 and 41 are turned off between rounds 3 and 4 to simulate withdrawing of nodes. Nodes 18 and 24 change their locations between rounds 4 and 5 to simulate moving of nodes.

Figure 4.3 shows that after the first round of execution, four nodes become dominators (black nodes). Each dominator forms a piece with itself as the piece master, graying its neighbors. A gray node is shown linked to its dominator by a line. All the nodes in the same piece share the same *pid* with the piece master. Figures 4.4 and 4.5 show that more dominators are elected and some of them are connected to form a larger piece. Connected dominators are shown linked to each other by thick lines. In Figure 4.6, nodes 25 and 45 join the network, and they fit themselves into the existing pieces, while nodes 28 and 41 withdraw from the network, leaving some gray nodes unattended. Thus, those gray nodes become white again with $strength = -1$ and $pid = nid$ (nodes 36, 21, 48, 4, and 17). Node 24 becomes a new piece master after the withdrawing of node 28, but the gray nodes around node 24 have not yet updated their *pid*s. In Figure 4.7, nodes 24 and 18 move to new locations and cause the dominating set to change accordingly. In Figure 4.8, a final CDS is constructed across the network in round 7, and all the nodes are in one piece and share the same *pid* (which is 48 in the figure). It takes DSP-CDS 7 rounds to generate the CDS with all the topology changes. The snapshot of round 6 is omitted.

4.3 Simulation and Discussion

This section has three parts. Section 4.3.1 evaluates the performance of DSP-CDS under different settings of the parameters. Section 4.3.2 discusses the comparison results of DSP-CDS with other two CDS algorithms. Section 4.3.3 how to reduce messages in static topology networks. All algorithms in discussion were implemented using MATLAB.

In simulations, all nodes are randomly deployed in a square area with side length (L) ranging from 40 to 120. Every node uses a fixed radio range $R = 10$ and has a perfect disk coverage. Therefore, the networks are sized 4 to 12 times the node radio range. The network size and node density determine the number of nodes (N) in the network. *Node density* (or *absolute node density*), ρ, is defined as the average number of nodes per unit area. For example, a 100×100 network with node density $\rho = 0.01$ has $N = 100 \times 100 \times 0.01 = 100$ nodes. *Relative node density* is defined as the number of neighbors per node. Because the radio range of each node is fixed to $R = 10$, the absolute node density and relative node density have a one-to-one correspondence. For example, given (absolute) node density $\rho = 0.01$, the relative node density is $\pi \times R^2 \times \rho = 3.14$.

All networks are checked to make sure that they are connected before simulations begin. Special care is taken in mobility simulations to make sure that the networks are still connected after some percentage of nodes withdraws.

In most simulations, evaluation metrics are CDS size, CDS diameter, and number of rounds it takes the algorithm to converge. Section 4.3.3 discusses the number of messages.

Table 4.2 summarizes all the network configurations used in simulations. The *Dynamic* column lists the type of dynamic topology simulated, if available. The $'-'$ item means that the parameter of this column is under evaluation in this configuration and different values for that parameter will be set during simulations. The estimated

maximum single-hop delay, D_{max}, is set to zero in all the simulations. In all figures of the simulation results, every data point is the average of 20 simulations under the same configuration.

Table 4.2: Summary of Network Configurations

Config	Network Length (L)	Node Density (ρ)	Number of Nodes (N)	Radio Range (R)	T_2/T_1	Dynamic
1	40-120	0.04	64-576	10	–	N.A.
2	40-120	–	-	10	4	N.A.
3	40-120	0.04	64-576	10	4	Withdrawing
4	40-120	0.04	64-576	10	4	Joining
5	40-120	0.04	64-576	10	4	Moving
6	40-120	0.04	64-576	10	4	N.A.
7	40-120	0.04	64-576	10	4	N.A.

4.3.1 Performance evaluation of DSP-CDS

Frequency of state decision

T_1 and T_2 are the two most important parameters that affect the performance of DSP-CDS. The *pid* of a piece propagates through **STATUS** messages at the speed of one hop per T_1 time. In a large piece, it takes multiple hops for a new *pid* to reach all the nodes. Therefore, it is possible for a node to view one piece as two when the node makes a state decision. If that happens, it means the diameter of the piece is large. A dominator that is selected by viewing one large piece as two will reduce the diameter of the CDS. Generally, a larger T_2 (relatively to T_1) leads to a smaller size CDS and longer convergence time. The redundant dominators with a smaller T_2 are caused by the longer delays of *pid* propagation in pieces of large diameters. Therefore, in addition to generating a CDS faster, a smaller T_2 can reduce the diameter of the resultant CDS at the price of a larger CDS.

Figure 4.9 and Figure 4.11 show that a larger T_2 leads to a smaller CDS size but

a larger CDS diameter most of the time. In larger networks ($L > 85m$), the trend for CDS size changes is more pronounced. In Figure 4.13, the algorithm with a larger T_2 takes fewer rounds to converge, the actual convergence time is larger because the actual time of each round depends on T_2.

Node density

Node density determines how many neighbors a node can have, since the radio range of a node is fixed. With a higher node density, a node has more neighbors to compete with to become a dominator. But after a node becomes a dominator, all its white neighbors are covered as gray nodes and share the same *pid*. Usually, a node that can cover more white nodes has a greater chance to become a dominator because of its greater strength. Thus, a new dominator will try to cover a new area of the network, given a connected network and a fixed radio range. Therefore, if the algorithm is well designed, the CDS size should be mainly determined by the network size and has less to do with the node density.

Figure 4.10 and Figure 4.12 show that DSP-CDS generates CDSs of almost the same size and the same diameter in networks with various node densities. But it takes longer time for the algorithm to converge in high density networks (Figure 4.14).

Node mobility

In the node withdrawing simulations (Figures 4.15, 4.17, and 4.19), a small percentage of nodes withdraws from the network randomly during the process of CDS construction. As stated earlier, node withdrawing after the CDS construction should have less impact on the resultant CDS. A network with higher withdrawing percentages needs more rounds to converge, as shown in Figure 4.19. But the DSP-CDS algorithm deals with node withdrawing well enough so that the CDS size and the CDS diameter do not change too much with up to 10% withdrawing percentages, as shown in Figure

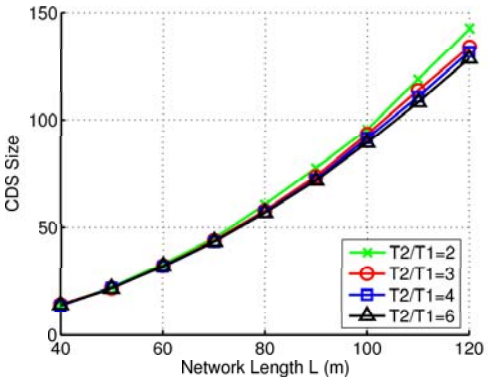


Figure 4.9: CDS size of Config-1: A larger T_2 leads to a smaller CDS size.

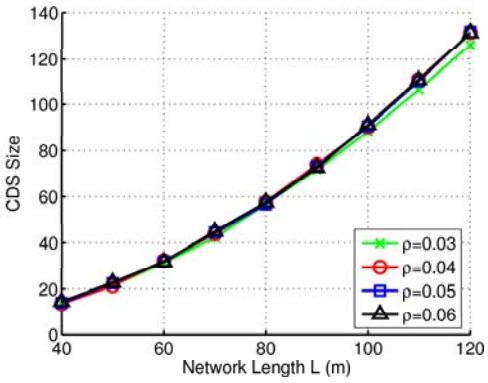


Figure 4.10: CDS size of Config-2: Node density has little impact on the CDS size.

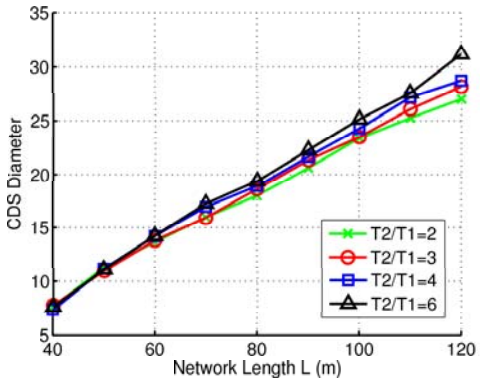


Figure 4.11: CDS diameter of Config-1: A larger T_2 leads to a larger CDS diameter.

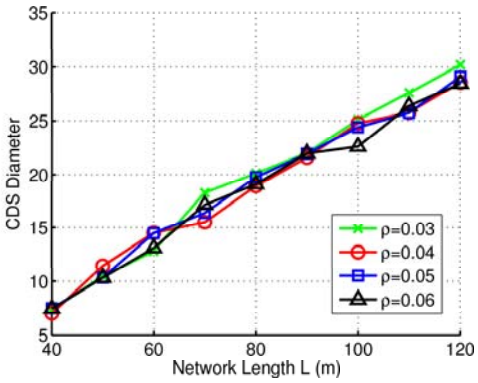


Figure 4.12: CDS diameter of Config-2: Node density has little impact on the CDS diameter.

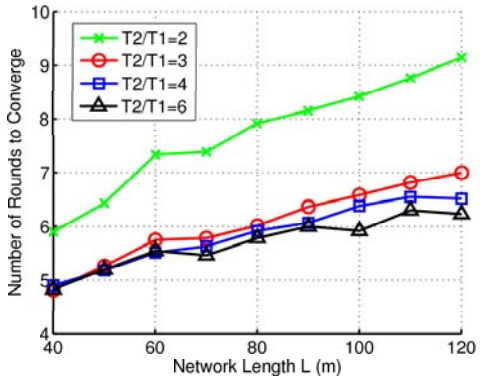


Figure 4.13: Number of rounds of Config-1: A larger T_2 takes fewer rounds to converge.

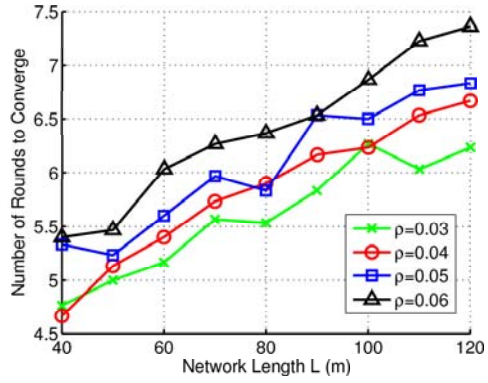


Figure 4.14: Number of rounds of Config-2: Higher node density leads to a little longer convergence time.

4.15 and Figure 4.17, respectively.

Joining nodes have very limited impact on the existing CDS. Most of them fit into the existing CDS as gray nodes. Although a small fraction may cause new dominators to be elected, they have no impact on the existing dominators. Therefore, a higher percentage of joining nodes during CDS construction has little impact on the CDS size and CDs diameter, as shown in Figure 4.16 and Figure 4.18. Joining nodes have less impact on the convergence time than withdrawing nodes as shown in Figure 4.20

Node movements were treated as two actions of withdrawing and joining. As the combined effect of those two actions, the impact of node movements during the CDS construction is very similar to that of the node withdrawing, as shown in Figures 4.21, 4.23, and 4.25.

In all the simulations, the timers related to mobility are set as follows:

$$NeighborExp = 3T_1$$

$$BeaconExp = 2T_1$$

$$PidExp = 3BeaconExp = 6T_1$$

Table 4.3: Message Saving in Static Topology Networks

Network Length	40	50	60	70	80	90	110
Msg# of DSP-CDS	570	1102	1841	2633	3710	5046	7952
Msg# of static DSP-CDS	74	137	233	342	470	639	1043
Saving %	87.01	87.56	87.34	87.01	87.33	87.33	86.88

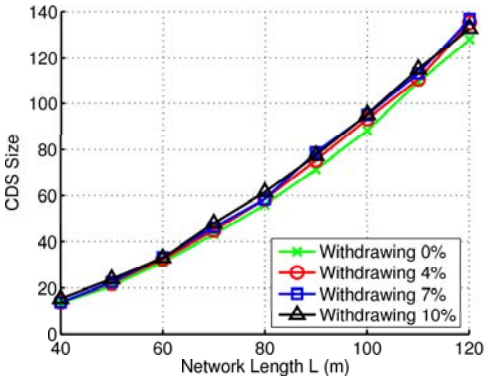


Figure 4.15: CDS size of Config-3: A small percentage of withdrawing nodes has very limited impact on the CDS size.

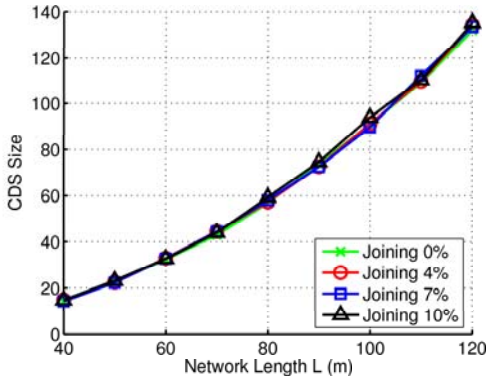


Figure 4.16: CDS size of Config-4: A small percentage of joining nodes has very limited impact on the CDS size.

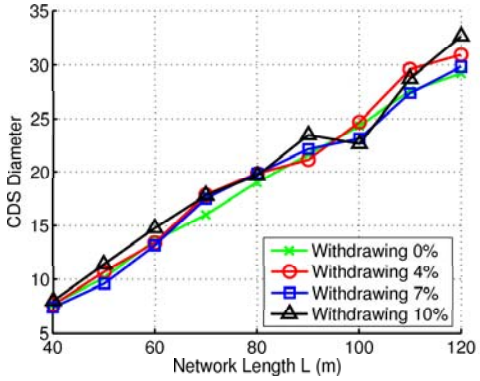


Figure 4.17: CDS diameter of Config-3: A small percentage of withdrawing nodes has very limited impact on the CDS diameter.

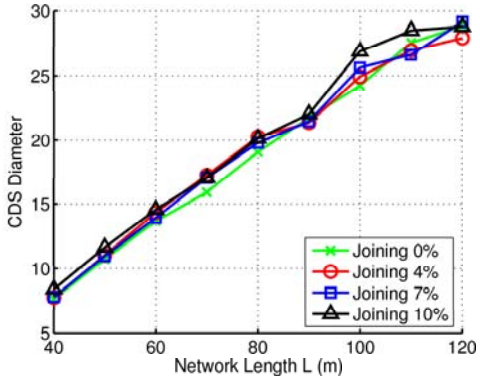


Figure 4.18: CDS diameter of Config-4: A small percentage of joining nodes has very limited impact on the CDS diameter.

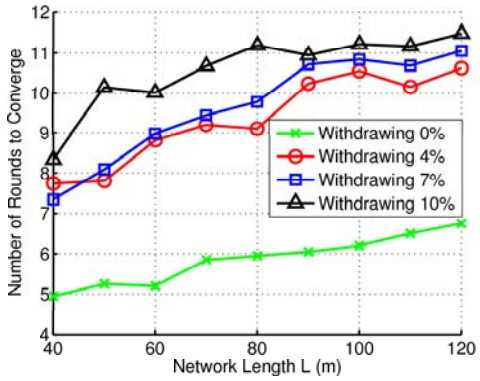


Figure 4.19: Number of rounds of Config-3: A small percentage of withdrawing nodes leads longer time to converge.

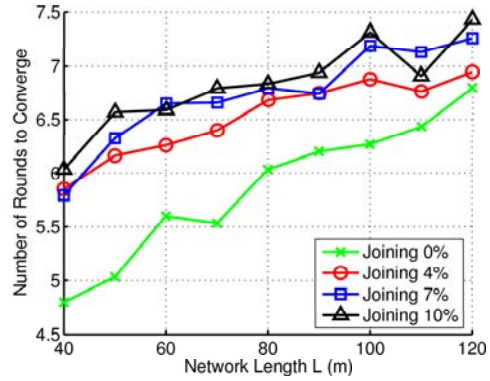


Figure 4.20: Number of rounds of Config-4: A small percentage of withdrawing nodes has very limited impact on convergence time.

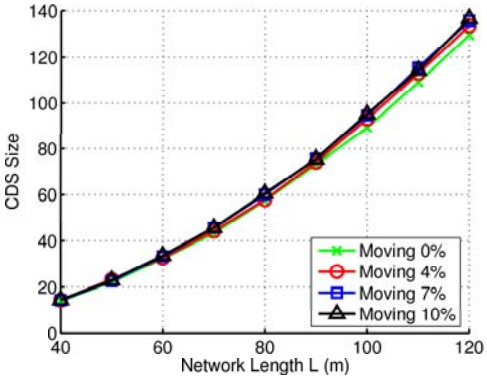


Figure 4.21: CDS size of Config-5: A small percentage of moving nodes has very limited impact on the CDS size.

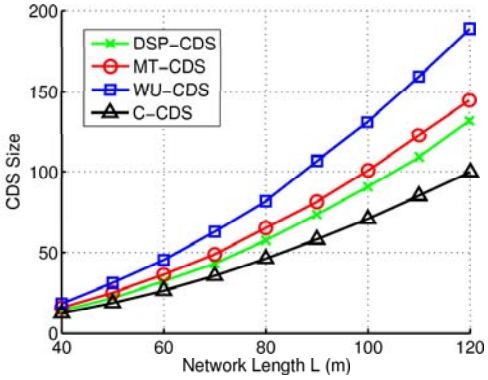


Figure 4.22: CDS size of Config-6: The CDS generated by DSP-CDS has smaller size than that of WU-CDS and MT-CDS.

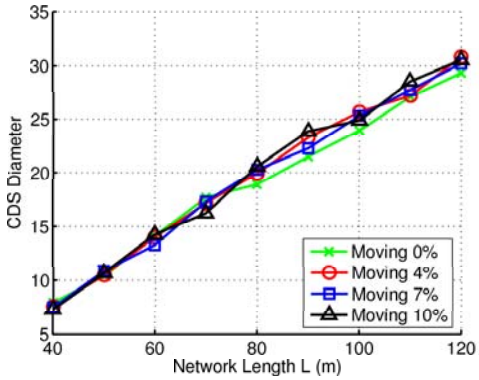


Figure 4.23: CDS diameter of Config-5: A small percentage of moving nodes has very limited impact on the CDS diameter.

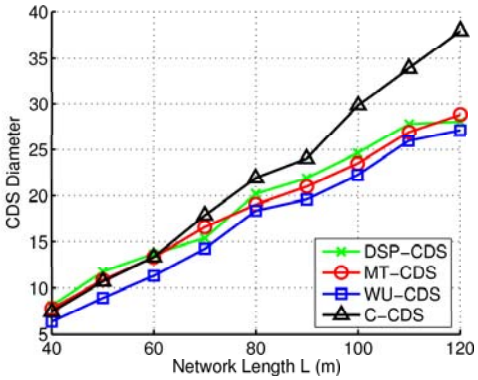


Figure 4.24: CDS diameter of Config-6: A CDS generated by WU-CDS has a smaller diameter than that of DSP-CDS and MT-CDS due to the large dominator population in WU-CDS.

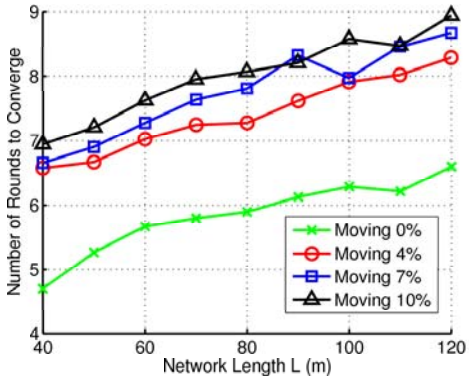


Figure 4.25: Number of rounds of Config-5: A small percentage of moving nodes leads longer time to converge.

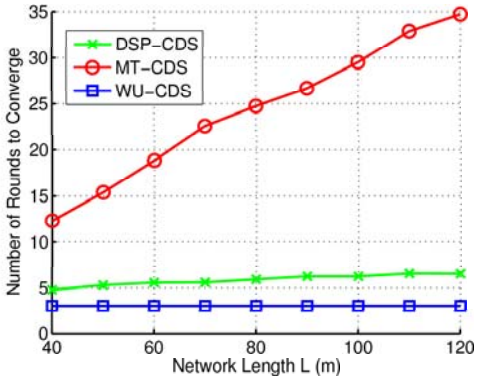


Figure 4.26: Number of rounds of Config-6: DSP-CDS always converges in no more than 8 rounds in a wide range of network sizes, which is much faster than MT-CDS.

4.3.2 Comparison with other CDS algorithms

The DSP-CDS algorithm is compared with two multiple-phase CDS algorithms: MT-CDS [83] and WU-CDS. WU-CDS is Wu and Li's algorithm [75]. In terms of CDS size, Wu and Li [75] show that WU-CDS performs better than Das et al.'s algorithm [20], while Zhou et al. [83] show that MT-CDS performs much better than WU-CDS and the algorithm by Wan et al. [71].

A centralized CDS algorithm, C-CDS, is used in the simulation for comparison purpose. In C-CDS, the node with a lowest ID is used as a root. First, add the root to the dominating set and dye its neighbors gray. Then, each time add the gray node, which can reduce the maximum number of white nodes, into the dominating set (dyed black), and its white neighbors are dyed gray. The process continues until there is no white nodes. The black nodes form a CDS. C-CDS is a centralized implementation of MT-CDS. C-CDS is expected to have smaller CDS sizes than the distributed algorithms.

Figure 4.22 shows that, in terms of CDS size, DSP-CDS performs much better than WU-CDS and MT-CDS. The connected dominating sets built by WU-CDS have smaller diameters in large networks (Figure 4.24), but the trade off is much greater dominator population.

DSP-CDS converges much faster than MT-CDS, as illustrated in Figure 4.26. The DSP-CDS algorithm always converges in no more than 8 rounds for a wide range of network sizes in our simulations, although the worst case time complexity is $O(|V|)$, where V is the set of nodes. Here, each round of MT-CDS is the time for generating a new layer of dominators away from the root.

The convergence time of MT-CDS is mainly affected by the network size, the node radio range, and the position of the initiator (who has the minimum node ID). The number of rounds in MT-CDS comes from its two phases. In the initiator election phase, the node with the minimum node ID broadcasts an announcement message

and waits until all the nodes in the network receive and accept it. In the dominator election phase, the generated dominators ripple away from the initiator round by round until they reach every corner of the network.

WU-CDS has a constant time complexity. In Figure 4.26, the number of rounds is set to 3, with a round for the marking process, pruning using rule 1, and pruning using rule 2, respectively. The main disadvantage of WU-CDS is the very large CDS size, as shown in Figure 4.22.

4.3.3 Message saving in static topology networks

To reduce the number of **STATUS** messages in a static topology network, a static version of DSP-CDS can be used. In the static version, a node broadcasts a **STATUS** message only upon the change of its status after the initial status exchanges with its neighbors.

4.4 Conclusion

This chapter has presented a distributed connected dominating set (CDS) construction algorithm, DSP-CDS, which constructs a CDS efficiently in a single phase in large *ad hoc* networks. DSP-CDS elects dominators across the network simultaneously. Each node only needs one-hop neighbor information and uses *strength* to decide locally whether to become a dominator. In a reliable and connected network, DSP-CDS converges when all nodes reach the zero strength, and all the dominators form a CDS. DSP-CDS uses several parameters to control the performance of the algorithm in terms of CDS size, CDS diameter, and the number of rounds to converge.

DSP-CDS adapts well to the dynamic network topology and updates a portion of the existing CDS in case of a topology change. The size and diameter of the updated CDS can be kept almost unchanged with 10% random node mobility.

DSP-CDS solves the phase-delay problem raised by many existing distributed CDS construction algorithms and generates a CDS of small size compared with those algorithms.

The work presented in this chapter has been published [79, 82].

Chapter 5

Energy Consumption in Clustered Wireless Sensor Networks

5.1 Introduction and Related Work

Energy is vital for many applications in wireless sensor networks. Clustering is one of the most important approaches to energy saving by keeping only a portion of nodes (cluster heads) active. Because high node density is typical in wireless sensor networks for fault tolerance purposes, turning redundant nodes into power-saving mode has significant effect for the energy saving purpose. Clustering is also an important approach to solving the capacity and scalability problems in wireless sensor networks where no physical infrastructure is available.

Most of the existing work on clustering in wireless sensor networks concerns a specific algorithm [11] [29] [76]. The primary considerations are energy consumption, throughput, and sensing coverage. Duarte-Melo and Liu [21] describe a heterogeneous network for data gathering and derive an energy analysis model. In their network, cluster heads are one hop away from the receiver and have higher communication capability. Normal nodes relay their sensing data to a nearby cluster head. Their

primary objective is to study the energy allocation among different types of nodes and estimate the network lifetime.

Transmission range adjustment as an energy saving approach has been the focus of numerous studies. Algorithms have been proposed either for the general topology control purpose [58] [55] [42] [37] [44] [47] or for special tasks [29] [9] [4] [72] , e.g., routing, data gathering, and broadcasting. The objective of energy saving is usually achieved by computing the best transmission range based on the geographic information and the energy model. Park and Sivakumar [51] have proposed a quantitative analysis model for the optimal transmission range problem in this category. They use throughput and throughput per unit energy as the optimization criteria and conclude that the optimal transmission power is determined by the network load, the number of nodes, and the network size. All nodes are active in their networks, and sleeping of nodes is not considered as an option for energy saving.

Little attention has been given to the transmission range issue of clustered networks in the existing research work, while both clustering and transmission range have great impacts on the energy consumption. This chapter introduces an energy analysis model for the optimal transmission range problem in clustered wireless sensor networks. The model is based on uniform networks with uniform passing traffic. It shows that the optimal transmission range in a homogeneous wireless network is a function of the traffic load and the node density. This also indicates that, in contrast to the claim in many studies ([5] [24] [48] [34] [12]), the minimal connected dominating set (MCDS) is usually not the optimal CDS structure in terms of energy saving. This chapter further discusses how this model can also be used in data gathering applications where the traffic is not uniform.

5.2 Energy Consumption in Clustered Wireless Sensor Networks

This chapter considers connected domating sets generated by the clustering methods. Assume a connected wireless sensor network where the nodes use the same transmission range and a subset of nodes forms a CDS of the network. Nodes in the CDS (called workers) work in the full-power mode, and nodes not in the CDS (called sleepers) work in the power-saving mode. We are to determine the optimal transmission range of the network, r_{opt}, so that the total energy consumption in the network is minimized under a given traffic load if all the nodes use r_{opt} as their transmission range.

In the full-power mode, a worker is always active in one of the three *active states*: transmitting, receiving, and idling. In the power-saving mode, a sleeper is active and performs its communication functions for a fraction of time. It sleeps to save energy most of the time. In the *sleeping* state, a node stops its communication functions, but may perform other activities, such as sensing. This is consistent with IEEE 802.11 Power Saving Mode (PSM) specification. While the receiving, idling, and sleeping powers are usually constants, the transmission power is related to the transmission range, because the received power of a signal falls off with the distance raised to the αth power, where α is the path loss exponent usually between 2 and 6 depending on the environment [56].

The total energy consumption in the network is determined by the time distribution among the four states of all the nodes in the network. The power of the four states are represented by: e_t, e_r, e_i, and e_s, respectively, and the periods of time node i spends on the four states are denoted by: t_t^i, t_r^i, t_i^i, and t_s^i, respectively. In a network with N notes, let

$$T_t = \sum_{i=1}^{N} t_t^i \qquad T_r = \sum_{i=1}^{N} t_r^i$$

$$T_i = \sum_{i=1}^{N} t_i^i \qquad T_s = \sum_{i=1}^{N} t_s^i \quad .$$

The total energy consumption in the network is:

$$E = e_t T_t + e_r T_r + e_i T_i + e_s T_s \quad . \tag{5.1}$$

Here, the sensing power can be included in one or more of the state powers.

This chapter considers the optimal radio range problem with respect to energy saving in wireless sensor networks. Two variable factors, traffic load and node density, that may impact the optimal radio range are studied.

Radio Range and Energy Consumption The transmission range affects the number of workers and sleepers and in turn affects the total energy consumption of the network. Most existing CDS algorithms tend to select smallest possible number of workers just enough to take care of the network [5] [24] [48] [34] [12]. Given a CDS algorithm, the number of workers selected is largely determined by the transmission range and the network size. A larger transmission range results in a smaller number of workers. More energy will be spent in each worker for forwarding packets due to the larger transmission range, but packet hops are reduced and more nodes work in the power-saving mode. Intuitively, there should be some optimal transmission range to minimize the total energy consumption in the network.

Optimal Radio Range and Traffic Load Heavier traffic surely increases the total energy consumption in the network. It may affect the optimal transmission range as well. A CDS algorithm mainly concerns topology information and the number of workers is not affected by traffic. Essentially, a heavier traffic (if not exceeding the node capacity) converts workers' idle time into transmitting time, and the total

energy consumption may be affected according to Equation (5.1), so is the optimal radio range.

Optimal Radio Range and Node Density In a uniformly deployed network, more nodes surely consume more energy if everything else remains unchanged, because the number of sleepers is increased. This essentially increases the proportion of sleeping time and idle time and may change the optimal radio range.

Each state time $(t_t^i, t_r^i, t_i^i, t_s^i)$ in Equation (5.1) is a function of transmission range, traffic load, and node density of the network. In a uniformly deployed network, if the traffic load and the node density are fixed, the optimal transmission range for energy saving can be determined. On the other hand, changes in traffic load or node density may affect the optimal transmission range.

5.3 Energy Consumption Analysis

5.3.1 Assumptions and notations

Network: Assume a uniformly deployed wireless sensor network within an $L \times W$ rectangular area with node density, ρ, defined as the number of nodes per unit area. All nodes in the network use the same transmission range, $r \in [0, r_{max}]$, and the data transmission rate is R_{tx} bps. Some clustering method is used to generate a connected dominating set (CDS) of the network as the communication backbone. A sleeper spends a fraction of time, f $(0 < f < 1)$, in active states. During a small time interval T, the number of workers and the number of sleepers can be regarded as unchanged.

Traffic: Assume no traffic is generated inside the network itself. Some outside traffic passes across the network from left to right, and the traffic load is proportional to the width of the network: μ packets per second per meter. Each packet has a fixed

length of l bytes. Packet forwarding happens only among workers. Sleepers never transmit packets, but they may overhear packets from neighbor workers during their active time. The traffic is medium and does not exceed the node capacity.

Although the analysis is based on the uniform network deployment and uniform passing traffic model, Section 5.3.4 will show that the optimal transmission range result based on this model can also be applied to non-uniformly deployed network with non-uniform traffic.

Energy Model: Sleeping, idling, receiving, and transmitting powers are e_s, e_i, e_r, and e_t, respectively, and

$$e_s = d, \quad e_i = a, \quad e_r = a + b, \quad e_t = a + cr^{\alpha} \quad , \tag{5.2}$$

where a, b, c, and d are constants determined by the electronic components of a node, and α is the communication path loss exponent.

Measures for low power radios [8] have shown that d is usually a very small number compared with a, and b is close to 0 so that $e_i \approx e_r$.

Table 5.1: Primitive Parameters of Energy Consumption Analysis

Parameters	**Determined by**
ρ, L, W	network deployment
R_{tx}, r_{max}	node capacity
T, f	MAC and topology control protocol
μ, l	traffic
a, b, c, d, α	node hardware and environment

5.3.2 Parameters

Primitive Parameters

Primitive parameters in the analysis are summarized in Table 5.1.

Derived Parameters

The other parameters used in the analysis can be derived.

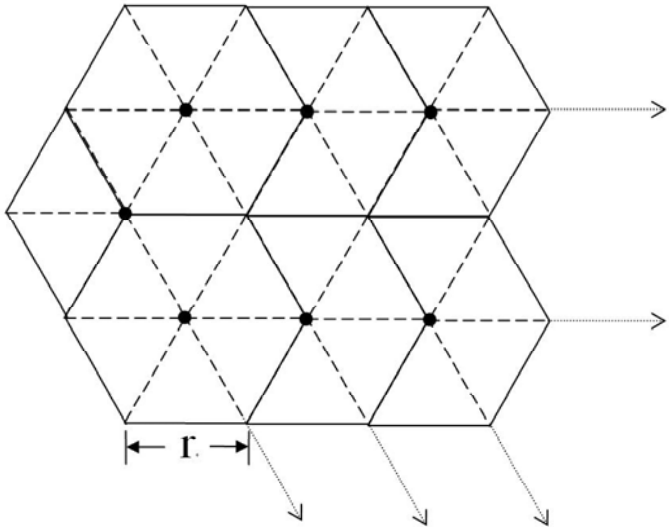


Figure 5.1: In the optimal layout of workers, each added worker can cover at most $\frac{2}{3}$ additional area of a hexagon and keep connection with at least one existing worker.

Network Parameters: Using a well-designed CDS clustering algorithm, the rotation between workers and sleepers should be scheduled so that the energy dissipation is evened out among all nodes. For a small time period T, assume the number of workers, N_w, and the number of sleepers, N_s, are fixed, and

$$N_w + N_s = N = \rho A \quad , \tag{5.3}$$

where $A = L \times W$ is the area of the network.

If each worker covers a regular hexagon, the optimal number of workers is determined by the network area A and the transmission range r shown in Figure 5.1. Each

added worker can cover at most $\frac{2}{3}$ additional area of a hexagon and keep connection with at least one existing worker. Therefore, the optimal number of workers is $\frac{A}{\sqrt{3}r^2}$. The number of workers is inversely proportional to the the covered area of a node, πr^2, and can be computed by

$$N_w = \frac{A}{h\pi r^2} \quad , \tag{5.4}$$

where h is a linear function of r^2 and A, and the coefficients are determined by the clustering algorithm. Then, the density of workers in the network is

$$\rho_w = \frac{N_w}{A} = \frac{1}{h\pi r^2} \quad . \tag{5.5}$$

Packet Transmitting/Receiving Time: The time to transmit a packet, t_p, can be computed with

$$t_p = l * 8/R_{tx} \quad , \tag{5.6}$$

and we assume it is equal to the time of receiving a packet.

Traffic Parameters: During T, the number of packets passing through the network is

$$P = \mu \times W \times T \quad . \tag{5.7}$$

Let r be the transmission range of the network, H the average hops of packet, λ the average number of packets received (or forwarded) by a worker, and J the estimated distance from a source to a sink. The total number of forwarded packets should be equal to the product of the number of incoming packets and the average number of hops:

$$N_w \cdot \lambda = P \cdot H \quad . \tag{5.8}$$

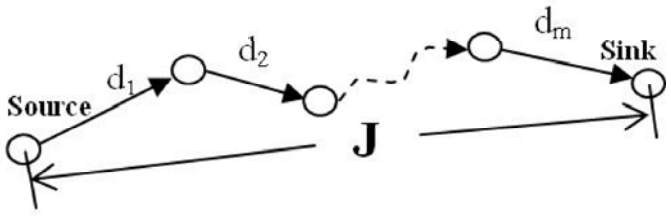


Figure 5.2: The packet path length from source to sink is sum of each hop distance $(d_1, d_2, ..., d_m)$, which is longer than the estimated distance (J).

Assume

$$P \cdot H \cdot r = P \cdot J \cdot \delta \quad , \tag{5.9}$$

where $\delta \geq 1$ is a constant so that $J \cdot \delta$ is the average path length from a source to a sink (as shown in Figure 5.2). A denser network has a smaller value of δ. The δ is a function of r and J, and the coefficients are determined by the clustering algorithm.

From (5.4)(5.7)(5.8)(5.9) we have

$$\lambda = \frac{h\pi TJ\delta W}{A} \cdot \mu r \quad . \tag{5.10}$$

Therefore, the number of packets one worker forwards is proportional to its transmission range and network traffic load.

5.3.3 Energy Consumption

According to Equation (5.1), the total energy consumption of the network can be computed if the time each node spends in the four states is known.

With the assumption that the network deployment and the passing traffic are uniform, the average time for workers and sleepers can be computed.

Let X be the packet arrival distribution in a worker and Y_w be the worker neighbor

distribution of a node. Assume X and Y_w are independent. Thus,

$$E(X) = \lambda, \quad E(Y_w) = \pi r^2 \rho_w = \frac{1}{h}, \quad E(XY_w) = \frac{\lambda}{h} \quad .$$

During the small time interval T, the average time of worker nodes is

$$\begin{aligned}
\bar{t}_{wt} &= E(X)t_p = \lambda t_p \\
\bar{t}_{wr} &= E(XY_w)t_p = \frac{\lambda}{h} t_p \\
\bar{t}_{wi} &= T - t_{wt} - t_{wr} = T - \lambda t_p (1 + \frac{1}{h}) \\
\bar{t}_{ws} &= 0 \quad ,
\end{aligned}$$

where $\bar{t}_{wt}$, $\bar{t}_{wr}$, $\bar{t}_{wi}$, and $\bar{t}_{ws}$ are the average transmitting time, receiving time, idling time, and sleeping time of workers, respectively.

A sleeper spends fT in active mode during T. Assume it overhears a proportional amount of traffic in the active mode. Therefore, the average time of sleepers is

$$\begin{aligned}
\bar{t}_{st} &= 0 \\
\bar{t}_{sr} &= f\bar{t}_{wr} = f\frac{\lambda t_p}{h} \\
\bar{t}_{si} &= fT - \bar{t}_{st} - \bar{t}_{sr} = fT - f\frac{\lambda t_p}{h} \\
\bar{t}_{ss} &= T(1-f) \quad ,
\end{aligned}$$

where $\bar{t}_{st}$, $\bar{t}_{sr}$, $\bar{t}_{si}$, and $\bar{t}_{ss}$ are the average transmitting time, receiving time, idling time, and sleeping time of sleepers, respectively.

Let T_t, T_r, T_i, and T_s be the total transmitting time, receiving time, idling time and sleeping time of all nodes, then

$$
\begin{aligned}
T_t &= N_w t_{wt} + (N - N_w) t_{st} \\
T_r &= N_w t_{wr} + (N - N_w) t_{sr} \\
T_i &= N_w t_{wi} + (N - N_w) t_{si} \\
T_s &= N_w t_{ws} + (N - N_w) t_{ss} \quad .
\end{aligned}
$$

Finally, the total energy consumption of the network during T can be computed using Equation (5.1).

The average energy consumption rate (energy consumption per second per unit area) is

$$
\begin{aligned}
e &= \frac{E}{AT} \\
&= ct_p \cdot \frac{\delta J}{L} \cdot ur^{\alpha-1} + fbt_p \rho\pi \cdot \frac{\delta J}{L} \cdot ur \\
&+ (1-f)\frac{bt_p}{h} \cdot \frac{\delta J}{L} \cdot \frac{\mu}{r} \\
&+ (1-f)\frac{(a-d)}{h\pi} \cdot \frac{1}{r^2} \\
&+ (d + f(a-d))\rho \quad . \qquad (5.11)
\end{aligned}
$$

Rewrite e as

$$
e = k_4 \mu r^{\alpha-1} + k_3 \rho\mu r + k_2 \frac{\mu}{r} + k_1 \frac{1}{r^2} + k_0 \rho \quad , \qquad (5.12)
$$

where

$$
\begin{aligned}
k_4 &= ct_p \cdot \frac{\delta J}{L} \\
k_3 &= fbt_p\pi \cdot \frac{\delta J}{L} \\
k_2 &= (1-f)\frac{bt_p}{h} \cdot \frac{\delta J}{L} \\
k_1 &= \frac{(1-f)(a-d)}{h\pi} \\
k_0 &= (d + f(a-d)) \quad .
\end{aligned}
$$

After the deployment of the network, k_i $(i = 0, 1, 2, 3, 4)$ are constants. The total energy consumption is a function of the transmission range r, traffic load μ, and node density ρ. The optimal transmission range for energy saving is a function of μ and ρ, the traffic load and node density.

The receiving power is often very close to the idling power for RF devices. Thus, Equation (5.12) can be simplified by assuming $e_i = e_r$, thus, $b = 0$, $k_2 = 0$, and $k_3 = 0$. Therefore:

$$
e = k_4\mu r^{\alpha-1} + k_1\frac{1}{r^2} + k_0\rho \quad . \tag{5.13}
$$

By taking $e' = 0$, we have

$$
\begin{aligned}
r_{opt} &= (\frac{2k_1}{(\alpha-1)\mu k_4})^{\frac{1}{\alpha+1}} \\
&= (\frac{2(1-f)(a-d)}{c(\alpha-1)h\pi} \cdot \frac{L}{\delta J} \cdot \frac{1}{t_p\mu})^{\frac{1}{\alpha+1}} \quad .
\end{aligned} \tag{5.14}
$$

Note that, $t_p\mu$ is the traffic load in terms of bits per second per meter.

5.3.4 Discussion

From Equation (5.14), we know $r_{opt} \propto \frac{1}{\mu}$, which means the optimal transmission range is smaller for heavier traffic. Note that r_{opt} is not related to node density ρ, which means ρ has no impact on the optimal radio range when $e_r = e_i$ holds. The smaller transmission range is preferred in environments with higher path loss exponent (α), because transmission power increases exponentially with α. A greater transmission range is preferred if nodes are equipped with higher data rate transmission devices (t_p is smaller), in which case the transmission time will be reduced.

With the estimated optimal transmission range, proper transmission devices for applications can be determined beforehand. The network lifetime can be estimated if the topology control algorithm can handle the worker and sleeper rotation well enough to achieve an even energy dissipation in the network. With the insight of the relationship between transmission range and traffic, traffic in the network can be better scheduled to extend the network lifetime.

The energy analysis model is based on the assumptions that the node deployment and traffic are uniform across the network (Section 5.3.1), and analysis result of Equation (5.14) is for the global optimal transmission range used by all nodes in the network. Since every node has the same view on the network, this global optimal transmission range should be the local optimal transmission range as well. Otherwise, any *better* local transmission range in one region should also apply to all regions in the network, which conflicts with the global optimal result. Therefore, in a network with non-uniform node densities and with non-uniform traffic loads, if there is a way for the nodes to estimate the traffic load and node densities in their *relatively uniform* local region, the result of Equation (5.14) still applies. Section 5.5 will discuss the application of this analysis result in data gathering applications.

5.4 Simulation

Matlab simulations are used to validate the accuracy of the energy analysis model and further study the energy consumption behavior in clustered wireless sensor networks.

5.4.1 Simulation environment

Table 5.2: Parameter Values of Energy Consumption Simulation

Parameters	Values
(a, b, c, d)	(0.083, 0.017, 0.00002, 0.013)
α	3.5
ρ	0.01 - 0.03 nodes per m^2
T	100 seconds
f	0.1
(L, W)	$(100m, 100m)$
μ	various
l	64 bytes
R_{tx}	250 kbps
r_{max}	48m
J	$1.08L$
h	$(-0.5372r^2 + 0.3269LW) \times 10^{-4}$
δ	$-0.0114r + 0.0186J$

In simulations, nodes are randomly deployed in the rectangular area. Uniform passing traffic is injected by 10 source nodes on the left edge of the network. The destination of a packet is randomly selected from the 10 sink nodes on the right edge of the network. A simple geographic routing algorithm is used, and each time the packet is forwarded to the worker node to reduce the distance to the destination most. DSP-CDS [82] is used as the clustering algorithm to select workers.

Parameter settings in the simulation are summarized in Table 5.2. Heuristic coefficients for the functions δ and h are derived from historical data of simulations.

5.4.2 Observations

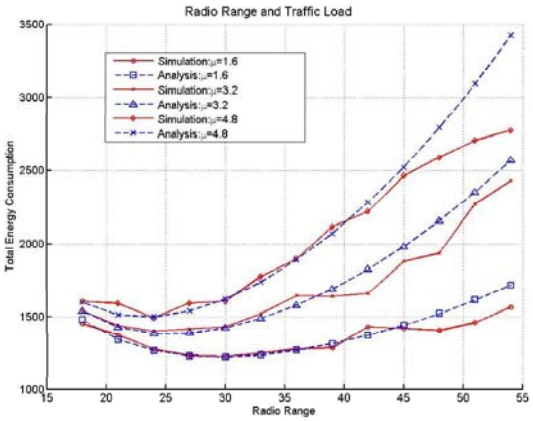


Figure 5.3: Radio range and traffic load ($\rho = 0.01$): Analysis results well fit simulation results. The optimal transmission range decreases as the traffic load increases.

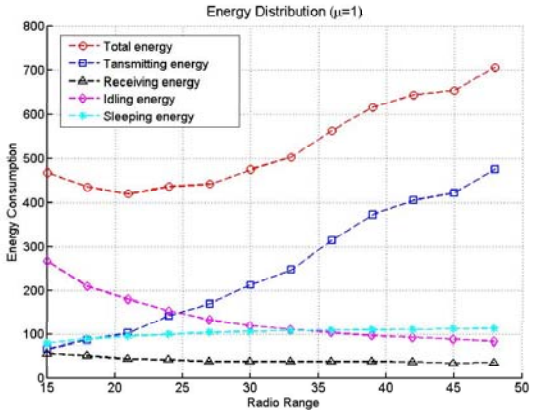


Figure 5.4: Energy distribution: Radio range affects energy distribution among different states dramatically.

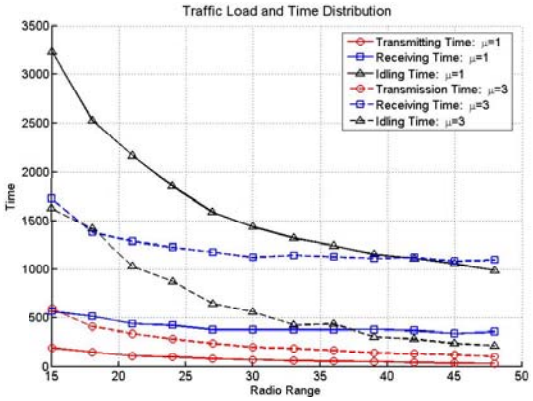


Figure 5.5: Traffic load and time distribution: A heavier traffic converts more idle time into transmitting time and receiving time.

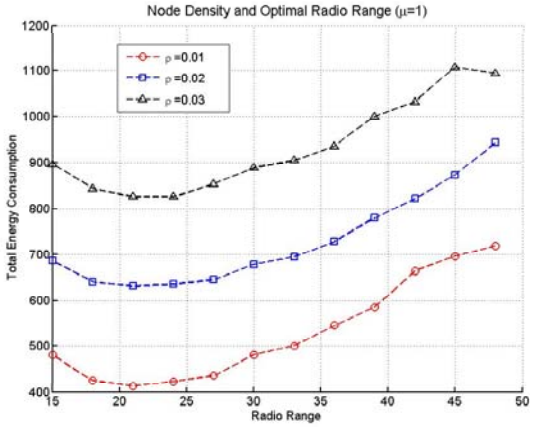


Figure 5.6: Node density: Node density shifts the energy consumption curves but does not change the optimal transmission range.

Figure 5.3 shows that the analysis results fit well the simulation results. The error is only significant when the transmission range is very large. The explanation for that is when the transmission range is very large (over 45 in the simulations) the border effect invalidates the uniform assumptions in the analysis.

In all the network configurations, with the transmission range varying from 15 to 48, the average energy consumption first decreases but then increases. The optimal transmission range is obviously somewhere in between. Energy consumptions on transmitting, receiving, idling, and sleeping states change differently as the transmission range increases, and the total energy consumption is their combined result, as shown in Figure 5.4.

Impact of Transmission Range and Traffic Load As Figure 5.3 shows, the optimal transmission range decreases as the traffic load μ increases. This is consistent with the discussion in Section 5.3.4.

The traffic load impact mainly results from the transmitting and receiving time changes in the network. A heavier traffic converts more idle time into transmitting time and receiving time, as illustrated in Figure 5.5.

Impact of Node Density Figure 5.6 shows that the node density shifts the energy consumption curves along Y axis and does not change the optimal transmission range. This is also consistent with the discussion in Section 5.3.4.

5.5 Optimal Transmission Range in Data Gathering Applications

Many applications of sensor networks require that data be collected from all nodes of the network. A node generates its own packets (sensing data) and may forward packets for other nodes in a multi-hop network. Given the network conditions in Section 5.2, the sleepers transmit their sensing data to nearby workers during the active time, and the workers forward all the incoming packets (together with its own sensing data) to sinks. If all nodes sense and transmit data at the same rate, the

energy consumption for the data sensing and transmitting to the first worker is the same for all the nodes.

Assume some sink nodes are deployed on the right edge of the network to collect sensing data. The nodes located nearer the sinks observe heavier traffic load unless very high data fusion rate can be achieved or only statistic data is reported.

As discussed in Section 5.3.4, the global optimal transmission range from our analysis model should also be the local optimal choice in non-uniform traffic network. In data gathering applications, although the traffic load is different across the network, it can be assumed constant in a relatively small region. In each small region, the uniform energy analysis model can still be used to find the optimal transmission range, and the regions near sinks will use smaller transmission ranges due to the heavier traffic, as illustrated in Figure 5.7.

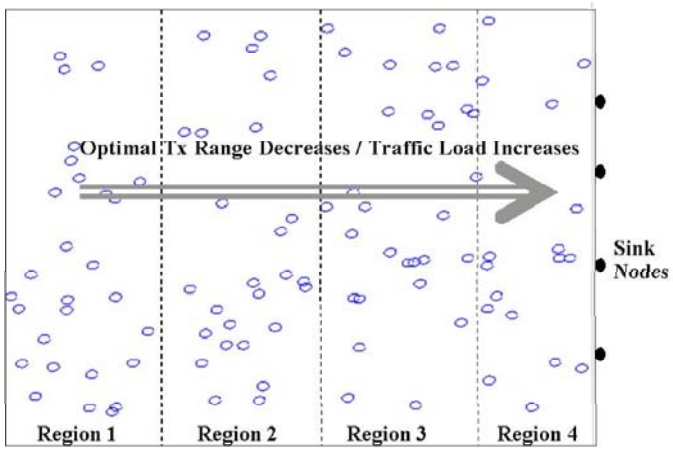


Figure 5.7: Transmission range setting in data gathering applications.

The traffic load in a small region can be estimated based on the data fusion rate, the data sensing rate, and the location of the region in the network. If the regions are small enough in Figure 5.7, nodes in each region should observe the same traffic load without considering the edge effect. They forward all the traffic that comes in from the left and their own traffic to the next region on the right, until the sinks are reached. If there are n nodes in each region, and each node reports sensing data at

the rate of m packets per second. During a time period of T, the nodes in region i forward $n \times m \times T \times i$ packets. This amount of traffic can be regarded as uniform traffic across region i, so that the uniform energy analysis model can be used.

5.6 Conclusion

This chapter has analyzed the optimal transmission range problem for energy consumption in clustered wireless sensor networks and established an energy consumption model based on the assumptions of uniform deployment and uniform traffic. The accuracy of the model is verified by simulations under various network configurations. The model provides an insight into the energy consumption behavior in clustered wireless sensor networks. The optimal transmission range in a clustered wireless sensor networks is a function of the traffic load and the node density, but the traffic load has a much greater impact on the optimal transmission range than the node density. The optimal transmission range can be estimated, and the energy consumption behavior can be predicted before network deployment, which is important for choosing RF devices for nodes, estimating lifetime of a network, and scheduling network traffic. Although the analysis is based on the uniform network deployment and uniform passing traffic model, the optimal transmission range result can be applied to non-uniformly deployed network with non-uniform traffic as well. Furthermore, the data gathering application is used as an example to discuss how to apply this model in networks where the traffic is not uniform.

The work presented in this chapter has been published [81].

Chapter 6

Adaptive Clustering and Transmission Range Adjustment

6.1 Introduction

Traffic, network deployment, and environment are major factors affecting the performance of a topology control algorithm and the total energy consumption in clustered wireless sensor networks. As discussed in Chapter 5, the optimal transmission range (in terms of energy saving) is a function of traffic load, while the traffic load often has unpredictable changes after deployment. In many applications, traffic load is not uniform across the network and changes during the network operation time. A traffic-adaptive clustering algorithm may better fit into this situation. Using a traffic-adaptive clustering algorithm, a node decides its transmission range based on its local traffic load and updates the local network topology accordingly. A traffic-adaptive clustering algorithm may not generate the optimal cluster structure (in terms of energy saving) all the time. But it is expected to outperform the fixed transmission range assignment in a dynamic traffic load environment.

Dynamic traffic load is common in wireless sensor networks. One example is the

data gathering application, where many source nodes send data packets to a few sink nodes. The traffic load in the network gradually increases as it comes closer to the sinks if no significant data fusion is in use. In may cases, the sinks are mobile agents, which makes the traffic distribution more complicated. Therefore, the optimal transmission range should be different across the network. Another example of dynamic traffic load is the surveillance application. The network traffic load can be very low most of the time, but it will increase abruptly as some phenomenon is observed. Therefore, the optimal transmission range should be different during the network operation time. In both examples, the time, location, and amount of traffic changes cannot be determined before deployment. Fixed transmission range clustering algorithms will not have the optimal energy saving effect all the time.

Although it is impractical to obtain accurate traffic load changes in the network, a node can estimate the traffic load in its local region by exchanging traffic information with its neighbors. If the local traffic load can keep relatively constant for a period of time, the node can decide its optimal transmission range based on the analysis in Chapter 5. If all nodes can choose their transmission ranges based on the proper estimation of their local traffic load in a traffic-adaptive clustering algorithm, more energy saving can be expected.

The traffic-adaptive clustering algorithm, RDSP-CDS, is designed with this goal in mind. RDSP-CDS extends DSP-CDS (Chapter 4) with a transmission range adjustment ability by utilizing the energy consumption analysis in Chapter 5. Using the RDSP-CDS algorithm, all nodes choose their optimal transmission ranges independently and form a network with asymmetric communication channels. The optimal transmission range of a node is based on its local knowledge about the network traffic and topology.

RDSP-CDS is a distributed algorithm that runs at each node. It does not require any location or angle-of-arrival information. It works in heterogeneous networks

where nodes have different maximal transmission ranges. RDSP-CDS adapts well to dynamic network topologies caused by transmission range changes, node mobility, node failure, and deployment of new nodes.

6.2 Related Work

Various adaptive algorithms have been proposed in wireless sensor networks with different objectives. Cerpa and Estrin have proposed ASCENT [8], a duty-cycle-based algorithm (Section 2.1). In ASCENT, a fraction of nodes are active, and the other nodes work in the passive mode initially to save energy. During the packet forwarding process, the unacceptable end-to-end packet loss rate triggers more nodes on the route to become active and participate in the multihop data transmission. Han et al.'s TCPE [28] and Lin et al's ATPC [46] are both transmission-power-based algorithms (Section 2.1). They adaptively control the transmission powers of nodes based on the link quality among nodes. In TCPE, the minimum transmission powers between nodes are collected using a power estimation technique. TCPE considers not only the various maximum transmission powers the nodes can use in a heterogeneous network, but also the various thresholds of receiving powers. In ATPC, pairwise models are built among neighboring nodes to describe the correlation between the transmission power and the link quality. Based on this model, the authors have developed an adaptive algorithm to control the transmission power of nodes based on the feedback of link qualities. Park and Sivakumar [52] show that, for a typical *ad hoc* network, the optimal topology is a function of the traffic load. They have further proposed an idea to estimate the traffic load by measuring local transmission contention time and adapt the transmission power to it. Three ATC algorithms (ATC-CP, ATC-IP, and ATC-MS) have been proposed to employ the idea. Su and Lee [63] have proposed a fault-tolerant gateway assignment protocol in sensor networks. The

protocol is a clustering method, and the assignment of cluster heads is adaptive to node failure, traffic load, and energy level.

RDSP-CDS and the ATC-IP algorithm by Park and Sivakumar [52] are both distributed algorithms where the nodes independently adapt their transmission powers to the local traffic load, but they are different in traffic load estimation, transmission power adjustment, and construction of network topology. RDSP-CDS estimates the local traffic information by exchanging data packet information with neighbors, while ATC-IP does it by measuring the channel contention time. RDSP-CDS adjusts the transmission power by using the optimal transmission range analysis (Chapter 5), while ATC-IP does it by using a lower threshold and a upper threshold. At last, RDSP-CDS is a clustering algorithm. In a typical wireless sensor network, nodes are densely deployed, and clustering is much more efficient in energy saving than transmission power adjustment [80].

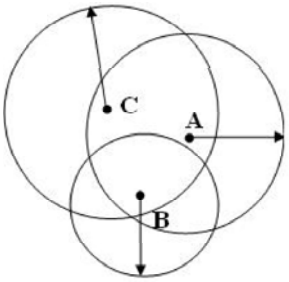
6.3 RDSP-CDS, a Traffic-Adaptive Clustering Algorithm

RDSP-CDS extends DSP-CDS in two aspects. First, it works in networks with asymmetric communication channels. Secondly, RDSP-CDS changes the transmission ranges of nodes according to their local traffic loads. The letter R in RDSP-CDS means *range*, which indicates both extensions.

6.3.1 Dealing with Asymmetric Communication Channels

In a network with asymmetric communication channels (i.e., a heterogeneous network where nodes have different maximal transmission ranges), two concepts of neighbors are defined. Node B is a *unidirectional neighbor*, or *uni-neighbor* of node A if A can receive packets sent by B. If two nodes are unidirectional neighbors of each other,

they are *bidirectional neighbors*, or *bi-neighbor* of each other (Figure 6.1).



Node	Uni-neighbors	Bi-neighbors
A	C	C
B	A C	
C	A	A

Figure 6.1: Uni-neighbors and bi-neighbors of a node characterize the direction of communication channels.

Each node keeps two separate *nid* (node ID) lists for its uni-neighbors and bi-neighbors, respectively. The **STATUS** message (Section 4.2.2) in DSP-CDS (Chapter 4) is extended to incorporate the uni-neighbor list of the sender. A receiver of the **STATUS** message adds the sender into its uni-neighbor list. If the receiver finds its own *nid* in the uni-neighbor list of the message, it will add the sender into its bi-neighbor list as well. After exchanging **STATUS** messages for several rounds, each node builds up its uni-neighbor list and bi-neighbor list.

The bi-neighbors in RDSP-CDS are used almost the same as the neighbors in DSP-CDS. A node keeps the status information (*pid*, *state*, and *strength*) for its bi-neighbors. Bi-neighbors are used for strength computation and data communication. In addition, RDSP-CDS also saves traffic information for each bi-neighbor, which will be used for transmission range adjustment (Section 6.3.2). The bidirectional communication channel between two bi-neighbors is ideal for network protocol design. In essence, RDSP-CDS generates a subnetwork N' with symmetric channels from the original network N with asymmetric channels and performs the clustering and data communication on N'. If $G(W)$ is the graph for a network W, $G(N')$ is a symmetric directed spanning graph of the directed graph $G(N)$.

The uni-neighbor set of a node is a superset of its bi-neighbor set. No status information is saved for a uni-neighbor, however. Although uni-neighbors are not directly

used by RDSP-CDS for clustering and communication, they are used for maintaining the bi-neighbor list and the network topology. The *neighborTimer* (Section 4.2.3) from DSP-CDS applies on uni-neighbors. If the *neighborTimer* for a uni-neighbor expires (due to transmission range changes, node movement, or node failure), the node removes the *nid* of the neighbor from both its uni-neighbor list and bi-neighbor list.

The *strength* indicates the ability of a node to connect with different pieces during the clustering process. The definition of *strength* should be as fair as possible, and the difference in transmission range of nodes in RDSP-CDS must be considered. If the *strength* definition from DSP-CDS is used in RDSP-CDS, a node with a longer transmission range is more likely to get a greater *strength* and thus has better chance to become a dominator. It is more reasonable if *strength* reflects the area of the region over which those points contributed to the *strength* are earned. In a uniformly deployed network, the number of bi-neighbors that may contribute to the *strength* of a node i is propotinal to r_i^2, where r_i is the transmission range of node i. In the RDSP-CDS, the *strength* of node i is defined as

$$strength_{i-rdsp} = \frac{strength_{i-dsp}}{r_i^2}, \tag{6.1}$$

where $strength_{i-rdsp}$ is the *strength* value of node i in RDSP-CDS definition, and $strength_{i-dsp}$ is the *strength* value of node i in DSP-CDS definition (Section 4.2.2).

6.3.2 Adjusting Transmission Range According to Traffic Load

In a uniformly deployed network with a uniform traffic load, the global optimal transmission range is also the local optimal transmission range, as discussed in Section 5.3.4. The local optimal transmission range can then be computed using Equation 5.14. In a network with non-uniform traffic, the best choice of a node with only

local traffic estimation is to use the local optimal result from Equation 5.14 as well. This will also be validated by simulations.

There can be many ways for a node in RDSP-CDS to estimate its local traffic. The implementation in this chapter uses the **A**verage estimation from its k-hop (close) **N**eighbors (AkN, for short) as its own estimation. In AkN, a node i makes a traffic load estimation ($\hat{\mu}_i^k$) every T seconds by

$$\hat{\mu}_i^k = \frac{C_i^k}{F^k D_i^k T} \quad , \tag{6.2}$$

where D_i^k is the radius of the local circular area covering the node and its k-hop neighbors, C_i^k is the total number of transmission when data packets pass across this area, F^k is the expected number of transmission when a packet passes across this area, and T is the traffic load estimation interval (Figure 6.2).

T is computed from clock readings of node i. C_i^k, F^k, and D_i^k are computed with

$$C_i^k = f_i + \sum_{j \in B_k(i)} f_j$$

$$F^k = \begin{cases} 2k+1, & \text{if } i \text{ is a dominator} \\ 2k, & \text{if } i \text{ is not a dominator} \end{cases}$$

$$D_i^k = \begin{cases} (k+1)r_i, & \text{if } k = 0, 1 \\ r_i + \sum_{j=2}^{k-1} \overline{R}_j(i) + 2\overline{R}_k(i), & \text{if } k \geq 2 \end{cases}$$

where f_i is the number of data packets forwarded by node i since the last traffic load estimation, $B_k(i)$ is the k-hop bi-neighbor set of node i, r_i is the current transmission range of the node, and $\overline{R}_j(i)$ is the average transmission range of the jth hop bi-neighbors of node i. The **STATUS** message is further extended to include the number of data packets the sender has forwarded so far. Each node remembers the number of data packets it has forwarded and reads the values of its bi-neighbors from the latest

STATUS messages.

All the other parameters in the optimal transmission range equation (Equation 5.14) are constants or can be derived from the constants after deployment. Therefore, with the estimated traffic load $\hat{\mu}_i^k$, node i can decide its optimal transmission range.

6.3.3 Re-clustering versus Dynamic Adjustment

Using RDSP-CDS, the network can be either scheduled to re-cluster with the new transmission range setting during its operation, or adjust dynamically to it. The transmission range changes lead to network topology changes, and RDSP-CDS is designed to deal with dynamic network topology.

With the re-clustering method, the network operates in a series of time frames, as shown in Figure 6.2(b). Nodes are synchronized to re-cluster the network and generate a new dominating set. The duration of each time frame is t (Figure 6.2(a)) and consists of a short protocol channel with the duration of ft ($f < 1$) and a traffic channel with the duration of $(1 - f)t$. The protocol channel is for maintaining the network topology, and the traffic channel is for data packet transmission. With the dynamic adjustment method, RDSP-CDS updates the local structure of dominating set upon transmission range changes. There is no fixed re-clustering phase, and the network-wide synchronization is not necessary. But the dynamic adjustment method may generate less optimal CDS structure than the re-clustering method in the long run due to the accumulated local changes on the dominating set.

6.4 Simulation and Discussion

All simulations are conducted in Matlab. In each simulation, nodes are randomly deployed in an $L \times W$ rectangular region. The number of nodes is decided by the rectangular area and the node density ρ, which is the number of nodes per unit area.

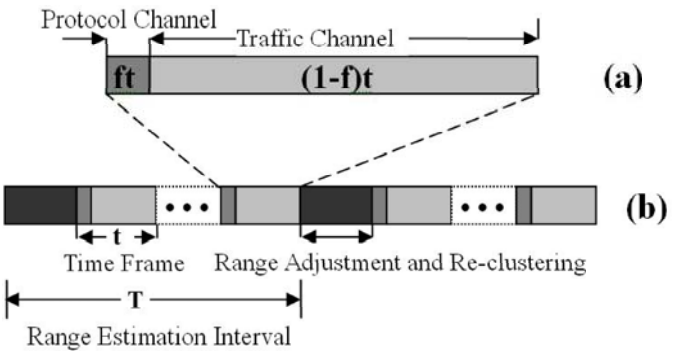


Figure 6.2: RDSP-CDS operates in a series of time frames.

The same energy model as in Section 5.2 is used. Data traffic will be described shortly in Section 6.4.1.

In a simulation, all nodes use the same initial transmission range $r_i = r_{init}$ ($i = 1...n,\ r_{init} \in [r_{min}, r_{max}]$), where n is the number of nodes, and r_{min} and r_{max} are the minimal transmission range and maximum transmission range a node can use. The r_{min} and r_{max} are determined by the transmission device on a node. The deployment should guarantee that the network is connected when all the nodes operate with $r_i = r_{min}$.

RDSP-CDS and DSP-CDS are compared by going through the same set of configurations. Each configuration is different from another only by the initial transmission range (r_{init}). To our best knowledge, there is no other clustering algorithm that adjusts the transmission ranges of nodes according to the traffic load, so no result from other adaptive algorithm is compared. As discussed in Section 6.2, the ATC-IP algorithm [52] does adjust the transmission range to traffic load, but it is not a clustering algorithm.

The re-clustering method (Section 6.3.3) is used for RDSP-CDS. The simulation time of each configuration is mT ($m \geq 1$). The first stage of T time is the initial stage for collecting traffic information, $r_i = r_{init}$ remains unchanged during this stage. At the beginning of each stage starting from the second one, each node computes a

new $r_{i.opt} \in [r_{min}, r_{max}]$ based on its local traffic load. DSP-CDS operates with no transmission range adjustment for the entire simulation time of mT.

In the first stage of T time, there is actually no significant difference between RDSP-CDS and DSP-CDS, because the transmission ranges of the nodes remain unchanged in both algorithms. Compared with the data traffic, the protocol traffic is very limited in all aspects including the duration, packet size, and number of packets. Therefore, the simulation results are only for the data traffic in the last $m-1$ stages.

6.4.1 Traffic Patterns

RDSP-CDS is designed for non-uniform traffic networks, where it is expected to outperform the fixed transmission range algorithms, like DSP-CDS. However, in order to clearly demonstrate the characteristics of RDSP-CDS, comparison with DSP-CDS in uniform traffic network is conducted as well.

Uniform Traffic Setting: Constant-Bit-Rate Passing Traffic

In this setting, constant bit rate traffic (CBR) is generated at the left border of the network at μ data packets per second per meter. The packets pass through the network and reach the destination nodes uniformly sitting along the right border of the network. The traffic load is relatively uniform across the network. Figure 6.3 illustrates the uniform traffic setting.

Non-uniform Traffic Setting: Data Gathering Traffic

In this setting, each node in the network generates μ data packets per second and sends them to one or more sinks sitting at the right border of the network. The nodes that locate nearer the sinks forward more data packets and observe heavier traffic. From the discussion in Chapter 5, they should use shorter transmission ranges than those nodes located further away from the sinks. The traffic load distribution is

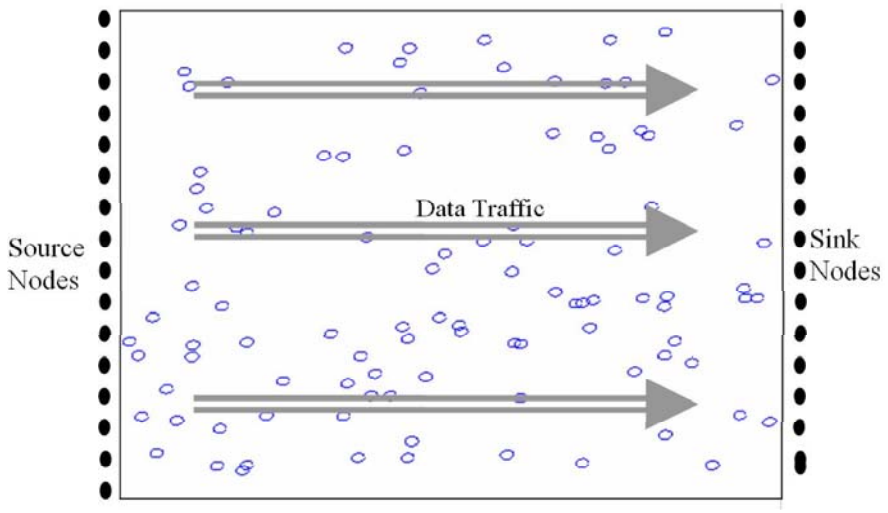


Figure 6.3: In the uniform traffic setting, packets generated at the left border pass through the network and reach the sinks at the right border.

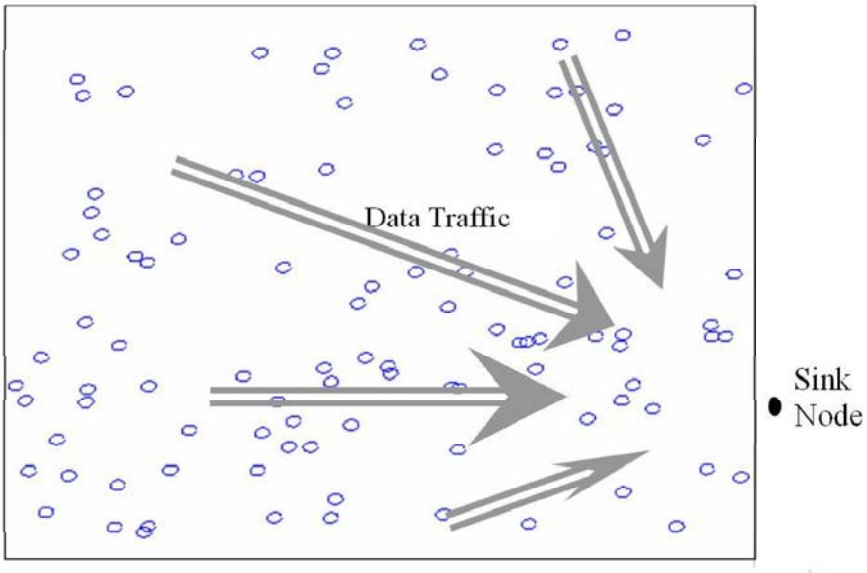


Figure 6.4: In the non-uniform traffic setting, packets generated at nodes in the network converge at the sink.

related to the sink locations. In many applications, the sink locations are not fixed, neither the distribution of traffic load. This is where the traffic-adaptive clustering algorithm as RDSP-CDS is more suitable. Figure 6.4 illustrates the non-uniform traffic setting.

6.4.2 Simulation Parameter Setting

Table 6.1 summarizes the variables used in simulation configurations. In addition to the ones described above, there are more variables in the table whose definitions can be found in Chapter 5.

Table 6.1: Simulation Parameter Setting

Parameters	Values
(a, b, c, d)	(0.083, 0, 0.00002, 0.013)
α	3.5
ρ	0.015
T	100 seconds
m	3
f	0.1
$(L \times W)$	$(200m \times 200m)$
μ	1.0
l	64 bytes
R_{tx}	250 kbps
r_{init}	configuration variable
(r_{min}, r_{max})	(16m, 48m)
J	$1.08L$
h	$(-0.5372r^2 + 0.3269LW) \times 10^{-4}$
δ	$-0.0114r + 0.0186J$

$A1N$ is used for traffic estimation, i.e., one-hop neighborhood traffic is used in the estimation.

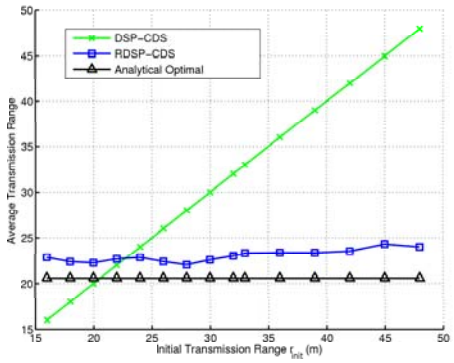


Figure 6.5: With the uniform traffic, RDSP-CDS has a consistent transmission range decision, and the decision is close to the analytical optimal value.

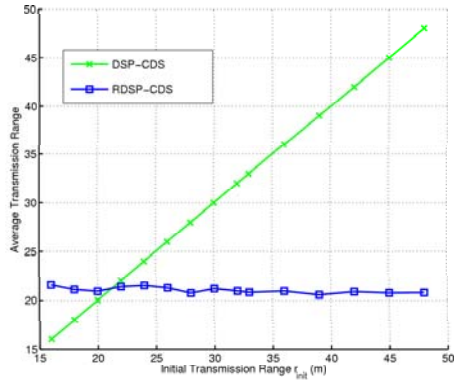


Figure 6.6: With the non-uniform traffic, RDSP-CDS has a consistent transmission range decision.

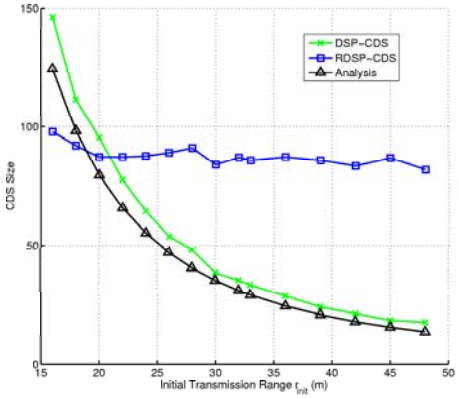


Figure 6.7: With the uniform traffic, RDSP-CDS generates a set of CDSs with similar sizes.

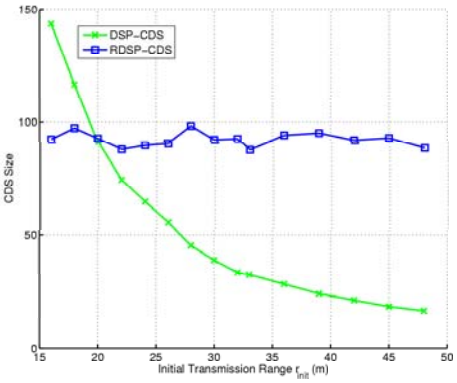


Figure 6.8: With the non-uniform traffic, RDSP-CDS generates a set of CDSs with similar sizes.

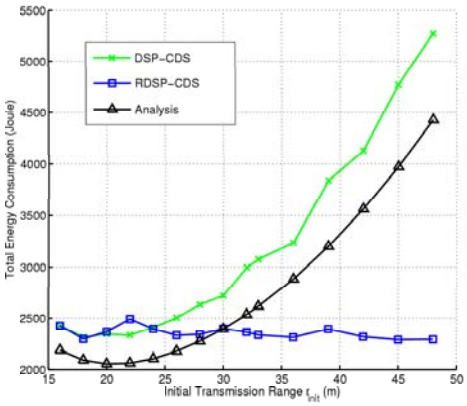


Figure 6.9: With the uniform traffic, RDSP-CDS consumes similar amount of energy to the optimal configuration (with $r = 20m$) of DSP-CDS.

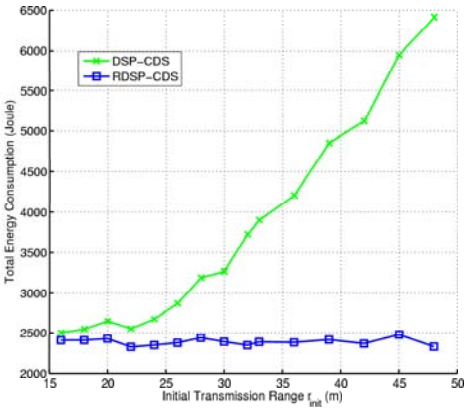


Figure 6.10: With the non-uniform traffic, RDSP-CDS consumes less energy than DSP-CDS with any transmission range.

6.4.3 Simulation Result

Figures 6.5, 6.7, 6.9, 6.11 and 6.13 are simulation results for the uniform traffic setting, compared one by one with Figures 6.6, 6.8, 6.10, 6.12 and 6.14, the simulation results for non-uniform traffic setting. Since the analysis assumptions hold only on the uniform traffic (Chapter 5), no analysis result appears for the non-uniform cases.

With RDSP-CDS, a node adjusts its transmission range according to its local traffic load. In other words, the transmission range of a node depends only on the estimation of the local traffic load. Since the traffic is the same for any configuration, nodes using RDSP-CDS should have a consistent decision on the transmission range for all configurations no matter what the initial transmission range (r_{init}) is, while nodes in DSP-CDS will keep r_{init} unchanged. This is exactly what Figure 6.5 and Figure 6.6 have shown. In the uniform traffic case (Figure 6.5), the average transmission range is very close to the analytical optimal value for the traffic in all configurations. Considering together with the unanimous choice of nodes on the transmission range in Figure 6.13, nodes using RDSP-CDS can well adapt their transmission ranges to their local traffic.

As a natural result from the transmission range conditions in Figure 6.5 and Figure 6.6, RDSP-CDS generates a set of CDSs with similar sizes for all configurations in contrast to the decreased sizes in DSP-CDS, as shown in Figure 6.7 and Figure 6.8.

The single most important design goal of RDSP-CDS is to reduce the total energy consumption of the network under non-uniform traffic. Figure 6.10 shows that RDSP-CDS consumes less energy than DSP-CDS no matter what transmission range is used by DSP-CDS. Even in the uniform traffic network, as shown in Figure 6.9, RDSP-CDS consumes similar amount of energy compared with the optimal configuration (with $r = 20m$) of DSP-CDS. More important, in all configurations, the total energy consumption remains almost unchanged in RDSP-CDS. For a given traffic load, the simulations show that regardless of the initial transmission range setting, RDSP-CDS

can always choose the similar transmission range setting (Figure 6.6 and Figure 6.8) which is most energy efficient (Figure 6.10). The effectiveness of RDSP-CDS in non-uniform traffic networks is validated.

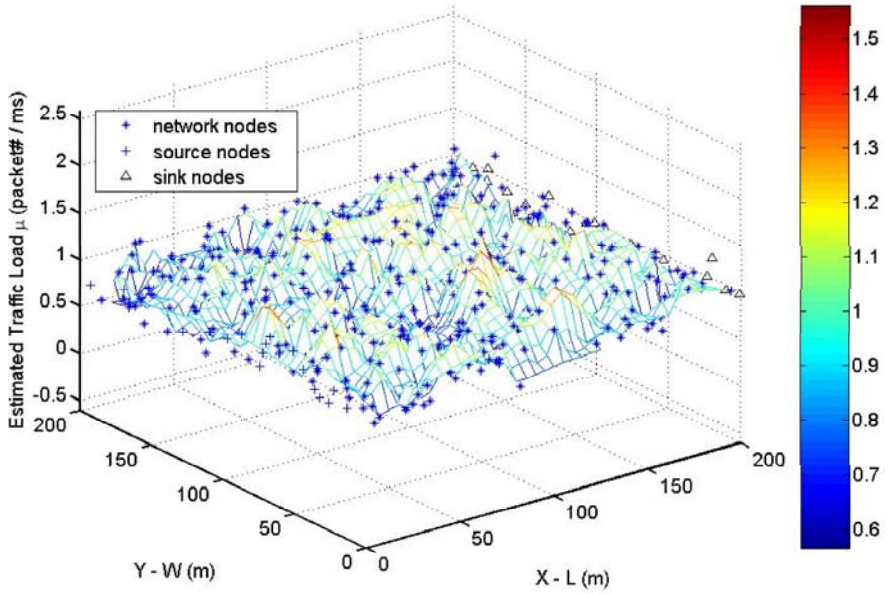


Figure 6.11: With the uniform traffic, nodes have similar estimated traffic load ($average = 0.96$ and $variance = 0.04$).

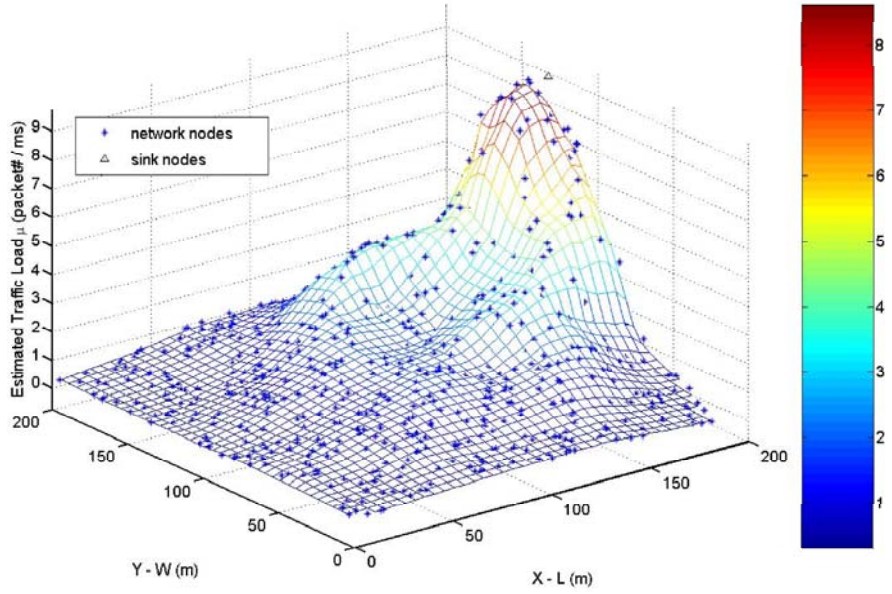


Figure 6.12: With the non-uniform traffic, nodes near to the sink have greater estimated traffic load ($average = 0.65$ and $variance = 0.13$).

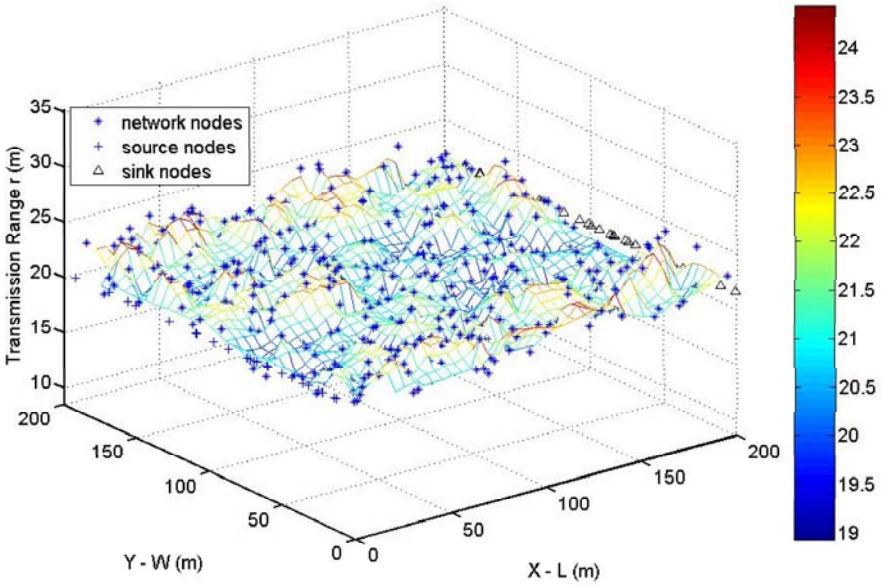


Figure 6.13: With the uniform traffic, nodes have similar transmission ranges ($average = 20.94$ and $variance = 1.23$).

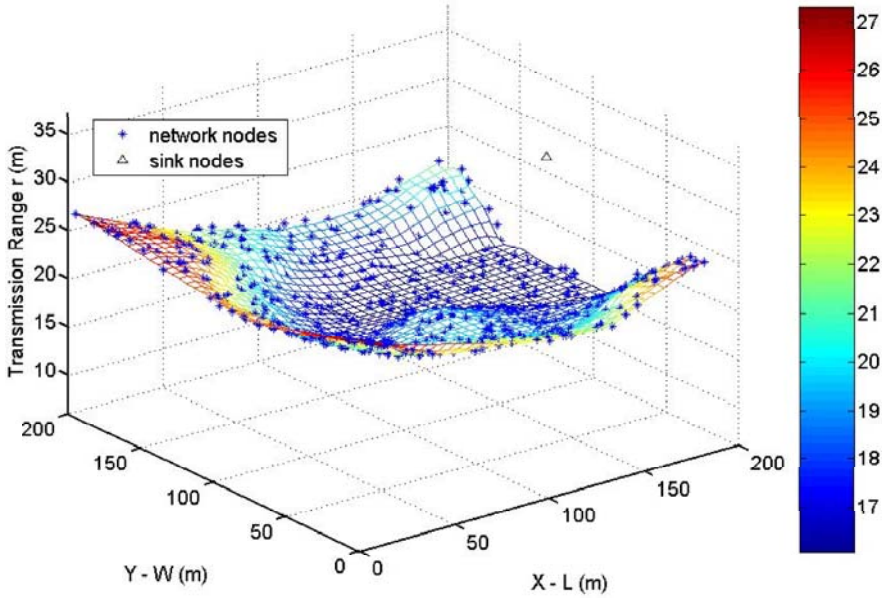


Figure 6.14: With the non-uniform traffic, nodes near to the sink have shorter transmission ranges ($average = 24.10$ and $variance = 7.68$).

Now, Let us take a closer look at the behavior of each node in one particular configuration with $\mu = 1.0$, $\rho = 0.015$, and $r_{init} = 27$. The mark $*$ shows the location of each node in the network together and the grids show the estimated traffic load (Figure 6.11 and Figure 6.12) or transmission range (Figure 6.13 and Figure 6.14). The mark Δ shows the location of each sink node.

Figure 6.11 shows that the local traffic loads estimated by nodes in the network are similar across the network in the uniform traffic setting. Especially, the overall result is close to the configured $\mu = 1$ with an average of 0.96 and a variance of 0.04. Figure 6.12 shows that the estimated local traffic load gradually increases as the nodes come closer to the sink in the non-uniform traffic setting. Both figures show that the nodes using RDSP-CDS algorithm can accurately estimate the network traffic load.

Figure 6.13 and Figure 6.14 are the transmission range results of each node after being adjusted according to the local traffic load. They are directly related to the traffic load results of Figure 6.11 and Figure 6.12, because the optimal transmission range is inversely proportional to the traffic load according to Equation 5.14.

6.5 Conclusion

This chapter has presented RDSP-CDS, an algorithm for adaptive clustering and transmission range adjustment in wireless sensor networks. In addition to all the features of DSP-CDS (Chapter 5), RDSP-CDS chooses the optimal transmission range for each node based on the node's local knowledge about the traffic load to save more energy. Through Matlab simulations, it has been shown that RDSP-CDS always consumes less energy than the fixed transmission range algorithm DSP-CDS regardless of the initial transmission range assignment in non-uniform traffic networks, while it consumes comparable amount of energy as DSP-CDS with the optimal transmission range setting in uniform traffic networks. Therefore, in terms of energy saving, RDSP-

CDS is a better choice as a CDS clustering algorithm for wireless sensor networks with dynamic traffic load.

RDSP-CDS is suitable for heterogeneous networks where nodes have different maximal transmission powers. It adapts to dynamic network topology.

Chapter 7

Conclusions and Future Work

This dissertation discusses the clustering and transmission range adjustment issues of topology control in wireless sensor networks (WSNs), especially how the two approaches can be used together to achieve better energy saving. As a summary, the contributions of the dissertation include a two-level topology control strategy, a distributed connected dominating set construction algorithm (DSP-CDS), an energy consumption analysis model to solve the optimal transmission range problem in clustered WSNs, and a distributed traffic-adaptive clustering algorithm (RDSP-CDS) for non-uniform traffic networks.

The two-level topology control strategy is designed to integrate the existing *duty-cycle-based* and *transmission-power-based* topology control approaches in WSNs to achieve further energy saving. While both the *duty-cycle-based* approach and the *transmission-power-based* approach have their preferred network conditions where one outperforms the other, the two-level topology control strategy can achieve better performance than either of them in terms of energy saving.

DSP-CDS is a distributed connected dominating set (CDS) construction algorithm, which constructs a CDS efficiently in a single phase in large *ad hoc* networks. DSP-CDS is asynchronous and elects dominators across the network simultaneously.

DSP-CDS uses several parameters to control the performance of the algorithm in terms of CDS size, CDS diameter, and number of rounds to converge. DSP-CDS adapts well to dynamic network topology and updates a portion of the existing CDS in case of a topology change. DSP-CDS solves the phase-delay problem raised by many existing distributed CDS construction algorithms and generates a CDS of small size compared with those algorithms.

The optimal transmission range problem for energy consumption in clustered WSNs is analyzed, and an energy consumption model is established. The accuracy of the model is verified by simulations under various network configurations. The optimal transmission range in a clustered wireless sensor network is shown as a function of the traffic load and the node density, but the traffic load has a much greater impact on the optimal transmission range than the node density. The optimal transmission range can be estimated, and the energy consumption behavior can be predicted before network deployment, which is important for choosing RF devices for nodes, estimating lifetime of a network, and scheduling traffic load. Although the analysis is based on the uniform network deployment and uniform passing traffic model, the optimal transmission range result based on this model can also be applied to non-uniformly deployed networks with non-uniform traffic.

A distributed traffic-adaptive connected dominating set construction algorithm, RDSP-CDS, is presented as an effort to utilize the energy analysis model in a distributed clustering algorithm. RDSP-CDS chooses the optimal transmission range for each node based on the node's local knowledge about the traffic load to save more energy DSP-CDS regardless of the initial transmission range assignment in non-uniform traffic networks. RDSP-CDS is suitable for heterogeneous networks and dynamic network topology.

In the future, the energy consumption model can be extended for both clustered and non-clustered networks. The general model should answer the question: Under

what specific conditions the network should be clustered and how? New adaptive algorithms can should be designed to adjust the transmission powers and the duty cycles based on the analysis model.

Bibliography

[1] M. S. Al-kahtani and H. T. Mouftah. Enhancements for clustering stability in mobile ad hoc networks. In *Q2SWinet '05: Proceedings of the 1st ACM international workshop on Quality of service & security in wireless and mobile networks*, pages 112–121, New York, NY, USA, 2005. ACM Press.

[2] K. M. Alzoubi, X.-Y. Li, Y. Wang, P.-J. Wan, and O. Frieder. Geometric spanners for wireless ad hoc networks. *IEEE Transactions on Parallel Distributed Systems*, 14(4):408–421, 2003.

[3] Atmel Corporation. ATMega128 and ATMega128L data sheet. In *www.atmel.com*, 2006.

[4] S. Banerjee and A. Misra. Minimum energy paths for reliable communication in multi-hop wireless networks. In *Proceedings of MobiHoc*, pages 146–156, 2002.

[5] L. Bao and J. J. Garcia-Luna-Aceves. Topology management in ad hoc networks. In *Proceedings of the 4th ACM Interational Symposium on Mobile Ad Hoc Networking and Computing, MobiHoc 2003*, pages 129–140, 2003.

[6] D. M. Blough, M. Leoncini, G. Resta, and P. Santi. On the symmetric range assignment problem in wireless ad hoc networks. In *Proceedings of IFIP TCS*, pages 71–82, 2002.

[7] M. Cardei, J. Wu, and S. Yang. Topology control in ad hoc wireless networks using cooperative communication. *IEEE Transactions on Mobile Computing*, 5(6):711–724, 2006.

[8] A. Cerpa and D. Estrin. Ascent: Adaptive self-configuring sensor networks topologies. *IEEE Transactions on Mobile Computing*, 3(3):272–285, 2004.

[9] J.-H. Chang and L. Tassiulas. Energy conserving routing in wireless ad-hoc networks. In *Proceedings of IEEE INFOCOM 2000*, pages 22–31, 2000.

[10] M. Chatterjee, S. K. Das, and D. Turgut. WCA: A weighted clustering algorithm for mobile ad hoc networks. *Cluster Computing*, 5(2):193–204, 2002.

[11] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris. Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks. *ACM Wireless Networks*, 8(5):481–494, September 2002.

[12] Y. P. Chen and A. L. Liestman. Approximating minimum size weakly-connected dominating sets for clustering mobile ad hoc networks. In *Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing (MobiHoc 2002:)*, pages 165–172, New York, NY, USA, 2002. ACM Press.

[13] C.-Y. Chiu, G.-H. Chen, and E. H.-K. Wu. A stability aware cluster routing protocol for mobile ad hoc networks. *Wireless Communications and Mobile Computing*, 3(4):503–515, 2003.

[14] I. Cidon and O. Mokryn. Propagation and leader election in a multihop broadcast environment. In *Proceedings of the 12th International Symposium on Distributed Computing (DISC 1998)*, pages 104–118, London, UK, 1998. Springer-Verlag.

[15] B. N. Clark, C. J. Colbourn, and D. S. Johnson. Unit disk graphs. *Discrete Mathematics*, 86(1-3):165–177, 1990.

[16] A. E. F. Clementi, P. Penna, and R. Silvestri. Hardness results for the power range assignmet problem in packet radio networks. In *Proceedings of RANDOM-APPROX*, pages 197–208, 1999.

[17] N. R. Council. *Embedded, Everywhere: A Research Agenda for Networked Systems of Embedded Computers.* National Academies Press, October 2001.

[18] Crossbow Technology Inc. Micaz mote developers kit. In *MOTE-KIT2400 Datasheet, www.xbow.com*, 2006.

[19] B. Das and V. Bharghavan. Routing in ad-hoc networks using minimum connected dominating sets. In *Proceedings of International Conference on Communications'97*, pages 376–380, 1997.

[20] B. Das, R. Shivkumar, and V. Bharghavan. Routing in ad hoc networks using a virtual backbone. In *Proceedings of the International Conference on Computers and Communication Networks (IC3N)*, pages 1–20, Septempber 1997.

[21] E. J. Duarte-Melo and M. Liu. Analysis of energy consumption and lifetime of heterogeneous wireless sensor networks. In *Proceedings of IEEE GLOBECOM 2002*, pages 21–25, Nov 2002.

[22] D. Estrin, D. Culler, K. Pister, and G. Sukhatme. Connecting the physical world with pervasive networks. *IEEE Pervasive Computing*, 1(1):59–69, January–March 2002.

[23] D. Estrin, R. Govindan, J. S. Heidemann, and S. Kumar. Next century challenges: Scalable coordination in sensor networks. In *Proceedings of Mobile Computing and Networking 1999*, pages 263–270, 1999.

[24] B. Gao, Y. Yang, and H. Ma. An effective distributed approximation algorithm for constructing minimum connected dominating set in wireless ad hoc networks. In *Proceedings of CIT*, pages 658–663, 2004.

[25] M. R. Garey and D. S. Johnson. *Computers and Intractability : A Guide to the Theory of NP-Completeness*. W. H. Freeman, January 1979.

[26] S. Guha and S. Khuller. Approximation algorithms for connected dominating sets. In *Proceedings of Fourth Annual European Symposium on Algorithms*, pages 179–193, 1996.

[27] P. Gupta and P. R. Kumar. Critical power for asymptotic connectivity in wireless networks. In W. M. McEneany, G. Yin, and Q. Zhang, editors, *Stochastic Analysis, Control, Optimization and Applications: A Volume in Honor of W.H. Fleming*, pages 547–566. Birkhauser, Boston, 1998.

[28] X. Han, C.-C. Shen, and Z. Huang. Adaptive topology control for heterogeneous mobile ad hoc networks using power estimation. In *IEEE Wireless Communications and Networking Conference, 2006. WCNC 2006*, pages 392 – 399, April 2006.

[29] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks. In *Proceedings of the Hawaii International Conference on System Sciences (HICSS 2000)*, pages 3005–3014, January 2000.

[30] J. L. Hill and D. E. Culler. Mica: A wireless platform for deeply embedded networks. *IEEE Micro*, 22(6):12–24, 2002.

[31] T.-C. Hou and V. O. LI. Transmission range control in multihop packet radio networks. *IEEE Transactions on Communications*, 34(1):38–44, January 1986.

[32] J. Hui, Z. Ren, and B. H. Krogh. Sentry-based power management in wireless sensor networks. In F. Zhao and L. Guibas, editors, *Proceedings of Information Processing in Sensor Networks (IPSN)*, volume 2634 of *Lecture Notes in Computer Science*, pages 458–472, Palo Alto, CA, USA, April 2003. Springer-Verlag.

[33] S. Hussain and A. Matin. Hierarchical cluster-based routing in wireless sensor networks. In *Proceedings of IPSN 2006, Work-in-progress track*, pages 214–226, April 2006.

[34] L. Jia, R. Rajaraman, and T. Suel. An efficient distributed algorithm for constructing small dominating sets. *Distributed Computing*, 15(4):193–205, 2002.

[35] R. Kershner. The number of circles covering a set. *American Journal of Mathematics*, 61:665–671, 1939.

[36] L. M. Kirousis, E. Kranakis, D. Krizanc, and A. Pelc. Power consumption in packet radio networks. *Theoretical Computer Science*, 243(1-2):289–305, 2000.

[37] M. Kubisch, H. Karl, A. Wolisz, L. C. Zhong, and J. Rabaey. Distributed algorithms for transmission power control in wireless sensor networks. In *Proceedings of IEEE Wireless Communications and Networking Conference (WCNC'03)*, pages 558–563, Mar 2003.

[38] F. Kuhn and R. Wattenhofer. Constant-time distributed dominating set approximation. In *Proceedings of 22nd ACM Symposium on Principles of Distributed Computing (PODC 2003)*, pages 25–32, 2003.

[39] S. Kutten and D. Peleg. Fast distributed construction of small k-dominating sets and applications. *Journal of Algorithms*, 28(1):40–66, 1998.

[40] P. Levis, N. Lee, M. Welsh, and D. Culler. Tossim: Accurate and scalable simulation of entire tinyos applications. In *Proceedings of the First ACM Conference on Embedded Networked Sensor Systems (SenSys 2003)*, November 2003.

[41] P. Levis, S. Madden, J. Polastre, R. Szewczyk, K. Whitehouse, A. Woo, D. Gay, J. Hill, M. Welsh, E. Brewer, and D. Culler. TinyOS: An operating system for sensor networks. *Ambient Intelligence*, April 2005.

[42] L. Li, J. Halpern, V. Bahl, Y.-M. Wang, and R. Wattenhofer. Analysis of a cone-based distributed topology control algorithm for wireless multihop networks. In *Proceedings of Twentieth ACM Symposium on Principles of Distributed Computing (PODC 2001), Newport, Rhode Island*, August 2001.

[43] N. Li and J. C. Hou. Localized topology control algorithms for heterogeneous wireless networks. *IEEE/ACM Transactions on Networking*, 13(6):1313–1324, 2005.

[44] N. Li, J. C. Hou, and L. Sha. Design and analysis of an mst-based topology control algorithm. In *Proceedings of IEEE INFOCOM 2003*, pages 1702–1712, Mar./Apr. 2003.

[45] B. Liang and Z. J. Haas. Virtual backbone generation and maintenance in ad hoc network mobility management. In *Proceedings of IEEE INFOCOM 2000*, pages 1293–1302, 2000.

[46] S. Lin, J. Zhang, G. Zhou, L. Gu, T. He, and J. A. Stankovic. Atpc: Adaptive transmission power control for wireless sensor networks. In *Fourth ACM Conference on Embedded Networked Sensor Systems (SenSys 2006)*, November 2006.

[47] J. Liu and B. Li. Distributed topology control in wireless sensor networks with asymmetric links. In *Proceedings of IEEE GLOBECOM 2003*, volume 3, pages 1257–1262, Dec 2003.

[48] K. Mnif, B. Rong, and M. Kadoch. Virtual backbone based on mcds for topology control in wireless ad hoc networks. In *Proceedings of the 2nd ACM international*

workshop on Performance evaluation of wireless ad hoc, sensor, and ubiquitous networks(PE-WASUN 2005:), pages 230–233, New York, NY, USA, 2005. ACM Press.

[49] M. Niedermayer, R. Thomasius, S. Kai, P. David, G. Stephan, W. John, and H. Reichl. Miniaturizion platform for wireless sensor nodes based on 3d-packaging technologies. In *Proceedings of IPSN 2006*, pages 391 – 398, 2006.

[50] A. Nosratinia, T. E. Hunter, and A. Hedayat. Cooperative communication in wireless networks. *IEEE Communications Magazine*, 42(10):74 – 80, October 2004.

[51] S.-J. Park and R. Sivakumar. Quantitative analysis of transmission power control in wireless ad-hoc networks. In *Proceedings of International Workshop on Ad Hoc Networking (IWAHN), Vancouver, Canada*, pages 56–63, Aug 2002.

[52] S.-J. Park and R. Sivakumar. Adaptive topology control for wireless ad hoc networks. *Mobile Computing and Communications Review*, 7(3):37–38, 2003.

[53] V. Paruchuri, A. Durresi, and R. Jain. Optimized flooding protocol for ad hoc networks. *Computing Research Repository(CoRR): Networking and Internet Architecture*, 2003.

[54] J. Polastre, R. Szewczyk, and D. Culler. Telos: Enabling ultra-low power wireless research. In *Proceedings of IPSN Track on Sensor Platform, Tools and Design Methods for Networked Embedded Systems (SPOTS 2005)*, 2005.

[55] R. Ramanathan and R. Hain. Topology control of multihop wireless networks using transmit power adjustment. In *Proceedings of IEEE INFOCOM 2000*, pages 404–413, 2000.

[56] T. S. Rappaport. *Wireless Communications: Principles and Practice*. Prentice-Hall, 2nd edition, 2002.

[57] RFM Monolithics, Inc. TR1000 916.50 MHz Hybrid Transceiver. In *http://www.rfm.com/products/data/tr1000.pdf*, 2005.

[58] V. Rodoplu and T. H. Meng. Minimum energy mobile wireless networks. In *Proceedings of the 1998 IEEE International Conference on Communications, ICC'98, Atlanta, GA*, volume 3, pages 1633–1639, June 1998.

[59] P. Santi. Topology control in wireless ad hoc and sensor networks. *ACM Computing Surveys*, 37(2):164–194, 2005.

[60] A. Scaglione, D. L. Goeckel, and J. N. Laneman. Cooperative communications in mobile ad hoc networks. *IEEE Signal Processing Magazine*, 23(5):18 – 29, September 2006.

[61] M. Stemm and R. H. Katz. Measuring and reducing energy consumption of network interfaces in hand-held devices. *IEICE Transactions on Communications*, E80-B(8):1125–31, 1997.

[62] I. Stojmenovic, M. Seddigh, and J. Zunic. Dominating sets and neighbor elimination-based broadcasting algorithms in wireless networks. *IEEE Transactions on Parallel and Distributed Systems*, 13(1):14–25, 2002.

[63] W. W. Su and S.-J. Lee. An adaptive and fault-tolerant gateway assignment in sensor networks. In *IEEE International Conference on Mobile Ad-hoc and Sensor Systems, 2004*, pages 576 – 578, October 2004.

[64] R. Szewczyk, A. M. Mainwaring, J. Polastre, J. Anderson, and D. E. Culler. An analysis of a large scale habitat monitoring application. In *Proceedings of SenSys 2004*, pages 214–226, 2004.

[65] The MathWorks, Inc. Matlab - the language of technical computing. In *http://www.mathworks.com/products/matlab/*, 2006.

[66] The Rice Monarch Project. Wireless and mobility extensions to ns-2. In *http://www.monarch.cs.cmu.edu/cmu-ns.html*, 2000.

[67] G. Tolle, J. Polastre, R. Szewczyk, D. E. Culler, N. Turner, K. Tu, S. Burgess, T. Dawson, P. Buonadonna, D. Gay, and W. Hong. A macroscope in the redwoods. In *Proceedings of SenSys 2005*, pages 51–63, 2005.

[68] University of Southern California Information Sciences Institute. The network simulator - ns-2. In *http://www.isi.edu/nsnam/ns/*, 2006.

[69] S. Verblunsky. On the least number of unit circles which can cover a square. *Journal of the London Mathematical Society (0024-6107)*, 24:164–170, 1949.

[70] N. Vlajic and D. Xia. Wireless sensor networks: To cluster or not to cluster? In *2006 International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM 2006)*, pages 258–268, 2006.

[71] P.-J. Wan, K. M. Alzoubi, and O. Frieder. Distributed construction of connected dominating set in wireless ad hoc networks. *Mobile Networks and Applications (MONET)*, 9(2):141–149, 2004.

[72] P.-J. Wan, G. Calinescu, X.-Y. Li, and O. Frieder. Minimum-energy broadcasting in static ad hoc wireless networks. *Wireless Networks*, 8(6):607–617, 2002.

[73] R. Wattenhofer, L. Li, P. Bahl, and Y.-M. Wang. Distributed Topology Control for Power Efficient Operation in Multihop Wireless Ad Hoc Networks. In *Proceedings of IEEE INFOCOM 2001*, April 2001.

[74] K. Whitehouse, G. Tolle, J. Taneja, C. Sharp, S. Kim, J. Jeong, J. Hui, P. Dutta, and D. Culler. Marionette: a tool suite for debugging and scripting of wireless sensor networks. In *Proceedings of IPSN 2006*, pages 416 – 422, 2006.

[75] J. Wu and H. Li. On calculating connected dominating set for efficient routing in ad hoc wireless networks. In *Proceedings of Third International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (DialM)*, pages 7–14, Aug 1999.

[76] Y. Xu, S. Bien, Y. Mori, J. Heidemann, and D. Estrin. Topology control protocols to conserve energy in wireless ad hoc networks. CENS Technical Report 0006, UCLA, January 2003.

[77] S. Yamashita, S. Takanori, K. Aiki, K. Ara, Y. Ogata, I. Simokawa, T. Tanaka, K. Shimada, and H. K. and. A 15x15mm, 1ua, reliable sensor-net module: Enabling application-specific nodes. In *Proceedings of IPSN 2006*, pages 383 – 390, 2006.

[78] B. Yin, H. Shi, and Y. Shang. A two-level topology control strategy for energy efficiency in wireless sensor networks. *International Journal of Wireless and Mobile Computing, to appear.*

[79] B. Yin, H. Shi, and Y. Shang. An efficient single-phase distributed algorithm for constructing connected dominating set in ad hoc networks. In *Proceedings of IEEE AWiN 2005, in conjunction with IEEE Globecom 2005*, 2005.

[80] B. Yin, H. Shi, and Y. Shang. A two-level strategy for topology control in wireless sensor networks. In *Proceedings of the 2005 11th International Conference on Parallel and Distributed Systems (ICPADS 2005)*, 2005.

[81] B. Yin, H. Shi, and Y. Shang. Analysis of energy consumption in clustered wireless sensor networks. In *International Symposium on Wireless Pervasive Computing (ISWPC) 2007*, Februrary 2007.

[82] B. Yin, H. Shi, and Y. Shang. An efficient algorithm for constructing connected dominating set in ad hoc networks. In *Proceedings of Consumer Communications and Networking Conference (CCNC) 2007*, Janurary 2007.

[83] D. Zhou, M.-T. Sun, and T.-H. Lai. A timer-based protocol for connected dominating set construction in IEEE 802.11 multihop mobile ad hoc networks. In *Proceedings of SAINT*, pages 2–8, 2005.

VITA

Bolian Yin is a PhD candidate in Computer Science at the University of Missouri-Columbia, working under the supervision of Professor Hongchi Shi in the Distributed Computing and Sensor Networks (DCSN) Research Laboratory. His research interests include wireless sensor networks and distributed computing. He received his Master of Engineering degree in Computer Science and Engineering from the Beijing University of Aeronautics and Astronautics, China in 2000 and his Bachelor of Engineering degree in Computer Science from Xidian University, China in 1997. Prior to joining the PhD program at the University of Missouri-Columbia, he was a software engineer at Sun Microsystems (China) from 2001 to 2003 and an R&D Engineer at Lucent Technologies (China) from 2000 to 2001. He is a student member of the IEEE.