8-2014

A LOAD-BASED APPROACH TO FORMING A CONNECTED DOMINATING SET FOR AN AD HOC NETWORK

Raihan Hazarika
Clemson University, rhazari@g.clemson.edu

A Load-Based Approach to forming a Connected Dominating Set for an Ad Hoc Network

---

A Thesis
Presented to
the Graduate School of
Clemson University

In Partial Fulfillment
of the Requirements for the Degree
Master of Science
Electrical Engineering

---

by
Raihan Hazarika
August 2014

Accepted by:
Dr. Harlan B. Russell, Committee Chair
Dr. Daniel Noneaker
Dr. Kuang-Ching Wang

Abstract

Efficient routing in mobile ad hoc networks (MANETs) is highly desired and connected dominating sets (CDS) have been gaining significant popularity in this regard. The CDS based approach reduces the search for a minimum cost path between a pair of source and destination terminals to the set of terminals forming the backbone network. Researchers over the years have developed numerous distributed and localized algorithms for constructing CDSs which minimize the number of terminals forming the backbone or which provide multiple node-disjoint paths between each pair of terminals. However none of this research focuses on minimizing the load at the bottleneck terminal of the backbone network constructed by the CDS algorithms. A terminal becomes a bottleneck if the offered traffic load is greater than its effective transmission rate. In this thesis we analyze the load-based performance of a popular CDS algorithm which has been employed in MANET routing and a k-connected k-dominating set (k-CDS) algorithm and compare it with our new centralized algorithm which has been designed to minimize the load at the bottleneck terminal of the backbone network. We verify the effectiveness of our algorithm by simulating over a large number of random test networks.

Acknowledgments

I would firstly like to thank my advisor, Dr. Harlan B. Russell, for his insight, dedication and guidance during the course of my graduate studies at Clemson University. I would also like to thank Dr. Daniel Noneaker and Dr. Kuang-Ching Wang for serving on my thesis committee.

I would also like to take this opportunity to thank my wonderful parents and my sister for their love and encouragement which has enabled me to pursue my dreams and ambitions. Lastly I would like to thank my friends, Siddhartha and Nilim, for their help and support during my stay at Clemson.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Ad Hoc Networks

The ability of a mobile ad hoc network (MANET) to temporarily form a network with mobile terminals equipped with radios has made it very popular in certain applications. Such networks do not require an established or centralized infrastructure as the terminals of the network communicate by coordinating their transmissions and routing the packets in a distributed manner [1]. Distributed transmission scheduling protocols are used for coordinating the packet transmissions among neighboring terminals to avoid collisions while routing protocols determine the path along which a packet is forwarded from the source to the destination terminal.

Ad hoc networks can be quickly deployed and can provide reliable communications in situations where it is difficult to establish any permanent network. Added to this, ad hoc networks have low maintenance cost, require minimal configuration, and are highly robust. The flexibility offered by ad hoc networks has led to their widespread application in the areas of environment monitoring [2], vehicular networks [3] [4], military applications [5] and for setting up emergency communication networks in disaster scenarios [6].

1.2 Virtual Backbones in Ad Hoc Networks

Even though an ad hoc network does not have a physical infrastructure, forming a virtual backbone provides a two-level hierarchical structure to the network [7], [8]. Such a virtual backbone can provide a significant advantage as it reduces the search space for routing in the network. Routing in the network can be constrained to the virtual backbone, minimizing the routing search time as well as reducing the routing table size. In general a dominating set (DS) is a set where every terminal of the network is either in the set or is a neighbor of a terminal in the set. When a DS is connected, it is referred to as a connected dominating set (CDS), i.e., any two terminals in the DS are connected via intermediate terminals of the DS. The CDS has become the preferred method for the construction of the virtual backbone in ad hoc networks.

The earliest research focused on keeping the virtual backbone as small as possible. Using a graph to represent an ad hoc network, the objective is to construct a minimum connected dominating set (MCDS). Distributed algorithms for constructing MCDS have been suggested in several papers such as [9], [8], [10], [11]. Wu *et al.* [9] proposes a localized algorithm for generating a CDS where a marking process is utilized to mark a node if it has two unconnected neighbors. For achieving an approximate MCDS pruning rules are also described in the paper. Dai *et al.* [10] further extends the pruning rules to k-hop neighborhoods for achieving better results. Chen *et al.* [11] also proposes a localized algorithm to construct a CDS for topology maintenance, where a terminal become part of the backbone when two of its neighbors cannot reach each other directly or via one or two terminals already in the backbone.

In order to add robustness and fault tolerance to the virtual backbone, distributed CDS algorithms with k-connectivity are investigated in [12], [13], [14]. The k-connectivity property requires that between any pair of terminals in the backbone there exists at least

k different node-disjoint paths. With k-connectivity, communication will not be disrupted even when upto k-1 paths fail. Along with the multi-path redundancy, k-connectivity also provides load-balancing among the terminals forming the virtual backbone.

1.3 Thesis Outline

In this thesis we analyze the load-based performance of an approximate MCDS algorithm and a *2*-connectivity CDS algorithm. We also propose a new centralized algorithm designed to improve the load-based performance of any approximate MCDS algorithm and we compare its performance with the approximate MCDS and *2*-connectivity CDS algorithms. Chapter 2 provides a description of the network model used in our study. We describe the backbone network and a distributed transmission scheduling algorithm utilized by the terminals forming the backbone. A performance metric is defined to permit comparative analysis of the different CDS algorithms. Chapter 3 gives the pseudocodes for the approximate MCDS algorithm, the *2*-connectivity CDS algorithm, and our new centralized load-based CDS algorithm. Chapter 4 describes the simulation model utilized for analyzing the performance of the three CDS algorithms. Results are provided for two different scenarios of network densities. Chapter 5 presents the conclusion of the thesis as well as an interpretation of our significant findings.

Chapter 2

Network Model

We model an ad hoc network as a unit-disk graph *G(V,E)* where *V* is the set of all the terminals in the network. Each terminal has an equal communication range. Two terminals which are within communication range share a half-duplex, bidirectional link represented as an edge *E* of the graph, and terminals may transmit and receive data packets without error as long as no packet collisions occur. Two terminals are said to be *1-neighbors* if they are within communication range and *2-neighbors* if they are not 1-neighbors and there exists a terminal which is a 1-neighbor to both. The *neighborhood* of a terminal comprises the terminal itself and its 1 and 2-neighbors.

Terminals use omnidirectional antennas to broadcast to all other terminals in range and we assume that the terminals are synchronized to time slot boundaries. A broadcast transmission is successful if it is received by all the terminals that are within range of the transmitter and unsuccessful if a collision occurs at any of the 1-neighbors of the transmitter. A packet collision occurs in two ways: first, if two terminals which are 1-neighbors transmit at the same time and second if two terminals which are 2-neighbors transmit simultaneously causing a collision in all their common 1-neighbor terminals. One approach for constructing a collision free transmission schedule is for each terminal to be assigned

a unique color number with respect to the other terminals in its neighborhood and then assigning each color number a unique transmission slot in a transmission frame.

2.1 The Backbone Network

A subset of the terminals is used for forming the backbone of the network. The backbone can be constructed by using any known algorithm which forms a CDS. Terminals forming the backbone network are the *dominant terminals* of the network. The backbone network constructed by the CDS algorithm ensures that the non-backbone terminals are at 1-hop distance from any dominant terminal. A non-backbone terminal may have multiple dominant terminals that are 1-neighbors. However any non-backbone terminal associates with just one dominant terminal in its 1-neighborhood. In this hierarchical network model, a dominant terminal utilizes Lyui's transmission scheduling algorithm [15] for channel access, while a non-backbone terminal uses a contention-based channel access protocol for forwarding traffic to the associated dominant terminal.

We assume the availability of multiple heterogenous channels to the network and also the ability of software-defined radios (SDRs) to support two transceivers and to operate independently on non-overlapping frequency channels. A dominant terminal uses one of the transceivers for forwarding traffic along the backbone network and the second transceiver for communicating with the associated non-backbone terminals.

2.2 Lyui's Transmission Scheduling Algorithm

For a collision free transmission among the dominant terminals we utilize a transmission scheduling algorithm developed by Lyui [15] where transmission slots are assigned to terminals based on the color number of the terminals. Lyui's algorithm assigns each ter-

Table 2.1: Lyui's candidate transmission slot for the first eight color numbers.

Candidate Transmission Slot Numbers																
Color Number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
2		x		x		x		x		x		x		x		x
3			x				x				x				x	
4				x				x				x				x
5					x								x			
6						x								x		
7							x								x	
8								x								x

minal a color number which is the smallest positive integer that is different from the color numbers of all the terminals in its neighborhood. A terminal k with color number c_k is a candidate for transmission in slots t which satisfy

$$t = c_k + n \times p(c_k) \tag{2.1}$$

or in slots t which satisfy

$$t \bmod p(c_k) = c_k \bmod p(c_k) \tag{2.2}$$

where $p(c_k)$ is the smallest power of two greater than or equal to c_k. For example, Table 2.1 illustrates the set of possible candidates for each slot considering the first eight color numbers. For a given slot t, terminal k uses (2.1) or (2.2) to determine which of the terminals in its neighborhood are candidates for transmission. The set of candidate terminals calculated by terminal k to transmit in slot t is $C(t,k) \subseteq N(k)$, where $N(k)$ is the set of terminals in the neighborhood of terminal k and $m \in C(t,k)$ if $t \bmod p(c_m) = c_m \bmod p(c_m)$. Finally terminal k transmits in slot t if k is a candidate in slot t and k has the largest color number among all the terminals in the neighborhood that are candidates for that particular

slot. Each terminal has a transmission frame of length equal to the smallest power of two greater than or equal to the maximum color number in the terminal's neighborhood. Lyui's algorithm ensures one guaranteed transmission slot per frame for each terminal, and a terminal may be assigned multiple transmission slots depending on the color numbers of the other terminals in its neighborhood. A description of Lyui's slot assignment algorithm and its properties can be found in [16].

2.3 Maximum Stable Throughput

Each terminal in the network (dominant or non-backbone) generates traffic for every other terminal with an equal rate. The dominant terminals are responsible for forwarding traffic along the backbone for each source-destination path that passes through it. We define Λ_i as the number of source-destination paths for which terminal *i* must forward traffic. Let C_i be the total number of slots in terminal *i*'s transmission frame and S_i the number of slots assigned to terminal *i* in its frame. The effective transmission rate for terminal *i* is S_i/C_i packets per slot. The *load-factor* for terminal *i* (LF_i) can be defined as the ratio of the number of source-destination paths for which terminal *i* must forward traffic to its effective transmission rate, i.e.,

$$LF_i = \frac{\Lambda_i \times C_i}{S_i} \tag{2.3}$$

We define the end-to-end throughput as the total rate at which all traffic reaches its destinations. The largest value of end-to-end throughput for which the arrival rate to each terminal is less than or equal to its forwarding rate is the *maximum stable end-to-end throughput*, denoted by Γ. In Appendix A it is shown that the maximum stable end-to-end

throughput for a network of N terminals is

$$\Gamma = \frac{N(N-1)}{Max[LF(i)]_{\forall i}} \tag{2.4}$$

Dominant terminals with high load-factor values have either a large number of source-destination paths flowing through them or only a small number of transmission opportunities per frame. We refer to the dominant terminal with the largest load-factor value as the *bottleneck* terminal. The *maximum load-factor*, i.e., the largest load-factor value of the network, is an indicator of the bottleneck that exists in the network. Minimizing the maximum load-factor value of the network reduces the traffic load at the bottleneck terminal. The LoBaTS protocol [17] alleviates traffic congestion by altering the transmission schedules and providing additional transmission slots to terminals with large load-factor values, thereby increasing their effective transmission rate and minimizing their corresponding load-factor values. However in this research work, we address this problem by reducing the number of terminals that participate in forwarding packets and finding suitable relay terminals to add to the backbone network to reduce the load-factor value of the bottleneck terminal.

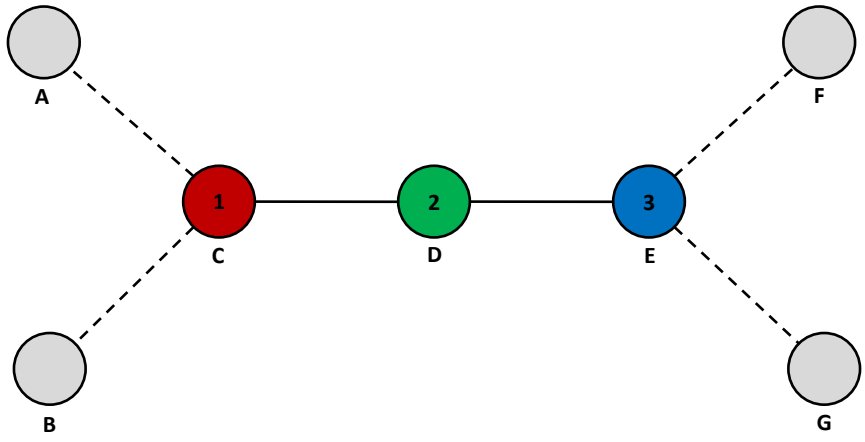

Figure 2.1: Network of 7 terminals. Terminal C, D, and E are the dominant terminals forming the backbone network. Terminals A, B, F, and G are the non-dominant terminals.

Figure 2.1 illustrates a network of 7 terminals. Terminals C, D, and E are the dominant terminals in the network and form the backbone. Terminals A, B, F, and G are the non-dominant terminals. Terminals A and B associate with C while terminals F and G associate with E. The dominant terminals C, D, and E are assigned the color numbers 1, 2 and 3, respectively, and transmission slots based on Lyui's transmission scheduling algorithm. The transmission frame size for this example network is 4 slots, which is the smallest power of 2 greater than the largest color number in the network.

Table 2.2: Load-Factor values of the dominant terminals for the 7 terminal network.

Dominant Terminal	Color	Slots	Λ_i	S_i/C_i	LF_i
C	1	1	12	0.25	48
D	2	2,4	24	0.50	48
E	3	3	12	0.25	48

Terminal C has 4 source-destination paths which originates at C and flow through the backbone network, i.e, from C to the terminals D, E, F, and G. The paths from C to the terminals A and B do not flow through the backbone network. Terminal C has 8 additional source-destination paths, 4 originating at terminal A and the other 4 originating at terminal B, flowing through it and along the backbone network. As such Λ_C = 12; by symmetry Λ_E = 12. Terminal D has 6 source-destination paths that originate at D, and flow through the backbone network. Additionally it has another 18 source-destination paths, 3 originating from each terminal other than D, that flow though it along the backbone network. As such for terminal D, Λ_D = 24. In Table 2.2, the slots assigned, the number of source-destination paths, the effective transmission rate, and the load-factor values for the 3 dominant terminals of the network are shown. Terminal D has twice the number of source-destination paths flowing through it compared to terminals C and E. However it has

twice the transmission rate per slot compared to that of terminals C and E as it is assigned two transmission slots in a four slot frame by Lyui's transmission scheduling algorithm. The load-factor values for all the three dominant terminals in the network are the same and the maximum load-factor value for the network is 48. The maximum stable throughput for this 7 terminal network is 0.875. A similar calculation shows that if all the terminals are in the backbone, the maximum stable throughput is 0.4375, which is half of the previous value.

2.4 Example

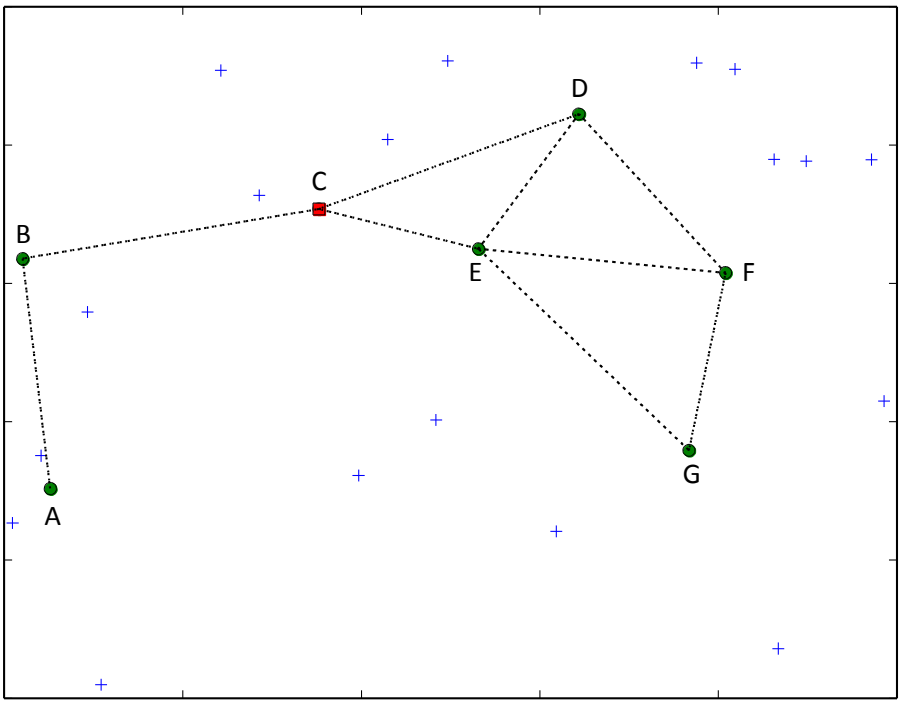

Figure 2.2: A network with 25 terminals. The backbone is constructed by an approximate MCDS algorithm.

Table 2.3: Load-Factor values of the 7 dominant terminals for the 25 terminal network.

Dominant Terminal	Color	Slots	Λ_i	S_i/C_i	LF_i
A	1	1,3,7	84	0.375	224
B	2	2,6	210	0.25	840
C	5	5	276	0.125	2208
D	4	4,8	164	0.25	656
E	1	1	146	0.125	1168
F	3	3,7	116	0.25	464
G	2	2,6	84	0.25	336

Figure 2.2 shows a network of 25 terminals in which 7 terminals form the backbone network. The backbone is constructed using an approximate distributed MCDS algorithm. Table 2.3 lists the color numbers, transmission slot assignments, number of source-destination paths, effective transmission rate, and load-factor values for each of the 7 dominant terminals forming the backbone network. The maximum load-factor value for this 25 terminal network is 2208 and the corresponding maximum stable throughput is 0.27.

In Figure 2.3 we have the same 25 terminal network. Two additional terminals (terminals H and K) are added to the backbone. The color numbers, transmission slot assignments, number of source-destination paths, effective transmission rate, and load-factor values for the 9 dominant terminals is shown in Table 2.4. From the table, it is observed that the maximum load-factor value for the network is 1744. The maximum stable throughput is 0.34

The results indicate that a backbone network constructed by an MCDS algorithm is not ideal for minimizing the traffic congestion in a network. By adding additional terminals in the backbone formed by an MCDS algorithm the throughput for a network can be increased.

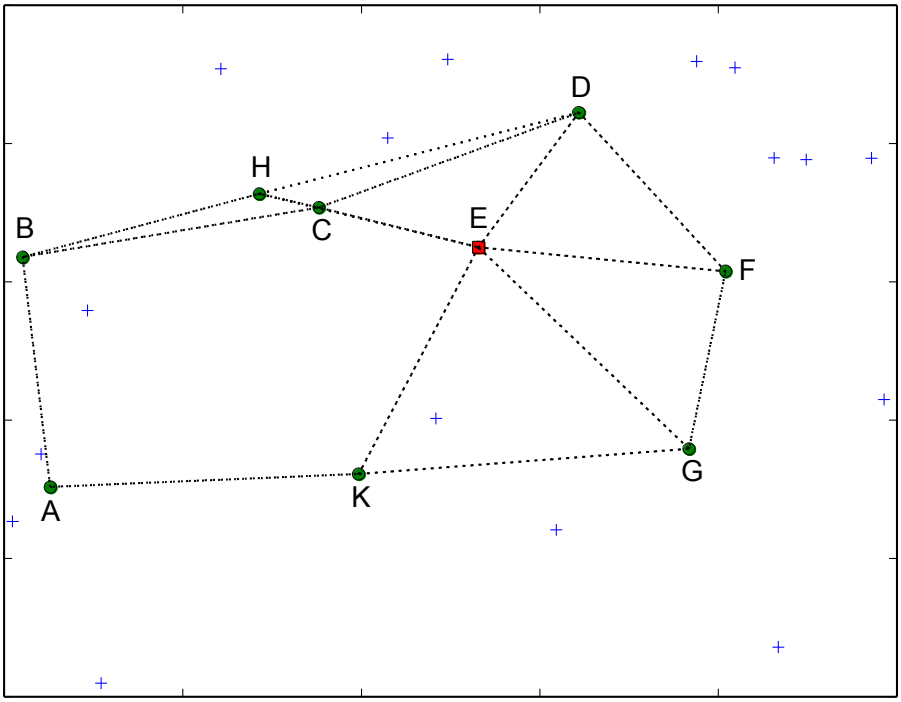

Figure 2.3: A network with 25 terminals. Terminals H and K are added to the backbone constructed by the MCDS algorithm.

Table 2.4: Load-Factor values of the 9 dominant terminals for the 25 terminal network.

Dominant Terminal	Color	Slots	Λ_i	S_i/C_i	LF_i
A	3	3	92	0.125	736
B	2	2	86	0.125	688
C	5	5	66	0.125	528
D	4	4,8	108	0.25	432
E	1	1	218	0.125	1744
F	3	3	66	0.125	528
G	2	2	84	0.125	672
H	6	6	94	0.125	752
K	7	7	142	0.125	1136

Chapter 3

Connected Dominating Set

3.1 Essential Connected Dominating Set

Computing the minimum connected dominating set (MCDS) is an NP-hard problem [18]. Over the years researchers have come up with a number of distributed algorithms for constructing an approximate MCDS. A number of such distributed CDS algorithms have also been applied to MANET routing [19], [20], [21]. In this paper we base our work on the essential connected dominating set (ECDS) algorithm, a distributed CDS algorithm, discussed in detail in Appendix A of the simplified multicast forwarding (SMF) specification [22]. In the ECDS algorithm a terminal requires knowledge of its local neighborhood only. The algorithm elects the dominant terminals of the CDS, thereby forming the backbone of the network. Several election heuristics have been proposed and used for electing the dominant terminals in the ECDS and other distributed CDS algorithms. For this study, we use the 1-neighborhood degree or *nodal degree* as the election heuristic. In the event of a tie, i.e., if two terminals have the same nodal degree, then the terminal having the greater id value is given priority. The ECDS selection algorithm is summarized in the following steps:

1. A terminal (T) calculates its nodal degree and initializes a set containing the ids of the terminals in its neighborhood.

2. If the nodal degree of T is less than 2, then T does not elect itself as a dominant terminal.

3. If T's nodal degree is greater than all the terminals in its neighborhood, then T elects itself as a dominant terminal.

4. If T is not elected as a dominant terminal, initialize a set containing all the terminals in its neighborhood and mark them as unvisited.

5. Find the terminal N_{1-max} which has the highest nodal degree in T's 1-neighborhood.

6. Initialize a queue (Q) containing N_{1-max} and mark terminal N_{1-max} as visited.

7. While the Q is not empty, remove a terminal (X) from the head of the Q. Mark each 1-neighbor (Z) of X as visited. If the nodal degree of Z is greater than that of T, then push Z into Q.

8. If any of T's 1-neighbors remain unvisited, then T elects itself as a dominant terminal. Otherwise T does not participate in the backbone network.

Figure 3.1 illustrates the dominant terminals selected by the ECDS algorithm for a 100 terminal network. The terminals are placed randomly in a square area of 1000 by 1000 with a uniform communication range of 200. Of the 100 terminals in the network, 29 are dominant terminals (forming the backbone network). In the figure, the dominant terminals forming the backbone network are marked with a green 'o' while the non-dominant terminals are shown with a blue '+'.

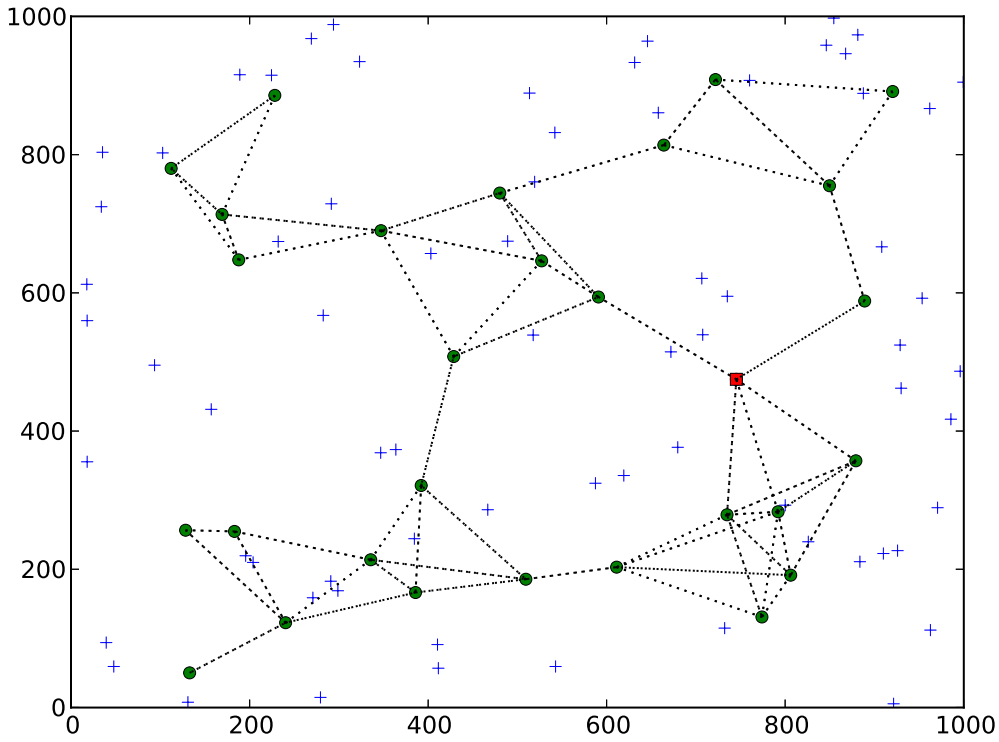

Figure 3.1: A network of 100 terminals with the communication range equal to 200. The backbone is formed by the distributed ECDS algorithm.

3.2 2-CDS

The availability of multiple node-disjoint paths between any source-destination terminal pair can be utilized to reduce the bottleneck. With an appropriate routing metric, the availability of additional paths allows the network to distribute its load. In [12], four localized algorithms are investigated for constructing a *k*-connected *k*-dominating set (*k*-CDS) backbone network. A network is *k*-connected if any two terminals in the network are connected via *k* terminal-disjoint paths. Also a network is *k*-dominating if every non-backbone terminal has at least *k* dominant terminals in its 1-neighborhood. Such a *k*-connected *k*-dominating set is simply referred to as *k*-CDS.

We use the color based *k*-CDS construction algorithm for our analysis, which is a hybrid of probabilistic and deterministic approaches. The algorithm provides a general framework for converting any CDS algorithm into a *k*-CDS algorithm. The algorithm can be summarized as follows:

1. The first step is probabilistic; each terminal randomly selects a color c_v ($1 \leq c_v \leq k$). The idea is to partition the network into *k* disjoint sub-networks.

2. The second step is deterministic; for each of the *k* sub-networks, any traditional CDS algorithm is applied for creating a backbone which covers the original network.

3. The final backbone network is the union of all the *k* colored backbone networks.

Figure 3.2 illustrates the backbone generated by the 2-CDS algorithm. Here we partition the network into two disjoint sub-networks. Terminals with even number ids belong to one half while the odd numbered terminals belong to the other half. In both the sub-networks, a backbone is constructed using the ECDS algorithm. Combining the two backbones formed in each sub-network we obtain the desired 2-CDS backbone network. In the figure we have the same 100 terminal network in a 1000 by 1000 square area. The

100 terminals are divided into two sub-networks; terminals marked with a '+' belong to one sub-network while the terminals marked with a 'x' belong to the other sub-network. Of the 100 terminals in the network, 42 terminals are used for constructing the backbone network and are represented in the figure by the green 'o' symbols.

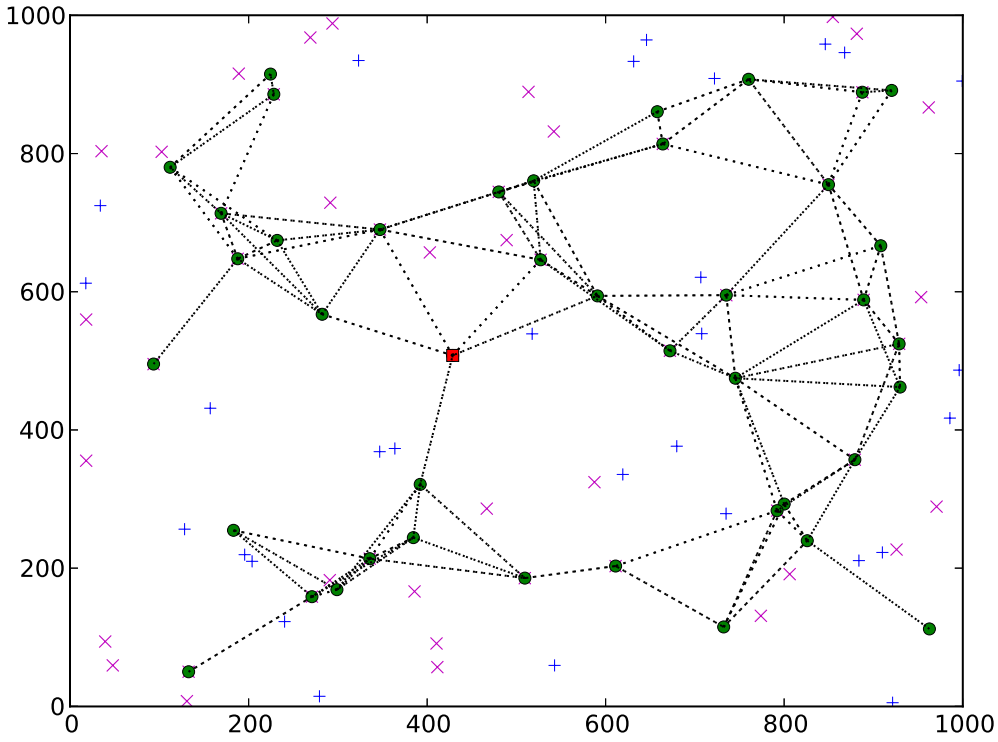

Figure 3.2: A network of 100 terminals where the communication range is equal to 200. The backbone is formed by the color based 2-CDS algorithm.

3.3 Load-Based CDS

Our new load-based CDS (LoB-CDS) algorithm extends the existing ECDS algorithm or for that matter any algorithm which can approximately generate an MCDS. The algorithm works in an iterative manner and attempts to minimize the maximum load-factor value of the network by finding an appropriate relay terminal in the 1-neighborhood of a terminal with a large load-factor value and adding the relay to the backbone network. By adding suitable relay terminals to the neighborhood of the terminals with large load-factor values and integrating the algorithm with an appropriate routing metric, the maximum load-factor value of the network can be reduced. The LoB-CDS algorithm is illustrated in Figure 3.3. Next we summarize the main steps of the algorithm:

1. Form the backbone network using any traditional MCDS algorithm.

2. Assign colors to the dominant terminals of the network; compute their transmission slots per frame based on Lyui's transmission scheduling algorithm.

3. Each dominant terminal computes its load-factor value which is the ratio of the number of source-destination paths for which it must forward traffic to its effective transmission rate. Initialize Fail_Counter = 0.

4. The dominant terminals are placed in a queue (Q) in decreasing order of its load-factor values. The head of the Q holds the terminal with the highest load-factor value. The current max load factor value ($MaxLF$) of the network is noted.

5. While the Q in not empty, remove a terminal (T) from the head of the Q. Mark all 1-neighbors of T as unvisited.

6. Pick a terminal (X), one at a time, from the 1-neighbors of T and mark X as visited. Terminal X is temporarily taken as a dominant terminal and assigned a color number

and a transmission slot based on Lyui's algorithm. Compute the load-factors of the dominant terminals. Also compute the max load-factor value ($MaxLF_{T_X}$) of the network and store the value with the terminal id of X.

7. Repeat step 6 for every 1-neighbor of T not visited. Find the terminal (Z) for which the network had minimum max load-factor value ($MaxLF_T$).

8. If $MaxLF_T < MaxLF$ (computed in step 4), then Z is permanently accepted as a dominant terminal. If however $MaxLF_T \geq MaxLF$, then increment the failure counter. Go to step 5.

9. The algorithm exits if the failure counter is equal to three (i.e. after 3 attempts a suitable relay terminal is not found which could minimize the max load-factor value of the network). Otherwise, goto step 5.

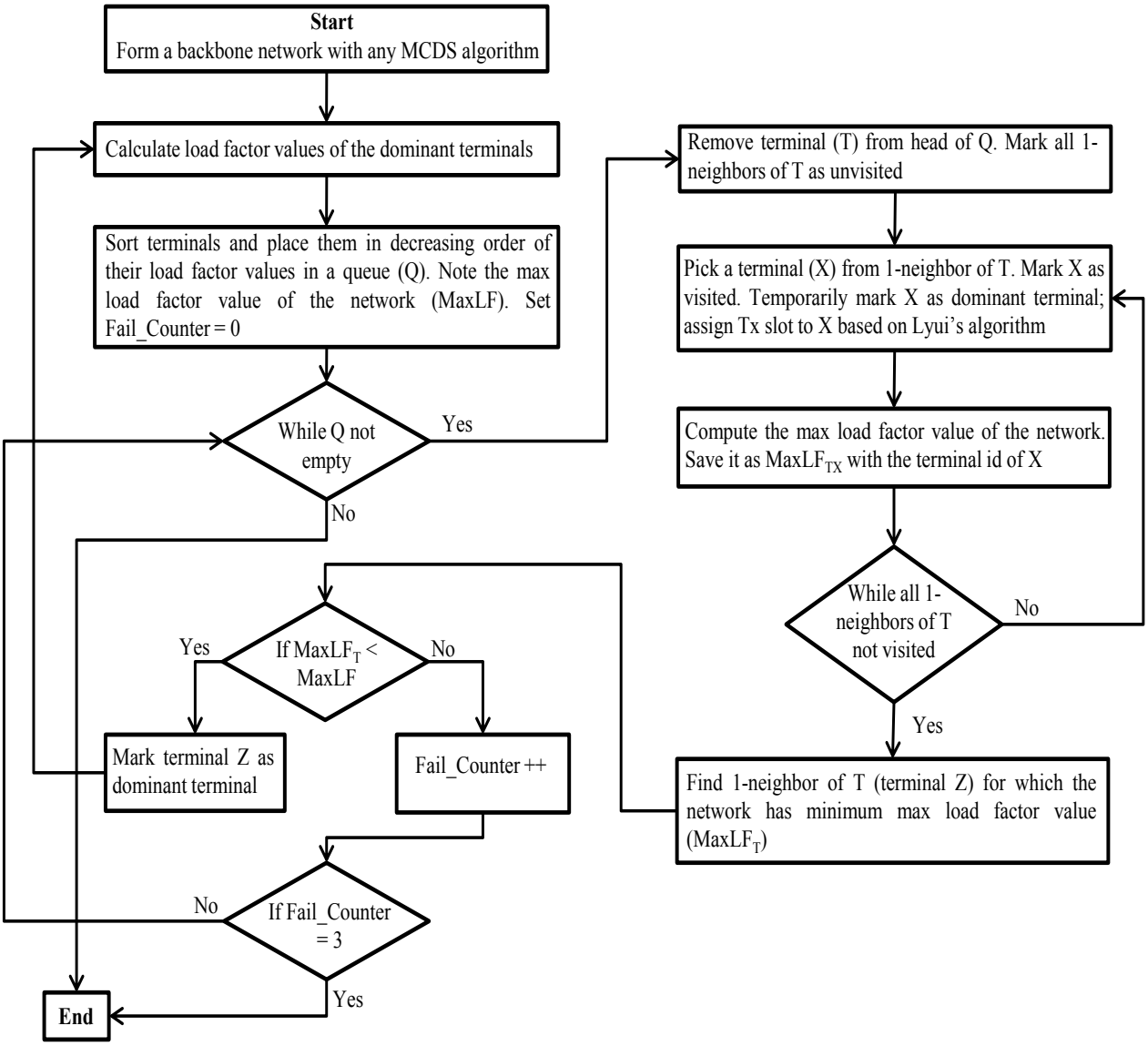

Figure 3.3: Outline of the load-based connected dominating set algorithm

In Figures 3.4, 3.5, and 3.6, the backbone constructed by the LoB-CDS algorithm for the same 100 terminal network using three different routing metrics is illustrated. As the LoB-CDS algorithm depends on the max load-factor values calculated at each iteration, utilizing different routing metrics leads to different backbone networks. The three different routing metrics which we have implemented with the LoB-CDS algorithm in the following figures are discussed in Chapter 4 in Section 4.2.

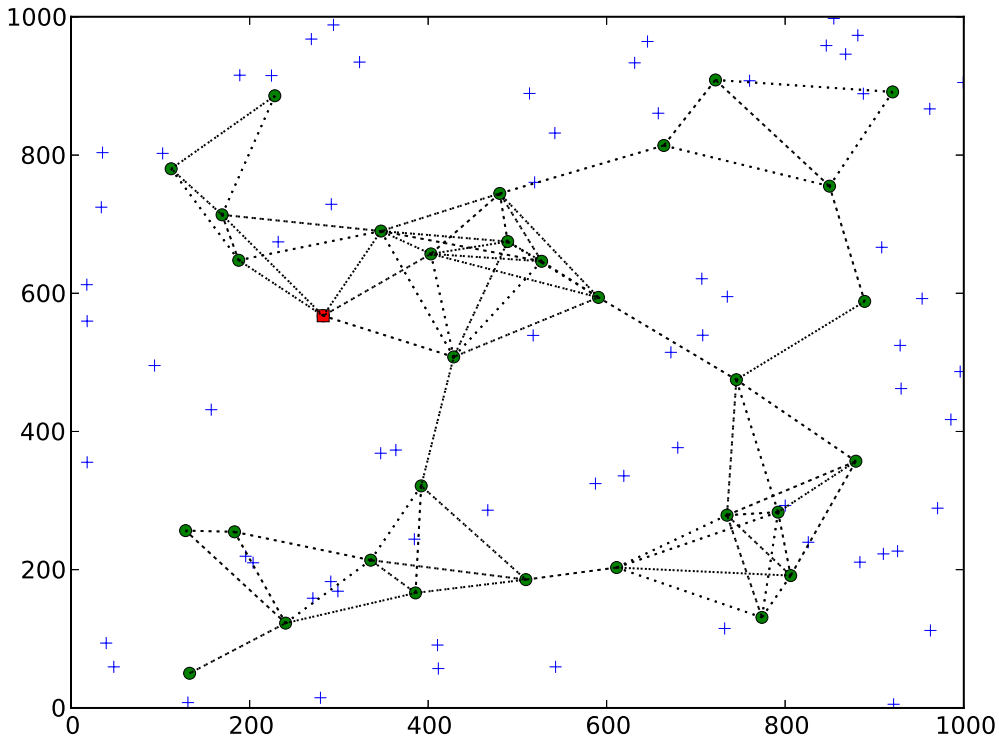

Figure 3.4: A network of 100 terminals with the communication range equal to 200. The backbone is formed by the LoB-CDS algorithm using the min-hop routing metric.

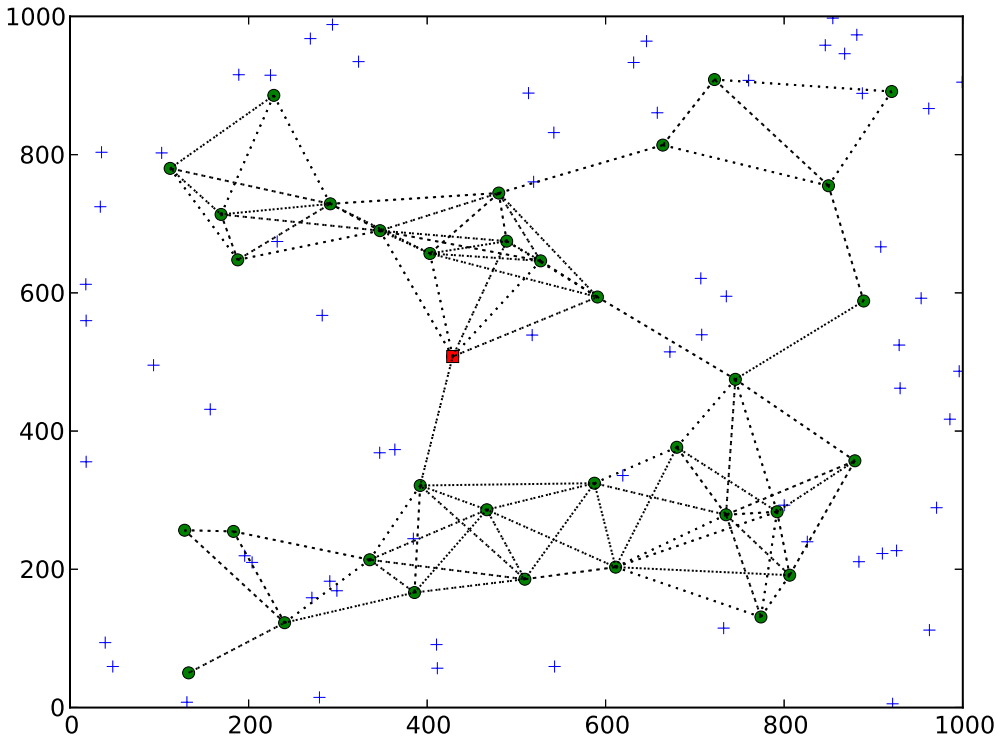

Figure 3.5: A network of 100 terminals with the communication range equal to 200. The backbone is formed by the LoB-CDS algorithm using the forwarding-rate routing metric.

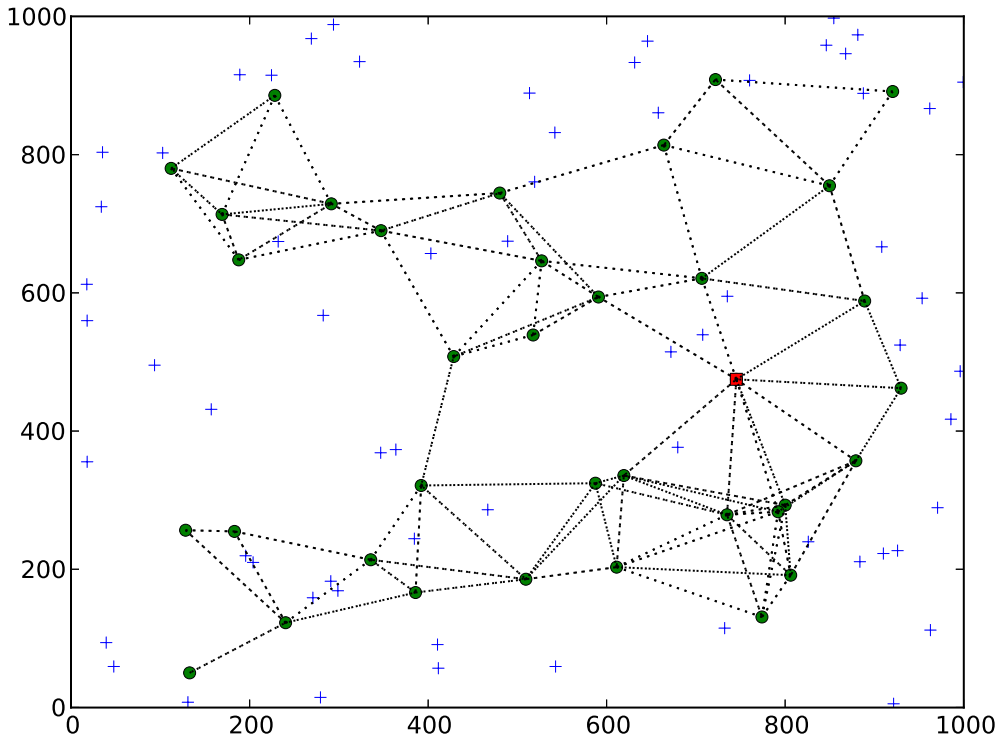

Figure 3.6: A network of 100 terminals with the communication range equal to 200. The backbone is formed by the LoB-CDS algorithm using the load-factor routing metric.

Chapter 4

Simulation Model and Results

We use a simulation program to investigate the performance of the LoB-CDS algorithm and compare its performance with that of the ECDS and 2-CDS algorithms. For each simulation, a random topology is created and a subset of the terminals are selected to form the backbone network using one of the three CDS algorithms discussed in the previous chapter. The dominant terminals forming the backbone network are assigned transmission slots based on Lyui's transmission scheduling algorithm and using appropriate routing metrics, the minimum cost path between all source-destination terminals are calculated. Based on the minimum cost paths that are computed, a count of the number of paths that each terminal must forward is found and the corresponding load-factor value calculated. The performance of the particular CDS algorithm for a specific network scenario is measured with the maximum stable end-to-end throughput which is based on the maximum load-factor value.

4.1 Simulation Model Description

Terminals are placed at random locations in a 1000 by 1000 square grid with a uniform distribution and all terminals have the same communication range. Two different scenarios are simulated. For scenario 1 there are 100 terminals and the communication range is set between 200 and 450. For scenario 2, the number of terminals is increased to 200 and the communication ranges that are examined vary from 150 to 450. For scenario 1, the minimum communication range of 200 ensures with a high probability that the network is connected, i.e., every terminal can reach every other terminal via the backbone network. Also the density of the network, i.e., the ratio of the number of terminals to the area for scenario 1 is $\frac{1}{100^2}$. For scenario 2, the minimum communication range of 150 gives with a high probability a connected network and the network density is $\frac{2}{100^2}$. For each CDS algorithm, scenario, and communication range, 500 randomly generated topologies are simulated and the performance measures averaged.

For each simulation run, the backbone network is first constructed and then using Lyui's transmission scheduling algorithm the dominant terminals forming the backbone network are assigned transmission slots. A non-dominant terminal of the network associates with one of the dominant terminals in its 1-neighborhood. A dominant terminal maintains a count of the number of non-dominant terminals that associates with it. Sequentially visiting each dominant terminal in its 1-neighborhood, a non-dominant terminal associates with a dominant terminal with the smallest count. For a non-dominant terminal, the dominant terminal to which it associates becomes its *parent terminal*. Each dominant terminal in the network computes the minimum cost path to every other dominant terminal in the network using one of three routing metrics. For the paths between non-dominant terminals, the minimum cost path between the associated parent terminals is utilized. Traffic is modeled with a fixed rate, full duplex flow between each pair of terminals. We set

the traffic rate for each source-destination terminal pair to be equal in order to focus on the bottleneck terminals of the network. Each dominant terminal counts the number of source-destination paths for which it must forward traffic to another dominant terminal. At the end of each simulation run, each dominant terminal calculates its load-factor value by computing the ratio between the count of source-destination flows it must forward to its effective transmission rate using Equation 2.3 from Chapter 2.

4.2 Routing Metrics

Three different routing metrics are utilized in the simulations when computing the minimum cost paths.

- **Min-Hop Routing Metric** : Each communication link is assigned a weight equal to one. We denote results from simulations using this routing metric with the label min.

- **Forwarding-Rate (FR) Routing Metric** : The link weights are inversely proportional to the effective forwarding rate of the associated terminals. For terminal *i*, C_i is the total number of slots in terminal *i*'s transmission frame and S_i is the number of slots assigned to this terminal. The link weight is given by

$$FR_i = \frac{C_i}{S_i} \tag{4.1}$$

- **Load-Factor (LF) Routing Metric** : Let $\Lambda_{i_{n-1}}$ be the running count of the number of source-destination paths for which terminal *i* needs to forward traffic after source terminal *n-1* computes its minimum cost paths to all destination terminals. When source terminal *n* computes its minimum cost paths to all destination terminals, the

link weight for terminal *i* is updated as

$$LF_{i_n} = \frac{\Lambda_{i_{n-1}} \times C_i}{S_i} \tag{4.2}$$

For the initial value of the Load-Factor Routing Metric (LF_{i_1}) at terminal i, Λ_{i_0} is taken as 1. Therefore for the initial case, $LF_{i_1} = FR_i$.

The higher the maximum load-factor value of a network, the greater is the load experienced at the bottleneck terminal. We use the maximum stable end-to-end throughput as our performance metric for the simulations. Simulation results comparing the performance of the LoB-CDS algorithm to that of the ECDS and 2-CDS algorithms for each of the three different routing metrics are given in the next section.

4.3 Simulation Results

The first set of investigations consider scenario 1 with 100 terminals. First, we compare the performance of our new LoB-CDS algorithm to the ECDS algorithm. Recall, our algorithm first builds the backbone using the ECDS algorithm and then iteratively attempts to improve the network performance by adding terminals to the backbone. Next, we compare the network performance of our algorithm to the 2-CDS algorithm described in Section 3.2.

Results for the LoB-CDS and ECDS algorithms are shown in Figure 4.1. When the minimum-hop routing metric is employed our LoB-CDS algorithm does not significantly improve the maximum stable throughput for many values of the communication range. Even though additional terminals are added to the backbone, the load at the bottleneck is often not reduced because the traffic is not routed away from the bottleneck terminal.

However, for either of the two other routing metrics, we see that significant improvement in the maximum stable throughput is achieved at all values for the communication range.

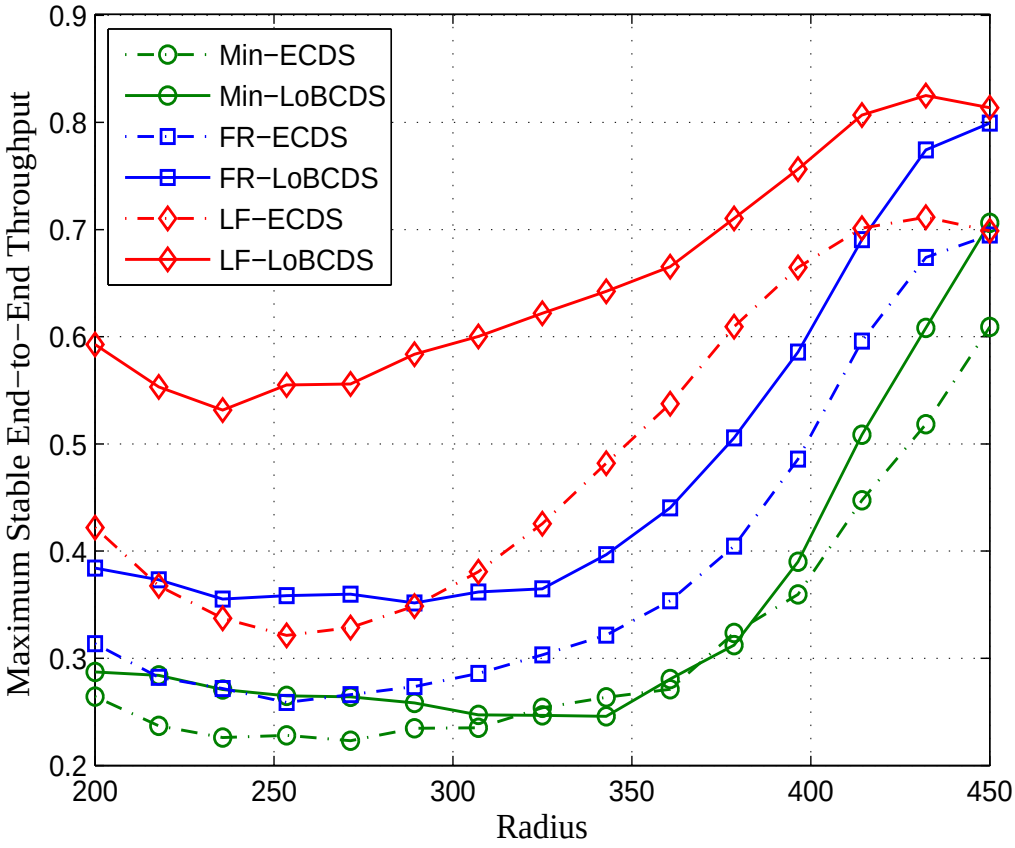

Figure 4.1: LoB-CDS versus ECDS for scenario 1 with 100 terminals.

From Figure 4.1 it can be seen that with the min-hop routing metric, the maximum performance improvement for the LoB-CDS algorithm over the ECDS algorithm is 19% at a communication range of 218. However the performance is worse at a communication range of 342. The forwarding-rate routing metric offers a maximum performance improvement of 38% at a communication range of 253 and a minimum performance improvement of 14% at a communication range of 432. With the load-factor routing metric, the maximum performance improvement observed is 72% at a communication range of 253 and a

minimum improvement of 13% is observed at a communication range of 396.

As a further point of comparison, the results reported in [23] consider the same network model with 100 terminals and a communication range equal to 200. In that model all terminals utilize transmission scheduling, and thus can be considered as backbone terminals. Minimum hop routing is employed, and it is shown that the maximum stable throughput is 0.246.

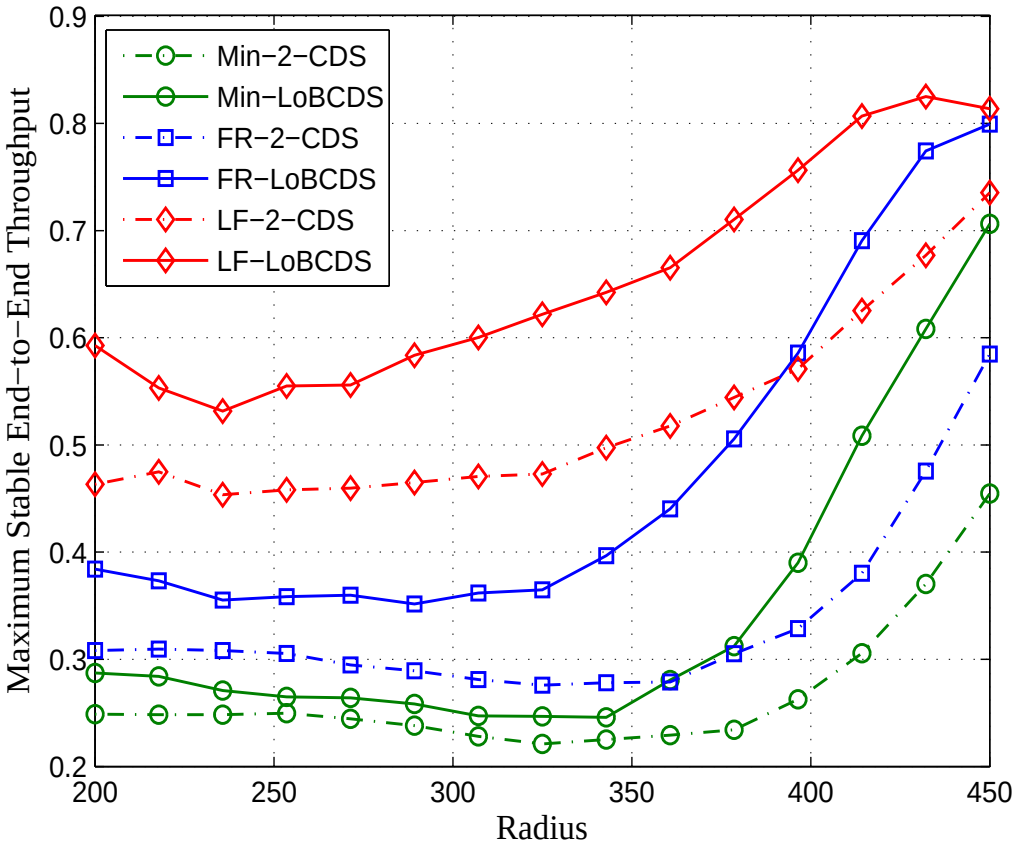

Figure 4.2: LoB-CDS versus 2-CDS for scenario 1 with 100 terminals.

Comparison between our LoB-CDS and the 2-CDS algorithm for the three different routing metrics is shown in Figure 4.2. The results for our LoB-CDS algorithm are the same as shown in Figure 4.1. With the min-hop routing metric, the maximum performance im-

provement observed is 66% at a communication range of 414 and a minimum improvement of 6% is observed at a communication range of 253. The forwarding-rate routing metric offers a maximum performance improvement of 81% at a communication range of 414. With the same routing metric, a minimum performance improvement of 15% is achieved at a communication range of 235. With the load-factor routing metric the maximum and minimum performance improvements are 32% and 10% at a communication range of 396 and 450, respectively.

From the investigation shown in Figures 4.1 and 4.2, it is clear that the routing metric plays a significant role in exploiting the addition of terminals to the backbone network. The LoB-CDS, ECDS and 2-CDS algorithms all show improved throughput performance when using the forwarding-rate and load-based routing metrics. The LoB-CDS algorithm shows improved performance over both the ECDS and 2-CDS algorithms for all communication ranges for both the forwarding-rate and the load-factor routing metrics. When utilizing the min-hop routing metric, the LoB-CDS algorithm shows improved performance compared to the 2-CDS algorithm but only at larger communication ranges. While modest improvements are achieved with the minimum-hop routing metric, the ability of the other routing metrics to move traffic off a route that has a bottleneck is needed to take advantage of the terminals that are added to the backbone.

The second set of simulation results utilize scenario 2 with 200 terminals. The greater density in this scenario increases the average number of non-dominant terminals that associate with a dominant terminal which significantly increases the load-factor values. Overall, the maximum stable throughput decreases compared to scenario 1. Also the LoB-CDS algorithm provides somewhat larger gains over the ECDS algorithm in denser networks when employing the forwarding-rate or load-factor routing metrics.

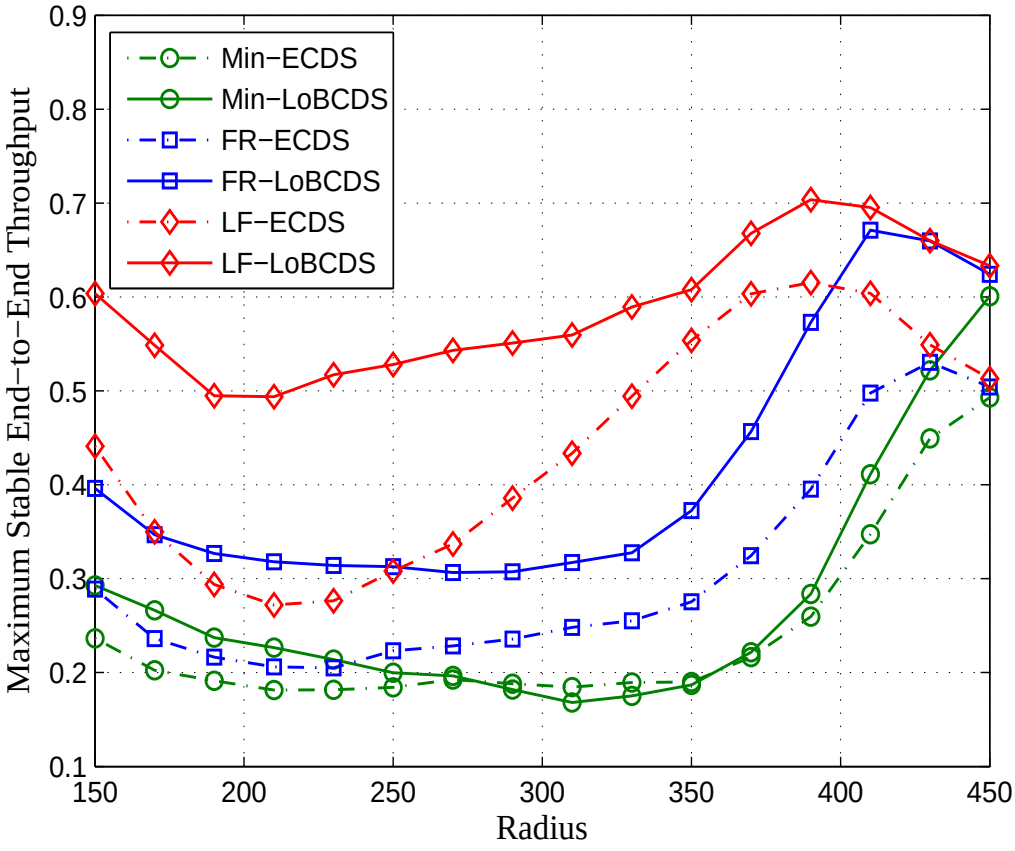

Figure 4.3: LoB-CDS versus ECDS for scenario 2 with 200 terminals.

Figure 4.3 compares the performance of the LoB-CDS algorithm with that of the ECDS algorithm with the three different routing metrics. With the min-hop routing metric, LoB-CDS shows a maximum performance improvement of 31% over ECDS at a com-

munication range of 170. However at a communication range of 310, LoB-CDS shows a decrease in performance of -9% over ECDS. The forwarding-rate routing metric provides a maximum performance improvement of 54% at a communication range of 210 and a minimum improvement of 23% at a communication range of 450. With the load-factor routing metric, a maximum performance improvement of 86% is observed at a communication range of 230 and a minimum improvement of 9% at a communication range of 350.

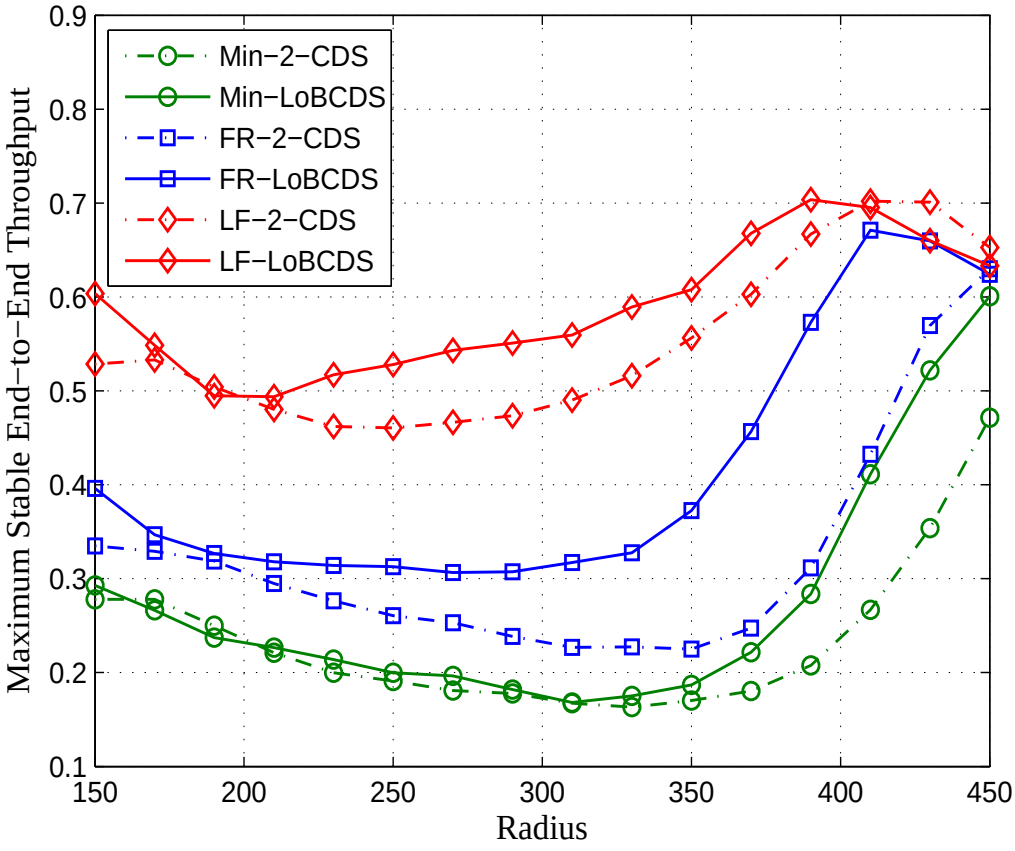

Figure 4.4: LoB-CDS versus 2-CDS for scenario 2 with 200 terminals.

Figure 4.4 compares the performance of the LoB-CDS algorithm with that of the 2-CDS algorithm with the three different routing metrics. The LoB-CDS algorithm offers a maximum performance improvement of 54% over the 2-CDS algorithm with the min-

hop routing metric and at a communication range of 410 and a minimum of -5% at a communication range of 190. The forwarding-rate routing metric provides a maximum performance improvement of 84% at a communication range of 370 and a minimum of -1% at a communication range of 450. With the load-factor routing metric, the maximum performance improvement observed is 16% at a communication range of 270 and the worst is observed at a communication range of 430 where the gain is -6%.

For scenario 2, the maximum stable throughput performance of the LoB-CDS algorithm is compared with that of the ECDS and 2-CDS algorithms with the three different routing metrics, shown in Figures 4.3 and 4.4. Again it can be observed from the figures that with the forwarding-rate and load-factor routing metrics, the maximum stable end-to-end throughput for any network can be significantly improved. Also from Figure 4.3, it can be seen that the LoB-CDS algorithm outperforms the ECDS algorithm over the entire communication range when implementing the forwarding-rate as well as the load-factor routing metric. With the min-hop routing metric, LoB-CDS provides a performance improvement over ECDS only at the lower and higher communication ranges. Figure 4.4 shows that LoB-CDS provides better performance than 2-CDS over the entire communication range when utilizing the forwarding-rate routing metric. When using the min-hop routing metric, LoB-CDS shows an improved performance only at higher communication ranges while with the load-factor routing metric, LoB-CDS only shows a modest performance improvement over 2-CDS between the communication range of 200 to 400.

From the results it can be observed that the LoB-CDS algorithm provides a better throughput performance in most situations compared to the ECDS and 2-CDS algorithms with the three different routing metrics. The LoB-CDS algorithm works in an iterative fashion and attempts to minimize the maximum load-factor value of the network at each iteration. As shown in Equation 2.4 from Chapter 2, the maximum stable end-to-end throughput is inversely proportional to the maximum load-factor value of the network. By minimizing

the maximum load-factor value of the network at each successive iteration, the LoB-CDS algorithm correspondingly increases the maximum stable throughput of the network.

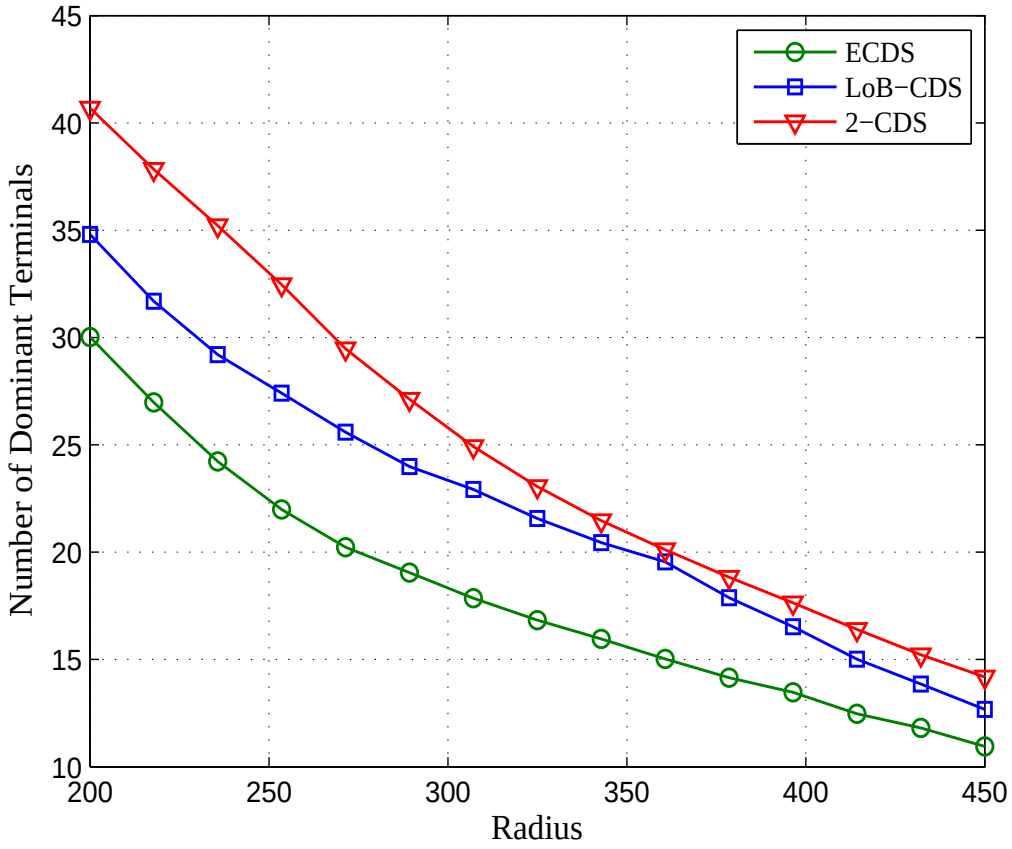

Figure 4.5: Number of dominant terminals at different transmission radius values for scenario 1 with 100 terminals.

Figures 4.5 and 4.6 show the average number of terminals required for constructing the backbone network at different communication ranges for scenario 1 and 2, respectively. From the figures it is apparent that as the communication range increases, the size of the backbone network decreases. In the simulations, the LoB-CDS algorithm is developed over the ECDS algorithm; hence the figures show the number of additional terminals added on average to the backbone formed by the ECDS algorithm for obtaining the LoB-CDS back-

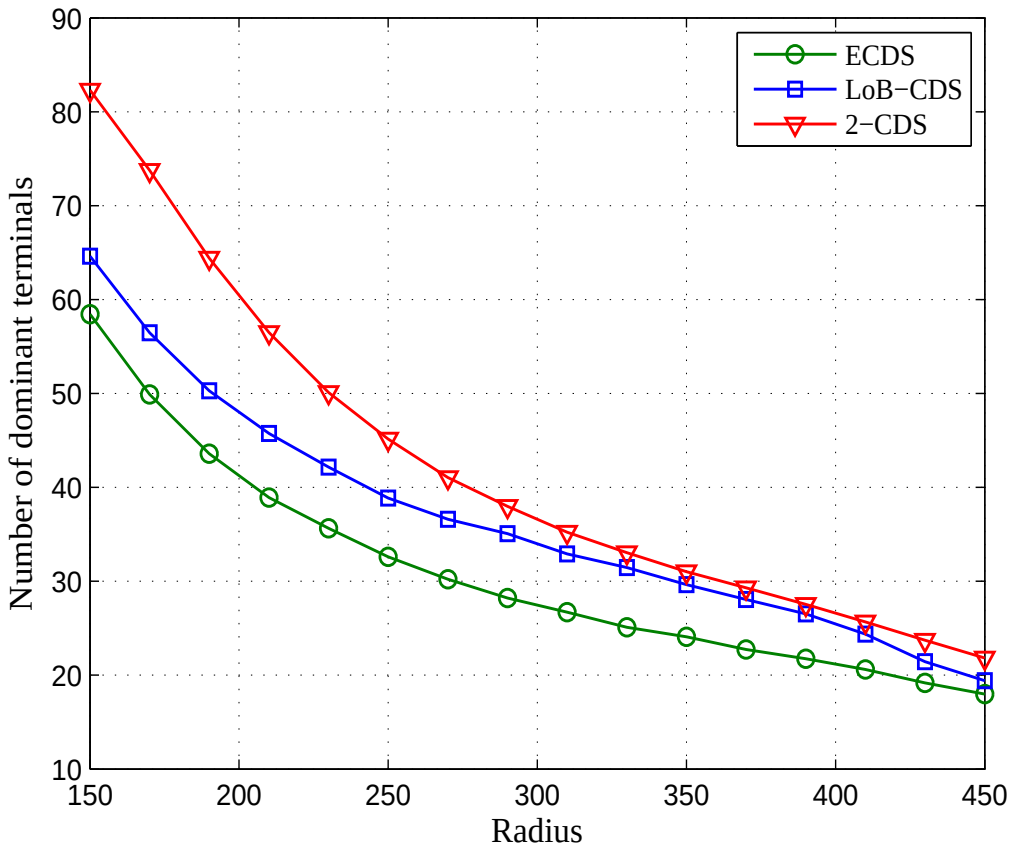

Figure 4.6: Number of dominant terminals at different transmission radius values for scenario 2 with 200 terminals.

bone at the different communication ranges. For scenario 1, it can be seen from Figure 4.5 that for constructing the LoB-CDS backbone network, the maximum number of terminals added on average to the backbone formed by ECDS is 6. Figure 4.6 shows that for scenario 2, the maximum number of terminals added on average is 8 over the entire communication range. The figures also show that a smaller number of terminals are required on average for constructing the backbone network with the LoB-CDS algorithm than with the 2-CDS algorithm particularly at the smaller communication ranges.

The primary focus of the investigations reported in this thesis is on the performance of the scheduled channel-access protocols at the backbone network. The performance of the contention-based channel-access protocol that is utilized by the non-backbone terminals to forward traffic to the dominate terminals in not included in our analysis. In Appendix B we present an approximate analysis of the relative load that must be handled by the contention-based channel-access protocol. We show that the contention-based traffic load is smaller than the traffic load that must be handled by the bottleneck terminal of the backbone network for nearly all of the scenarios that we have examined. Only for scenarios in which the number of non-backbone terminals associated with a backbone terminal is large is it likely that contention from forwarding traffic between the dominant and the associated non-backbone terminals will be a significant factor. It is a topic for future work to account for the impact of the contention-based channels on the selection of terminals to include in the backbone.

Chapter 5

Conclusion

In this research work we analyze the load-based performance of a popular CDS algorithm whch has been implemented in MANET routing and its 2-connectivity variant and compare its performance with a new centralized algorithm designed to improve the overall load-based performance of the network. The results indicate that for a two-level hierarchical network the backbone constructed by a CDS algorithm which approximates an MCDS is not the best choice for minimizing the bottleneck in the network. Such minimal backbone networks put a greater load on the bottleneck terminals. The 2-connectivity CDS algorithm provides redundancy and alternate routes which can be utilized with appropriate routing metrics to achieve load-balancing. However the 2-connectivity CDS algorithm requires a greater number of terminals compared to our new LoB-CDS algorithm for constructing the backbone network. A greater number of terminals in the backbone network reduces the effective transmission rate of the dominant terminals which increases the load-factor value of the network. Also our new LoB-CDS algorithm offers better load-based performance compared to the 2-connectivity CDS algorithm especially for a sparse network.

Appendix A

Development of the maximum stable end-to-end throughput follows the approach originally presented in [23].

Table A.1: A summary of the symbols used and their definitions.

Symbol	Definition
Γ	Maximum stable end-to-end throughput
C_i	Total number of transmission slots in terminal i's transmission frame.
S_i	Number of transmission slots assigned to terminal i in its frame.
Λ_i	Count of the number of source-destination paths for which terminal i needs to forward traffic.

We define γ as the end-to-end throughput for a network, i.e., the total rate at which all traffic reaches its destinations and λ as the average traffic rate on each source-destination path. For a network of *N* terminals, the number of source-destination terminal pairs is *N(N-1)*. We set the traffic rate for each source-destination terminal pair to be equal in order to focus on the bottleneck caused by the network topology. We can now write λ as

$$\lambda = \frac{\gamma}{N(N-1)} \tag{A.1}$$

The average rate of incoming traffic at terminal i can be expressed as

$$\lambda\Lambda_i = \frac{\Lambda_i\gamma}{N(N-1)} \tag{A.2}$$

The effective transmission rate for terminal i is S_i/C_i packets per slot. A terminal is said to be stable if the average rate of incoming traffic is less than its effective transmission rate,

i.e., for terminal i to be stable

$$\lambda \Lambda_i < \frac{S_i}{C_i} \tag{A.3}$$

Using Equation 2.3 from Chapter 2, Equation A.3 can be written as

$$\lambda < \frac{1}{LF_i} \tag{A.4}$$

Combining Equations A.1 and A.4 we have,

$$\gamma < \frac{N(N-1)}{LF_i} \tag{A.5}$$

Now Γ is the maximum stable throughput for the network, i.e., the largest value of γ for which arrival rate is less than or equal to its forwarding rate for all terminals in the network. Therefore,

$$\Gamma = \frac{N(N-1)}{Max[LF(i)]_{\forall i}} \tag{A.6}$$

Appendix B

In this appendix we examine the traffic load that must be handled by the non-dominant terminals and compare it to the traffic load handled by the bottleneck terminal. Let M_i be the number of non-dominant terminals that associate with the dominant terminal *i*. For a network of *N* terminals the average *uplink-downlink traffic* rate from the non-dominant terminals that associate with the dominant terminal *i* can be written as

$$ClientTraffic_i = 2(N-1)M_i\lambda \tag{B.1}$$

Therefore, the maximum average uplink-downlink traffic at any dominant terminal in the network is

$$Max[ClientTraffic_i]_{\forall}i = 2(N-1)M_{max}\lambda \tag{B.2}$$

where M_{max} is the maximum number of non-dominant terminals that associate with any dominant terminal of the network.

The average rate that a dominant terminal *i* forwards traffic on the backbone network is given by $\lambda\Lambda_i$. Therefore, the maximum average rate that any dominant terminal can forward traffic on the backbone network can be expressed as $Max[\lambda\Lambda_i]_{\forall}i$. The bottleneck of a network will exist in the backbone, and not in the uplink-downlink channels of a dominant terminal if

$$Max[\lambda\Lambda_i]_{\forall}i > Max[ClientTraffic_i]_{\forall}i \tag{B.3}$$

which can be written as

$$Max[\Lambda_i]_{\forall}i > 2(N-1)M_{max} \tag{B.4}$$

For the simulation results shown below, we use the ECDS algorithm for forming the backbone network with the forwarding-rate routing metric. For each scenario and communication range, 500 random generated topologies are simulated and the performance measures averaged. In Figure B.1, the red line denotes the $Max[\Lambda_i]_{\forall}i$ and the green line denotes $2(N-1)M_{max}$ over the different communication ranges for scenario 1. From the figure it can be observed that the network congestion is more severe at the backbone network than at any dominant terminal due to uplink-downlink traffic from the associated non-dominant terminals for all communication ranges below 400.

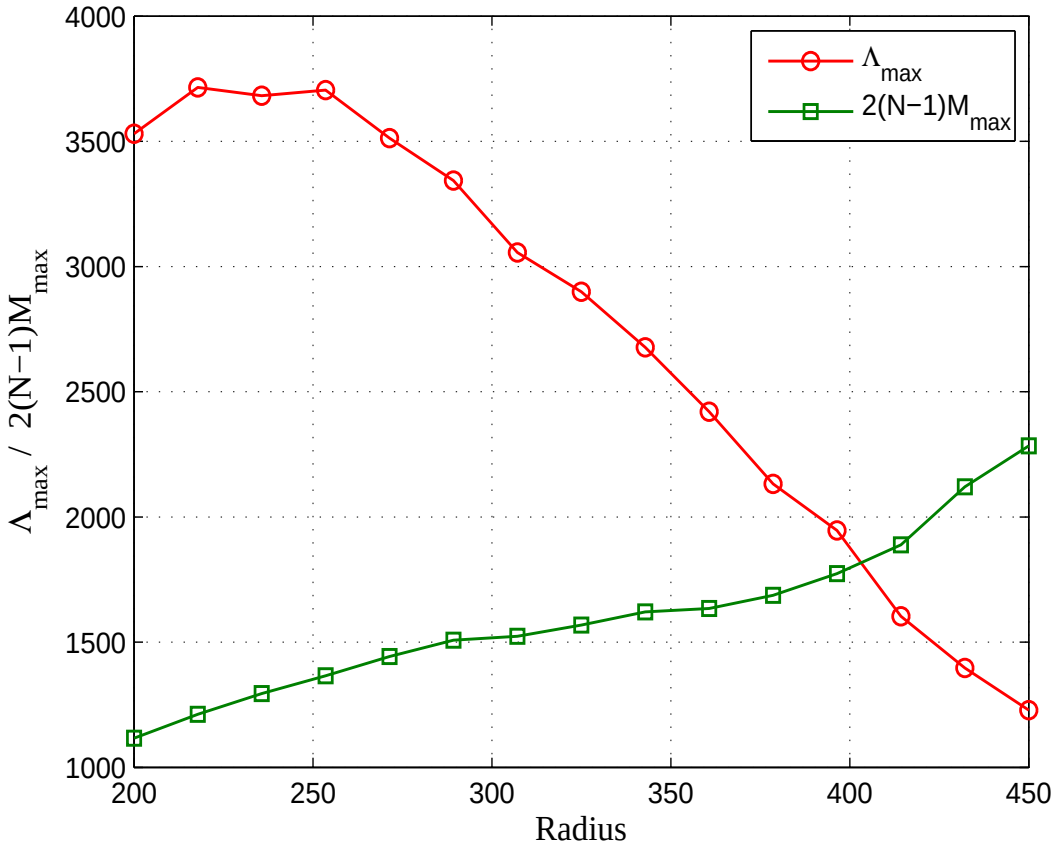

Figure B.1: $Max[\Lambda_i]_{\forall}i$ versus $2(N-1)M_{max}$ for scenario 1 with 100 terminals.

In Figure B.2, the red and the green lines denote the $Max[\Lambda_i]_{\forall}i$ and $2(N-1)M_{max}$ over the different communication ranges for scenario 2. Again it can be observed from the figure that the network congestion is more severe at the backbone network than at any dominant terminal due to the uplink-downlink traffic from the associated non-dominant terminals for all communication ranges below 400.

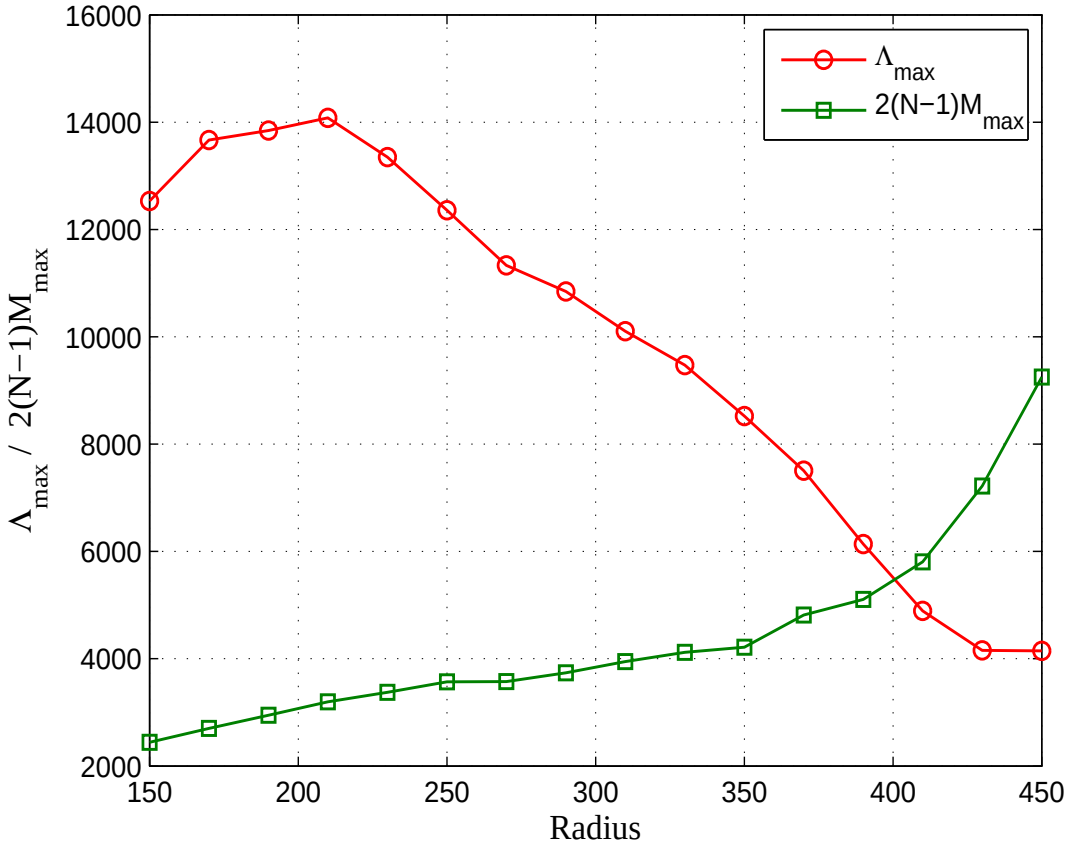

Figure B.2: $Max[\Lambda_i]_{\forall}i$ versus $2(N-1)M_{max}$ for scenario 2 with 200 terminals.

References

[1] S. Basagni, M. Conti, S. Giordano, and I. Stojmenovic, *Mobile Ad Hoc Networking*. IEEE Press, 2003.

[2] A. Mainwaring, D. Culler, J. Polastre, R. Szewczyk, and J. Anderson, "Wireless sensor networks for habitat monitoring," in *WSNA '02 Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, pp. 88–97, ACM, 2002.

[3] H. Moustafa and Y. Zhang, *Vehicular Networks: Techniques, Standards, and Applications*. Auerbach Publications Boston, MA, USA 2009, 2009.

[4] S. Olariu and M. C. Weigle, *Vehicular Networks: From Theory to Practice*. Chapman & Hall/CRC 2009, 2009.

[5] J. A. Freebersyser and B. Leiner, "A dod perspective on mobile ad hoc networks," in *Ad Hoc Networking* (C. E. Perkins, ed.), pp. 29–51, Addison-Wesley, 2001, 2001.

[6] T. Fujiwara and T.Watanabe, "An ad hoc networking scheme in hybrid networks for emergency communications," *Ad Hoc Networks*, vol. 3, no. 5, 2005.

[7] V. Bharghavan and B. Das, "Routing in ad hoc networks using minimum connected dominating sets," in *International Conference on Communications 97*, pp. 376–380, June 1997.

[8] B. Das, R. Sivakumar, and V. Bhargavan, "Routing in ad-hoc networks using a spine," in *International Conference on Computers and Communications Networks 97*, pp. 376–380, September 1997.

[9] J. Wu and H. Li, "On calculating connected dominating set for efficient routing in ad hoc wireless networks," in *Proc. Third Intl. Workshop Discrete Algorithms and Methods for Mobile Computing and Communication*, pp. 7–14, 1999.

[10] F. Dai and J. Wu, "An extended localized algorithm for connected dominating set formation in ad hoc wireless networks," *IEEE Transactions. Parallel and Distributed Systems*, vol. 15, no. 10, 2004.

[11] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris, "Span: An energy-ecient coordination algorithm for topology maintenance in ad hoc wireless networks," *Wireless Network*, vol. 8, no. 5, 2002.

[12] F. Dai and J. Wu, "On constructing k-connected k-dominating set in wireless networks," *Journal of Parallel and Distributed Computing (JPDC)*, vol. 66, no. 7, 2006.

[13] F. Wang, M. Thai, and D.-Z. Du, "On the construction of 2-connected virtual backbone in wireless networks," *IEEE Transactions on Wireless Communications*, vol. 8, no. 3, 2009.

[14] Y. Wu, F. Wang, M. T. Thai, and Y. Li, "Constructing k-connected m-dominating sets in wireless sensor networks," in *Military Communications Conference*, pp. 1–7, Oct 2007.

[15] W.-P. Lyui, *Design of a new operational structure for mobile radio networks*. PhD thesis, Clemson University, Clemson, SC, 1991.

[16] J. L. Hammond and H. B. Russell, "Properties of transmission assignment algorithm for multiple-hop packet radio networks," *IEEE Trans. on Wireless Communications*, vol. 3, no. 4, July 2004.

[17] B. J. Wolf, J. L. Hammond, and H. B. Russell, "A distributed load-based transmission scheduling protocol for wireless ad hoc networks," in *International Conference on Wireless Communications and Mobile Computing (ACM IWCMC)*, pp. 437–442, July 2006.

[18] M. Garey and D. Johnson, "Computers and intractability: A guide to the theory of np-completeness." Freeman and Company, 1979.

[19] C. Adjih, P. Jacquet, and L. Viennot, "Computing connected dominated sets with multipoint relays," *Ad hoc & Sensor Wireless Networks*, vol. 1, no. 1-2, pp. 27–39, 205.

[20] T. Clausen and P. Jacquet, "Rfc3626: Optimized link state routing protocol (olsr)." RFC Editor United States, 2003.

[21] J. P. Macker, W. Chao, and J. Dean, "Simplified multicast forwarding in mobile ad hoc networks," in *Proceedings of MILCOM 2004*, 2004.

[22] J. P. Macker, "Simplified multicast forwarding." Mobile Ad hoc Networks Working Group, Internet Engineering Task Force, Work in Progress, March 6, 2012.

[23] B. J. Wolf, J. L. Hammond, and H. B. Russell, "Designing transmission schedules for wireless ad hoc networks to maximize network throughput," in *MILCOM 2005*, pp. 2012–2018, Oct 2005.