UNIVERSITY OF CALIFORNIA
SANTA CRUZ

NEIGHBOR-AWARE CONTROL IN AD HOC NETWORKS

A dissertation submitted in partial satisfaction of the
requirements for the degree of

DOCTOR OF PHILOSOPHY

in

COMPUTER SCIENCE

by

Lichun Luke Bao

December 2002

The Dissertation of Lichun Luke Bao
is approved:

Professor J.J. Garcia-Luna-Aceves, Chair

Professor Katia Obraczka

Professor Patrick Mantey

Frank Talamantes
Vice Provost and Dean of Graduate Studies

Report Documentation Page	*Form Approved* *OMB No. 0704-0188*

1. REPORT DATE **DEC 2002**	2. REPORT TYPE	3. DATES COVERED **00-12-2002 to 00-12-2002**

4. TITLE AND SUBTITLE **Neighbor-Aware Control in Ad Hoc Networks**	5a. CONTRACT NUMBER
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **University of California at Santa Cruz,Department of Computer Engineering,Santa Cruz,CA,95064**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES
The original document contains color images.

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**		**176**	

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std Z39-18

Contents

List of Figures

List of Tables

Abstract

Neighbor-Aware Control in Ad Hoc Networks

by

Lichun Luke Bao

Ad hoc networks have very unique features, such as dynamic topologies, relatively limited bandwidth and wireless signal propagation schemes, which present difficult challenges for wireless communication. We propose control mechanisms for channel access scheduling and topology control in ad hoc networks, respectively, which utilize the neighborhood information within two hops to cope with the difficulties of communicating in ad hoc networks. First, we present the neighbor-aware contention resolution (NCR) algorithm, and analyze its generic performance with regard to the contention delay and system throughput. The required neighbor information in NCR for ad hoc networks is acquired through the neighbor protocol, which is based on a random channel access mechanism and a reliable message propagation scheme using retransmissions. Then, four channel access protocols based on NCR are presented, namely NAMA, LAMA, PAMA and HAMA, which correspond to node-, link-, pair-wise- and hybrid-activation multiple access protocols, respectively. These protocols are aimed at ad hoc networks with omnidirectional antennas, and their performance is analyzed. Furthermore, channel access protocols, adapted from NAMA and PAMA, are considered for heterogeneous ad hoc networks that include unidirectional links, which may occur due to power and signal propagation differences between wireless

stations. Furthermore, we apply the NCR algorithm to the channel access mechanisms in ad hoc networks with directional antennas. Directional antenna can enhance or cancel out the radiating electromagnetic waves in certain directions, thus providing great potential for network throughput improvements. Especially, we consider the use of multi-beam adaptive array (MBAA), which is capable of forming multiple beams and communicating with multiple neighbors concurrently. The received-oriented multiple access (ROMA) controls channel access in ad hoc networks with MBAA antennas. The last part of this dissertation applies NCR to topology management by priority ordering (TMPO) for constructing the virtual overlay networks in ad hoc network topologies. The virtual overlay network is a fundamental structure for efficient communication in ad hoc networks.

Acknowledgements

First and foremost, I thank my advisor J.J. Garcia-Luna-Aceves for accepting me as his doctoral student and guiding me through all these years. With such an endearing and encouraging mentor, I find the pursuit for the Ph.D. more rewarding than I thought, and has even dramatically changed my view of satisfaction when developing a meaningful profession.

Prof. Katia Obraczka and Prof. Patrick Mantey have kindly served on my dissertation committee and provided insightful comments and advice on various aspects of the dissertation. Especially, I thank Prof. Ira Pohl, Prof. Glen Langdon and Dr. Chane Fullmer for serving in my Ph.D. advancement committee. Chane has given a lot of valuable advice to the research.

I thank my fellow *CoCo*s in the computer communication research group (CCRG) for their inventive and pioneering work that has given me incentives and directions to the research in this dissertation. I am grateful to their kindness and friendship which I will cherish for life.

I wish to thank my family for their love and faith in me. My wife Haburina has been a constant support in achieving my goals. Each and every step in my graduate study have been accompanied with her love, trust and encouragement.

All works from this dissertation were co-authored by my advisor J.J.

- Chapter 2: The neighbor-aware contention resolution (NCR) algorithm was the result of an almost night-long reflection after a discussion with J.J. about the

topology-dependent channel access methods in ad hoc networks, which is the foundation of the work.

- Chapter 3: NCR soon evolved into several solutions to the channel access scheduling problem for ad hoc networks with omnidirectional antennas [11] [12] [13]. The hybrid channel access scheduling for ad hoc networks with omnidirectional antennas was studied under the suggestions from George Vardakas and Wendell Y. Kishaba from Raytheon Company [14] [15].

- Chapter 4: The specialized channel access mechanisms [12] were the results of the quest to realize collision-free communication capability in ad hoc networks with unidirectional links, and to further the research on routing in the same area [8] [10].

- Chapter 5: The needs from Raytheon Company for supporting directional antennas in ad hoc communication environments led to the results on the receiver-oriented multiple access protocol presented [16] [17] [45].

The work carried out in this dissertation was sponsored in part by the Defense Advanced Research Projects Agency (DARPA) under Grant No. F30602-97-2-0338 and No. DAAD19-01-C-0026, and by the U.S. Air Force/OSR under Grant No. F49620-00-1-0330.

All in all, this dissertation is accomplished through the help of all the people around me and behind the scene. I heartily appreciate the blessing that fate has arranged me to be among them.

To Haburina and Lambert

Chapter 1

Introduction

Ad hoc networks provide a large number of static or mobile stations with spontaneous interconnections and instantaneous access to information resources. In such networks, the communication channel is shared among the independent stations. Hence, ad hoc networks need a set of control mechanisms to regulate the access to the shared channel for maximum channel utilization. The design of channel access protocol depends on the physical technologies provided and the topology information presented. This dissertation focuses on the channel access in ad hoc networks to handle different physical and topological characteristics, such as antenna technologies, and directionalities of network topologies.

1.1 Channel Access with Omnidirectional Antennas

Channel access protocols for ad hoc networks can be contention-based or scheduled. The contention-based approach started with ALOHA and CSMA [46], and continued with several collision avoidance schemes, of which the IEEE 801.11 standard series for wireless LANs [26] [1] are the most prominent to date. However, as the network load increases, network throughput drastically degrades because the probability of collisions rises in contention-based protocols, preventing stations from acquiring the channel.

In contrast, scheduled access schemes set up timetables for individual nodes or links, such that the transmissions from the nodes or over the links are conflict-free in the code, time, frequency or space divisions of the channel. The schedules for conflict-free channel access can be established based on the topology of the network, or it can be topology independent.

Topology-dependent channel access control algorithms can establish transmission schedules by either dynamically exchanging and resolving time slot requests [23] [82], or by pre-arranging a time-table for each node based on the network topologies. Setting up a conflict-free channel access time-table is typically treated as a node- or link- coloring problem on graphs representing the network topologies. The problems of optimally scheduling access to a common channel are some of the classic NP-hard problems in graph theory (k-colorability on nodes or edges) [31] [34] [69]. Polynomial algorithms are known to achieve suboptimal solutions using heuristics based on such graph attributes as the degree of the nodes.

A unified framework for TDMA/FDMA/CDMA channel assignments, called UxDMA algorithm, was described by Ramanathan [65]. UxDMA summarizes the patterns of many other channel access scheduling algorithms into a single framework.

These algorithms are represented by UxDMA with different parameters. The parameters in UxDMA are the constraints put on the graph entities (nodes or links) such that entities related by the constraints are colored differently. Based on the global topology, UxDMA computes the node or edge coloring, which correspond to channel assignments to these nodes or links in the time, frequency or code domains. Obviously, collecting the complete topology of the network and distributing the corresponding schedule pose a major challenge when applying UxDMA in ad hoc networks.

A number of topology-transparent scheduling methods have been proposed [22] [44] [49] to provide conflict-free channel access that is independent of the radio connectivity around any given node. The basic idea of the topology-transparent scheduling approach is for a node to transmit in a number of time slots in each time frame. The times when node i transmits in a frame corresponds to a unique code such that, for any given neighbor k of node i, node i has at least one transmission slot during which node k and none of node k's own neighbors are transmitting. Therefore, within any given time frame, any neighbor of node i can receive at least one packet from node i conflict-free. An enhanced topology-transparent scheduling protocol, TSMA (Time Spread Multiple Access), was proposed by Krishnan and Sterbenz [49] to reliably transmit control messages with acknowledgments. However, TSMA performs worse than CSMA in terms of delay and throughput [49]. Chapter 2 presents a novel neighbor-aware contention resolution (NCR) algorithm that is used to derive several channel access control mechanisms. According to NCR, each entity among a group of contending entities knows its direct and indirect contenders in a certain context (*e.g.*, a time slot on a frequency band). Contention between the entities is resolved by each context according to the priorities assigned to the entities, which are derived from the message digest of the context number and the respective identifiers. The winner is entitled to the entity with the highest priority among its contenders without conflicts. We analyze the packet delay encountered in NCR using a general queuing model under certain contention level. A neighbor protocol for ad hoc networks is presented for collecting the critical neighborhood information required by NCR.

Based on NCR, Chapter 3 derives four multiple access protocols for ad hoc networks with omnidirectional antennas, which schedule node activation multiple access (NAMA) suitable for broadcast communication [9], link activation multiple access (LAMA) and pair-wise link activation multiple access (PAMA) for unicast communication [11] [13], and hybrid activation multiple access (HAMA) [14] [15] for both unicast and broadcast communications, respectively. The throughput attributes of these protocols are analyzed using a generic ad hoc network topology model.

1.2 Communication over Unidirectional Links

Unidirectional links may exist in a network due to differences in transmission power, code rate, terrain, antenna capability, transmission media used among routers, and other reasons. Unidirectional links can provide shorter communication paths between stations, and improve the latency for end-to-end transmissions. The motivation to support unidirectional links in heterogeneous wireless networks have drawn considerable amount of research interests in providing routing capability in the presents of unidirectional links [2] [37] [60] [62]. Especially, the link state algorithms based on topology flooding [56] [29], the unidirectional link routing (UDLR) based on creating tunnels in the reverse direction of the unidirectional links [28], the circuit-based link-state approach for unidirectional routing [30] [33] [27] and many others ([5] [43] [52] [59] [61] [79]) have been proposed to take advantage of the existence of unidirectional links and improve network throughput in ad hoc networks.

However, the utilization of the unidirectional links for data communication is a two-sided sword. Although stable and usable unidirectional links can provide shorter paths to reach certain destinations, transmissions over the unidirectional links may create severe unwanted interference at the other stations without proper channel access coordinations. There has been little research in providing collision-free channel access control mechanisms for ad hoc networks with unidirectional links.

Channel access control in ad hoc networks with unidirectional links faces two options: contention-based or scheduled. The contention-based coordination between the end-points of a unidirectional link appears impossible because it may require coordination between the end-points over a multihop path, which further needs large scale knowledge about the network topology. Even the ALOHA protocol applied on terrestrial links requires an acknowledgment to the sender in order for the sender to decide the successful delivery of the message. Therefore, the utilization of unidirectional links demands a channel access scheme with *a priori* topology knowledge to ensure prompt and collision-free data transmission.

Chapter 4 introduces a new family of collision-free channel access protocols for ad hoc networks with unidirectional links, which we call PANAMA (Pair-wise link Activation and Node Activation Multiple Access), to enable correct channel access in the presence of unidirectional links [12]. PANAMA is composed of NAMA-UN (node activation multiple access for unidirectional networks) and PAMA-UN (pair-wise activation multiple access for unidirectional networks) to support broadcast and unicast services, respectively.

1.3 Channel Access with Directional Antennas

Omnidirectional antennas spread the electromagnetic energy of the signal over a large regions of space, and only a very small portion is actually received by the intended station. This limits system performance and capacity due to multipath fading, delay spread, and co-channel interference (CCI) [78]. Recently, the availability of low-cost computing capacity for processing digital signals from arrays of simple antennas have made "smart" directional antennas practical for wireless communication systems [25]. By actively controlling the temporal pacing between the radiating elements of an antenna array with the digital signal processing (DSP) component, directional antennas can enhance or cancel out the radiating electromagnetic waves in certain directions. In this way, radio propagation energy is concentrated in specific directions from the standpoint of the transmitter. Similarly, the receiver can enhance or nullify the sensitivity of the antenna in certain directions, thus eliminating many of the multipath effects and co-channel interference (CCI).

With M antenna elements, an antenna array generally provides an increased antenna gain of M plus a diversity gain against multipath fading [70] [78]. When a constant signal gain is maintained along the direction of interest and the nulls are adjusted toward the sources of interference, it can dramatically increase the performance characteristics of a wireless system in terms of the capacity, coverage and quality. Based on more complex DSP technologies than the directional antennas that are capable of forming a single beam, an antenna array, called multi-beam adaptive array (MBAA) capable of forming multiple beams for several simultaneous receptions or transmissions, can even enlarge the capacity of the networks many folds [77].

When directional antennas are installed on mobile stations for both transmission and reception in ad hoc networks, the media access control (MAC) protocols face two problems:

1. How to track the directional positions of mobile neighbor stations in order to point antenna beams.

2. How to couple neighboring stations for concurrent transmissions and receptions, given that every node has multiple neighbors and each node may intend to either transmit or receive.

In order to find out neighbor locations, the omnidirectional mode of the antenna is usually employed to allow the detection of neighbor angular positions [66]. Zander [81] and Ward [77] presented channel access protocols based on slotted ALOHA and directional antennas with single beam and multiple beam forming capabilities.

Data packets are transmitted in omnidirectional fashion and are received in directional mode. A special preamble is added to each packet for signal detection and beam orientation at the receivers. In contrast, Ko *et al.* [47] and Nasipuri *et al.* [58] presented carrier sense multiple access with collision avoidance (CSMA/CA) in which the transmitters use the directional mode of the antennas for transmitting request-to-send (RTS) signals and receiving the corresponding clear-to-send (CTS) replies. The receiver antennas stay in omnidirectional mode for both RTS and CTS. Nasipuri *et al.* proposed to utilize the switched multi-beam forming capability of the directional antennas for establishing communicating pairs [58].

Because omnidirectional transmission and reception are susceptible to interference and collisions, these contention-based access protocols suffer from severe network throughput degradation when the channel access demand increases and causes great probability of collisions. Using a technique that caches the angle of arrival (AoA) information, Takai *et al.* [74] partly eliminated the dependency on the omnidirectional mode of directional antennas, and only fell back to the omnidirectional mode if the AoA profile is not available.

Based on the NCR algorithm, the receiver-oriented multiple access (ROMA) for ad hoc networks with directional antennas is presented in Chapter 5 [45] [17] [16]. ROMA addresses the topology-dependent channel access scheduling problem for directional antennas that are capable of forming multiple beams.

1.4 Topology Control

In mobile ad hoc networks, the network topology changes frequently, and communications control protocols usually require large amount of topology information exchanges in order to maintain entity reachability and network connectivity. Weeding out redundant and unnecessary topology information so as to provide only the minimum and sufficient connectivity information is usually referred to as *topology control* or *topology management*. Topology control in ad hoc networks can improve the efficiency of the routing control protocols [71] [72] and provide useful information for efficient channel access, such as the propagation of broadcast data [24].

There are two approaches to topology management in ad hoc networks — power control and hierarchical topology organization. Power control mechanism tries to balance network connections by adjusting the power of each node such that just enough one-hop neighbors are covered within the transmission range [40] [57] [73]. Li *et al.* [53] proposed that network connectivity is minimally maintained as long as the decreased power level keeps at least one neighbor remaining connected at every $2\pi/3$ to $5\pi/6$ angular separation. Ramanathan *et al.* [67] proposed to incrementally adjust nodes' power levels so as to keep network connectivity at each step. However, except for the early work by Takagi and Kleinrock [73], topologies derived from power control often result in unidirectional links that create harmful interference due to the different transmission ranges among one-hop neighbors [63]. The dependencies on volatile information in mobile networks, such as node locations [40], signal strength or angular positions [53] also contribute to the instability of these topology control algorithms. Furthermore, some distributed implementations of the topology control algorithms can hardly improve the throughput of mobile networks [67].

In this dissertation, we focus on hierarchical topology management, which entrusts a selected subset of the nodes in the network to impose a backbone topology and to carry out essential forwarding and control functionalities (e.g. [48]). It is required that the selected subset of nodes be minimum as well as connected. In graph theory, the minimum dominating set (MDS) problem and the relevant connected dominating set (CDS) problem best describe the hierarchical approach in communication networks, often referred to as *clustering*. In the clustering approach, a set of *clusterheads* are selected, comprising the MDS, and *gateways* are also chosen to connect clusterheads, so that the union of gateways and clusterheads forms the CDS.

Clusterheads can be elected via negotiation and by using some deterministic criteria. Negotiations require multiple non-deterministic and incremental steps, and may incur an election jitter during the process, due to the lack of consensus about the nodes that should be the clusterheads. Examples of this approach are the "core" extraction algorithm [72] and the spanning-tree algorithm [39]. SPAN [20] allows a node to delay clusterhead announcement for different amounts of time to inspect the clusterheads in its one-hop neighborhood.

On the other hand, deterministic criteria can resolve the clusterheads in a single round. Different heuristics have been used to form clusters and to elect clusterheads. Many authors [3] [6] [32] [38] [54] chose to use the node identifiers to elect the clusterhead within one or multiple hops. We refer to this approach as **Lowest ID**.

Banerjee and Khuller [7] assigned a weight value to each node for clusterhead election, which is essentially the same as **Lowest ID**.

The node degree is another commonly used metric in which nodes with more one-hop neighbors are more likely to become clusterheads [39] [41] [72]. We refer to this approach as **Max Degree**. Chiang *et al.* [21] have shown that the **Lowest ID** algorithm performs better than the clusterhead election algorithms based on node degrees in terms of clusterhead stability in mobile multihop networks.

Basu *et al.* [18] suggested to use the mean received-signal strength variations as the metric for clusterhead election, which favors relatively stationary nodes to become the clusterheads. We refer to this approach as **MOBIC**.

Once elected, the minimum dominating set and the connected dominating set help reduce the complexity of maintaining topology information, and can simplify such essential functions as routing, bandwidth allocation, channel access, power control or virtual-circuit support. Therefore, clusterhead stability is an important factor in keeping the cluster maintenance cost down [3]. However, it may be harmful to the clusterheads, because a clusterhead drains its energy more quickly than a normal node. Therefore, a clusterhead election algorithm must also consider load balancing of the clusterhead role to avoid node or network failure. Except for **Lowest ID**, the aforementioned election algorithms inherently provide clusterhead load balancing in mobile networks. To improve **Lowest ID**, Amis *et al.* [4] provided clusterhead load balancing, referred to as **Load Balance**, by running virtual identifier (VID) and budget counters at each node. **Load Balance** uses the VID for elections, and the budget for the clusterhead term, thus posing equal opportunity for each node to become a clusterhead. The algorithm assumes that all nodes start with the same power and consume it at the same rate.

Clustering algorithms that build clusters within d hops from the clusterhead (called d-clustering) have also been proposed [3] [48] [68]. However, d-clustering requires flooding in search of clusterheads [3], thus obviates the purpose of the topology management for efficiency. We only consider two-hop clustering in this dissertation.

Chapter 6 presents the application of NCR to the topology management mechanism in mobile ad hoc networks, which we call TMPO (topology management by priority ordering). TMPO elects the minimal dominating set and the connected dominating set of ad hoc networks by the priorities assigned to mobile stations. The priority of a node is periodically recomputed from the node's identifier and incorporates two key factors for topology management in ad hoc networks — the nodal battery life, mobility, thus providing load balancing among mobile stations in TMPO.

Chapter 2

Neighbor-Aware Contention Resolution

Channel access problem in ad hoc networks is a special dynamic leader-election problem in which multiple leaders are elected from time to time for transmissions to their neighbors through the shared channel. Elected leaders are required to be unique in their neighborhood for successful transmissions.

Channel access protocols for ad hoc networks can be contention-based, or scheduled. Contention-based channel access protocols depends on spontaneous message exchanges to resolve the eligibility of transmission, such as request-to-send (RTS) and clear-to-send (CTS) message coordinations. However, the network throughput may drastically degrade when the leader election becomes increasingly competitive due to the growing traffic demand and the potential collisions between the RTS/CTS messages.

We propose a neighbor-aware contention resolution (NCR) algorithm to decide the eligibility of an entity for accessing a shared resource. NCR assumes the knowledge of the identifiers of the contenders and the current contention context number, from which it derives a randomized priority for each contender in the given contention context. Then, each contender locally determines its eligibility to access the resource in the contention context by comparing its priority with those of other contenders. Because the scheduling is dynamic, depending on the contention context, a different schedule is established in each contention context.

Section 2.1 presents the NCR algorithm and analyzes the packet delay encountered in a general queuing model under certain contention level. In ad hoc networks, we can adapt the NCR algorithm to various network models for channel access scheduling purposes. Because the contention to the channel happens among neighbors within two hops from each node in ad hoc networks, Section 2.2 presents a novel neighbor protocol for acquiring the two-hop neighbor information.

2.1 Neighbor-Aware Contention Resolution

2.1.1 Specification

In general beyond the ad hoc network scenarios, the neighbor-aware contention resolution (NCR) envisions a special election problem for an entity to locally decide the leadership status of itself among a known set of contenders in any given contention context. We assume that the knowledge of the contenders for each entity is acquired by an appropriate means, depending on the specific applications. For example, in the ad hoc networks of our interest, the contenders of each node are the neighbors within two hops, which can be obtained by each node periodically broadcasting the identifiers of its one-hop neighbors [17]. Furthermore, NCR requires that each contention context be identifiable, such as the time slot number in networks with a time-division multiple access scheme.

The election problem for neighbor-aware contention resolution is be formulated as “Given a set of contenders, M_i, against entity i in contention context t, how should the precedence of entity i in the set $M_i \cup \{i\}$ be established, such that every other contender yields to entity i whenever entity i establishes itself as the leader for the shared resource?”

To decide the precedence of entity i without incurring communication overhead between its contenders, we assign each entity a priority that depends on the identifier of the entity and varies according to the known contention context so that the criterion for the leadership is deterministic and fair among the contenders. Eq. (2.1) provides a formula to derive the priority for entity i in contention context t, denoted by $i.\texttt{prio}$.

$$i.\texttt{prio} = \texttt{Hash}(i \oplus t) \oplus i, \tag{2.1}$$

where the function $\texttt{Hash}(x)$ is a fast message digest generator that returns a random integer in range $[0, M]$ by hashing the input value x, and the sign ‘$\oplus$’ is designated to carry out the concatenation operation on its two operands. Note that, while the Hash function can generate the same number on different inputs, each priority number is unique because the priority is appended with identifier of the entity.

```
NCR(i, t)
{
    /* Initialize. */
    for (k ∈ M_i ∪ {i})
        k.prio = Hash(k ⊕ t) ⊕k;

    /* Resolve leadership. */
    if (∀k ∈ M_i, i.prio > k.prio)
        i is the leader;
} /* End of NCR. */
```

Figure 2.1: NCR Specification.

Figure 2.1 describes the NCR algorithm. Basically, NCR generates a *permutation* of the contending members, the order of which is decided by the priorities of all participants. Because the priority is a pseudo-random number generated from the contention context that changes from time to time, the permutation also becomes random, such that each entity has certain probability that is commensurate to its contention level to be elected in each contention context. Denote the probability of winning the election as q_i for entity i, we have

$$q_i = \frac{1}{|M_i \cup \{i\}|} \ . \tag{2.2}$$

Because it is assumed that contenders have mutual knowledge and t is synchronized, the order of contenders based on the priority numbers is consistent at every participant, thus avoiding any conflict among contenders.

2.1.2 Dynamic Resource Allocation

The description of NCR provided thus far evenly divides the shared resource among the contenders. In practice, the demands from different entities may vary, which requires appropriate allocation of the shared resource. There are several approaches for allocating variable portions of the resource according to individual demands. In any approach, an entity needs to specify its demand by an integer value chosen from a given integer set. In the following discussion, we denote the resource demand from entity i by p_i. Because the demands need to be propagated to the contenders before the contention resolution process, the integer set should be small and allow enough granularity to accommodate the demand variations while avoiding the excess control overhead caused by the demand fluctuations.

Suppose the integer set is from 0 to P. The following three approaches for resource allocation differ in the portion of the resource allocated on a given integer value. If the resource demand is 0, the entity has no access to the shared resource.

Pseudo identities An entity assumes p pseudo identities, each defined by the concatenation of the entity identifier and a number from 1 to p. For instance, entity i with resource demand p_i is assigned with the following pseudo identities:

$$i \oplus 1,\ i \oplus 2,\ \cdots,\ i \oplus p_i.$$

Each identity works for the entity as a contender to the shared resource. Figure 2.2 specifies NCR with pseudo identities (NCR-PI) for resolving contentions among contenders with different resource demands.

```
NCR-PI(i, t)
{
    /* Initialize each entity k with demand p_k. */
    for (k ∈ M_i ∪ {i} and 1 ≤ l ≤ p_k)
        (k ⊕ l).prio = Hash(k ⊕ l ⊕ t) ⊕k ⊕ l;

    /* Resolve leadership. */
    if (∃k, l : k ∈ M_i, 1 ≤ l ≤ p_k and
        ∀m : 1 ≤ m ≤ p_i, (k ⊕ l).prio > (i ⊕ m).prio)
        i is not the leader;
    else
        i is the leader;
} /* End of NCR-PI. */
```

Figure 2.2: NCR-PI Specification.

The portion of the resource available to an entity i in NCR-PI is proportional to its resource demand as follows:

$$q_i = \frac{p_i}{\sum_{k \in M_i \cup \{i\}} p_k}. \tag{2.3}$$

Root operation Assuming enough computing power for floating point operations at each node, we can use the root operator to achieve the same proportional allocation of the resource among the contenders as in NCR-PI.

Given that the upper bound of function `Hash` in Eq. (2.1) is M, substituting line 2 in Figure 2.1 with the following formula generates a new algorithm which provides the same resource allocation characteristic as shown in Eq. (2.3).

$$k.\texttt{prio} = \left(\frac{\texttt{Hash}(k \oplus t)}{M}\right)^{\frac{1}{p_k}} . \tag{2.4}$$

Multiplication Simpler operations, such as multiplication in the priority computation, can provide non-linear resource allocation according to the resource demands.

Substituting line 2 in Figure 2.1, Eq. (2.5) offers another way of computing the priorities for entities.

$$k.\texttt{prio} = (\texttt{Hash}(k \oplus t) \cdot p_k) \oplus k \ . \tag{2.5}$$

According to Eq. (2.5), the priorities corresponding to different demands are mapped onto different ranges, and entities with smaller demand values are less competitive against those with larger demand values in the contentions, thus creating greater difference in resource allocations than the linear allocation schemes provided by Eq. (2.3) and Eq. (2.4). For example, among a group of entities, a, b and c, suppose $p_a = 1$, $p_b = 2$, $p_c = 3$ and $P = 3$. Then the resource allocations to a, b and c are statistically $\frac{1}{3} \cdot \frac{1}{3} = 0.11$, $\frac{1}{3} \cdot \frac{1}{3} + \frac{1}{3} \cdot \frac{1}{2} = 0.28$, $\frac{1}{3} \cdot \frac{1}{3} + \frac{1}{3} \cdot \frac{1}{2} + \frac{1}{3} \cdot \frac{1}{1} = 0.61$, respectively.

For simplicity, the rest of this chapter addresses NCR without dynamic resource allocation.

2.1.3 Performance

System delay We assume NCR as an access mechanism to a shared resource at a server (an entity), and analyze the average delay experienced by each client in the system according to the M/G/1 queuing model, where clients arrive at the server according to a Poisson process with rate λ and are served according to the first-come-first-serve (FIFO) discipline. Specifically, we consider the time-division scheme in which the server computes the access schedules by the time-slot boundaries, and the contention context is the time slot. Therefore, the queuing system with NCR as the access mechanism is an M/G/1 queuing system with server vacations, where the server takes a fixed vacation of one time slot when there is no client in the queue at the beginning of each time slot.

The system delay of a client using NCR scheduling algorithm can be easily derived from the extended Pollaczek-Kinchin formula, which computes the service waiting time in an M/G/1 queuing system with server vacations [19]

$$W = \frac{\lambda \overline{X^2}}{2(1 - \lambda \overline{X})} + \frac{\overline{V^2}}{2\overline{V}},$$

where X is the service time, and V is the vacation period of the server.

According to NCR, the service time X of a head-of-line client is a discrete random variable, governed by a geometric distribution with parameter q, where q is the probability of the server accessing the shared resource in a time slot, as given by Eq. (2.2). Therefore, the probability distribution function of service time X is

$$P\{X = k\} = (1 - q)^{k-1} q \ ,$$

where $k \geq 1$. Therefore, the mean and second moments of random variable X are:

$$\overline{X} = \frac{1}{q} \ , \quad \overline{X^2} = \frac{2 - q}{q^2} \ .$$

Because V is a fixed parameter, it is obvious that $\overline{V} = \overline{V^2} = 1$. Therefore, the average waiting period in the queue is:

$$W = \frac{\lambda(2 - q)}{2q(q - \lambda)} + \frac{1}{2} \ .$$

Adding the average service time to the queuing delay, we get the overall delay in the system:

$$T = W + \overline{X} = \frac{2 + q - 2\lambda}{2(q - \lambda)} \ . \tag{2.6}$$

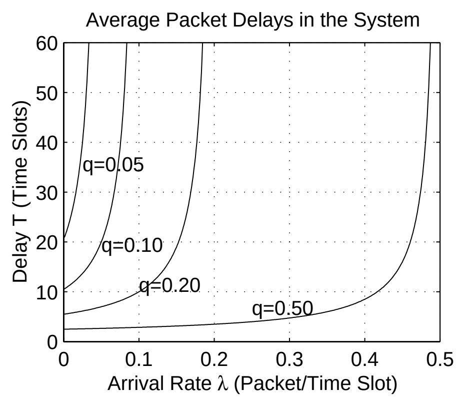


Figure 2.3: Average system delay of packets.

The probabilities of the server winning a contention context are different, and so are the delays of clients going through the server. Figure 2.3 shows the relation between the arrival rate and the system delay of clients in the queuing system, given different resource access probabilities. To keep the queuing system in a steady state, it is necessary that $\lambda < q$ as implied by Eq. (2.6).

System throughput NCR guarantees successful service to the clients because of the collision freedom it provides. Therefore, the throughput of the server (the entity) over the shared resource is the minimum of the client arrival rate and the resource access probability. Considering all contenders for the shared resource, the overall system throughput is the summary of the throughput at individual entities. We have the following system throughput S combined from each and every entity k that competes for the shared resource:

$$S = \sum_k \min(\lambda_k, q_k) \tag{2.7}$$

where q_k is the probability that k may access the resource, and λ_k is the client arrival rate at k.

2.1.4 Delay Improvement

Transmission without collision is the main objective of NCR. However, because the interval between successive resource accesses is a random variable that follows a Geometric distribution, it is possible for a given entity to not being able to access the shared resource for a long time. To achieve bounded intervals between successive accesses by the same station, we provide a mechanism that allows transmission with the possibility of collisions.

After the NCR operates for a certain period of time within a group of contenders, there is a high probability that each and every entity has accessed the shared resource at least once, and entities that have not won in the contentions are very rare. If such entities do exist, the delay-bounding mechanism requires that these entities acquire the shared resource as soon as possible. We allocate a special time slot, called *dark slot*, once after every fixed period of time, and allow entities that have not won in the previous contentions access the shared resource without priority comparison process, while forbidding any entity that has accessed the resource from using the dark slot. The interval between successive dark slots is set long enough so that most entities have accessed before the dark slot, and that accesses to the resource in the dark slot are so sparse that collisions in the dark slots happen at a very small probability.

We analyze the required interval between successive dark slots to achieve a high probability that all entities have transmitted before each dark slot during each period. We denote the number of contenders to an entity by n, and the required interval by k. We also set the probability of all entities having won in the contentions to 99%.

The above problem is similar to an equivalent occupancy problem in combinatorial mathematics [35] [51] as formulated below:

> There are n distinct balls in a bin. Each time grab one ball randomly and put it back to the bin for the next grab. Repeat k times. How many ways are there in which exactly j balls are never grabbed?

Then the number of ways that specific balls 1, 2, $\cdots$, i are untouched in k grabs is $(n-i)^k$. Furthermore, the number of ways that these balls are untouched in the grab with exactly j distinct balls not grabbed ($j \geq i$) is:

$$U_j \frac{C(n-i,j-i)}{C(n,j)} = U_j \frac{j!(n-i)!}{n!(j-i)!}$$

in which $C(m,n)$ denotes the number of combinations of m members from a group of n elements. Let $j = i, i+1, i+2, \cdots, n-3, n-2, n-1$, we have:

$$(n-i)^k = U_i \frac{i!(n-i)!}{n!0!} + U_{i+1} \frac{(i+1)!(n-i)!}{n!1!} + U_{i+2} \frac{(i+2)!(n-i)!}{n!2!} + \cdots$$
$$+ U_{n-3} \frac{(n-3)!(n-i)!}{n!(n-3-i)!} + U_{n-2} \frac{(n-2)!(n-i)!}{n!(n-2-i)!} + U_{n-1} \frac{(n-1)!(n-i)!}{n!(n-1-i)!}$$

In addition, let $i = 0, 1, 2, \cdots, n-1$, we have:

$$
\begin{pmatrix}
1 & 1 & 1 & 1 & \cdots & 1 & 1 & 1 \\
 & \frac{1!(n-1)!}{n!0!} & \frac{2!(n-1)!}{n!1!} & \frac{3!(n-1)!}{n!2!} & \cdots & \frac{(n-3)!(n-1)!}{n!(n-4)!} & \frac{(n-2)!(n-1)!}{n!(n-3)!} & \frac{(n-1)!(n-1)!}{n!(n-2)!} \\
 & & \frac{2!(n-2)!}{n!0!} & \frac{3!(n-2)!}{n!1!} & \cdots & \frac{(n-3)!(n-2)!}{n!(n-5)!} & \frac{(n-2)!(n-2)!}{n!(n-4)!} & \frac{(n-1)!(n-2)!}{n!(n-3)!} \\
 & & & \frac{3!(n-3)!}{n!0!} & \cdots & \frac{(n-3)!(n-3)!}{n!(n-6)!} & \frac{(n-2)!(n-3)!}{n!(n-5)!} & \frac{(n-1)!(n-3)!}{n!(n-4)!} \\
 & & & & \cdots & \cdots & \cdots & \cdots \\
 & & & & & \frac{(n-3)!3!}{n!0!} & \frac{(n-2)!3!}{n!1!} & \frac{(n-1)!3!}{n!2!} \\
 & & & & & & \frac{(n-2)!2!}{n!0!} & \frac{(n-1)!2!}{n!1!} \\
 & & & & & & & \frac{(n-1)!1!}{n!0!}
\end{pmatrix}
\begin{pmatrix}
U_0 \\ U_1 \\ U_2 \\ U_3 \\ \vdots \\ U_{n-3} \\ U_{n-2} \\ U_{n-1}
\end{pmatrix}
$$

$$
= \begin{pmatrix} n^k & (n-1)^k & n-2^k & n-3^k & \cdots & 3^k & 2^k & 1^k \end{pmatrix}'
$$

Computing U_j as $j = n-1, n-2, \cdots, 1, 0$, we have the number of ways that exactly j balls are not touched in k grabs is:

$$
U_j = K_j \cdot \begin{pmatrix} n \\ j \end{pmatrix}
$$

where $K_{n-i} = i^k - \sum_{t=1}^{i-1} K_{n-i+t} \cdot \begin{pmatrix} i \\ t \end{pmatrix}$, and $K_{n-1} = 1$.

Other discussions on similar problems can be found in [35] [51]. Feller [35] gives a more explicit result on U_j:

$$
U_j = \binom{n}{j} \sum_{t=0}^{n-j} (-1)^t \binom{n-j}{t} (n-j-t)^k
$$

Furthermore, Feller provides the formula for computing the probability of having j balls never being grabbed after the k trials.

$$p_j(k,n) = n^{-k} \begin{pmatrix} n \\ j \end{pmatrix} \sum_{v=0}^{n-j} (-1)^v \begin{pmatrix} n-j \\ v \end{pmatrix} (n-j-v)^k \qquad (2.8)$$

We searched for k such that $p_0(k,n) > 0.99$ is greater than 99%, which means that almost every entity has a chance to transmit before the dark slot. We also let $p_1(k,n)$ above a given constant, so that transmissions with collision possibilities are mostly collision-free. Figure 2.4 shows the minimum number of grabs vs. number of balls such that the probability of all balls getting grabbed is greater than 99%.

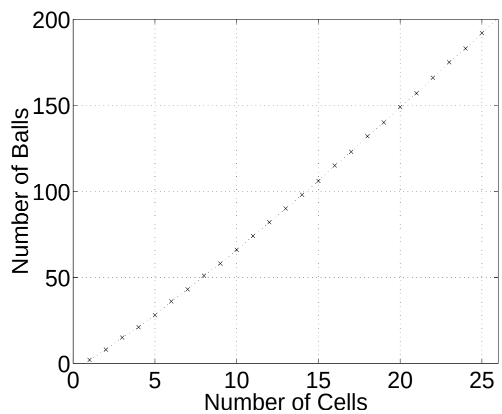


Figure 2.4: Number of grabs vs. number of balls for which the probability of all are grabbed is higher than 99%

Using these results and the number of contenders to an entity, we can derive the size of the period after which we can allocate a dark slot to entities for accessing the shared resource without priority comparison, and yet the probability of collisions is low ($< 1\%$).

We discuss NCR without delay improvement in the rest of the dissertation for simplicity.

2.2 Neighbor Protocol

2.2.1 Random Access with Signals

In ad hoc networks, the two-hop neighbor information needed by topology-dependent scheduling protocols is acquired by each node propagating its one-hop neighbor states. However, exchanging neighborhood information among known and unknown neighbors cannot take advantage of the dynamic collision-free scheduling mechanisms described so far, because those mechanisms assume *a priori* knowledge of the neighborhood. Hence, neighborhood information needs to be transmitted over a common channel on a best-effort basis in omnidirectional mode. The neighbor protocol relies on an additional time section for coordinating neighbor information.

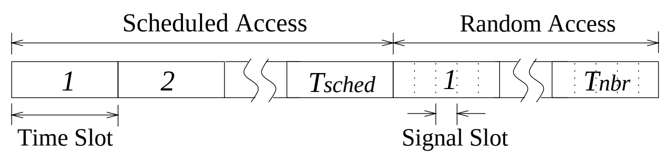


Figure 2.5: Time division scheme: T_{sched} time slots for scheduled channel access are followed by T_{nbr} time slots for random access to send smaller signal frames.

Figure 2.5 shows that the additional time section is inserted after every T_{sched} scheduled-access time slots, and lasts for T_{nbr} time slots. In addition, every time slot for random access is subdivided into a number of smaller time segments, called *signal slots*, for transmitting short signals, each containing up to a few hundreds of bytes. The format of the signal packets depends on the specific control information required by the channel access protocols or the topology management functionalities, including neighbor identification, bandwidth demand, and link status information.

Signals are used by the neighbor protocol for two purposes. One is for a node to say periodically “hello” to its one-hop neighbors in order to maintain connectivity. The other is to send neighbor updates when a neighbor is added, deleted or needs to be refreshed. If a new link is established, both ends of the link need to notify their one-hop neighbors of the new link, and exchange their complete one-hop neighbor information with each other. The weight of a new link is initialized to one.

If a link breaks, a neighbor-delete update needs to be sent out, which is indicated by zero weights assigned to both the incoming and outgoing links with the neighbor. For robustness, an existing neighbor connection also has to be refreshed periodically to the one-hop neighbors. If a neighbor-delete update is not delivered to some one-hop neighbors, those neighbors age out the obsolete link after a period of time.

2.2.2 Signal Transmission Scheduling

To avoid the periodic transmissions of signal packets from synchronizing with one another, which would result in undue collisions of signal packets, the neighbor protocol adds random jitters to the interval value between signal packet transmissions.

However, because of the randomness of signal packet transmissions, it is still possible for a signal sent by a node to collide with signals sent by some of its two-hop neighbors. Due to the lack of acknowledgments in signal transmissions, multiple retransmissions are needed for a node to assure the delivery of the same message to its one-hop neighbors.

Retransmissions of a signal packet can only achieve a certain probability of delivery without acknowledgments. Even though the message delivery probability approaches one as the neighbor protocol sends out the same message in signals repetitively for some time period, the neighbor protocol has to regulate the rhythm with which signals are sent, so that the desired probability of the message delivery is achieved with a small number of retransmissions in a short time, while incurring only a small amount of interference to other neighbors' signal transmissions.

We analyze the time interval and the number of retransmissions needed to achieve a certain probability of message delivery by broadcasting signals.

For simplicity, denote the number of neighbors within two hops by N, the retransmission interval in terms of the number of signal slots by T, the number of retransmissions by n, and the desired probability of message delivery by p. We assume that the signal slots chosen by two-hop neighbors to transmit signals are uniformly distributed over the interval T after a period of time during which the neighbor protocol has been operating. Therefore, the success probability of a transmission is $(1-1/T)^N$. When a single message is retransmitted for n times, the probability p of at least one successful delivery to all one-hop neighbors satisfies the following formula:

$$1 - \left(1 - \left(1 - \frac{1}{T}\right)^N\right)^n = p$$

which gives

$$n = \frac{\ln(1-p)}{\ln\left(1 - \left(1 - \frac{1}{T}\right)^N\right)} \,. \tag{2.9}$$

Hence, the duration of the required retransmissions is represented by the function:

$$f(T) = T \cdot n = \frac{T \ln(1-p)}{\ln\left(1 - \left(1 - \frac{1}{T}\right)^N\right)} \ . \tag{2.10}$$

Because a signal needs to be statistically delivered to one-hop neighbors as soon as possible, the parameter T should be chosen such that $f(T)$ is minimal for given N and p. Let $f'(T) = 0$, we get

$$\frac{1}{\ln(1 - \left(1 - \frac{1}{T}\right)^N)} \cdot \frac{N(1-\frac{1}{T})^N}{1 - \left(1 - \frac{1}{T}\right)^N} \cdot \frac{1}{T-1} = -1 \ , \tag{2.11}$$

which becomes independent of p.

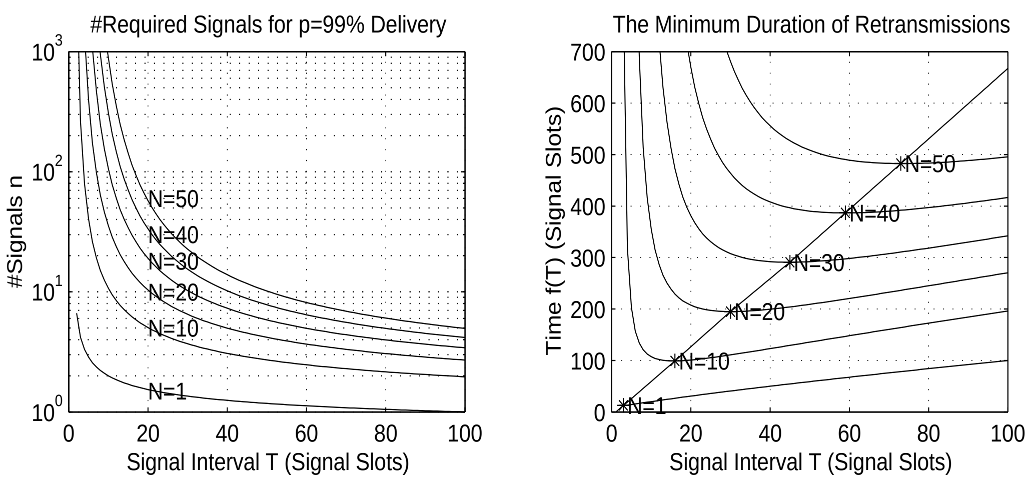


Figure 2.6: The minimum number of retransmissions and the minimum retransmission duration required to successfully deliver signals with probability $p = 0.99$.

To find out the relation between T and N from Eq. (2.11), Eq. (2.9) and (2.10) are plotted in the left and right diagrams of Figure 2.6, respectively, when the required message delivery probability is $p = 0.99$. As shown in the figure, the minimum number and duration of retransmissions required to achieve the desired probability of message delivery are not constant, but vary depending on the interval T chosen to send signals. However, the lowest point on each curve happens at $T \approx 1.44N$, which suggests an approximately *linear relation* between parameter T and N for achieving the desired probability within the shortest time. If we let $t = 1 - \frac{1}{T}$, Eq. (2.11) becomes:

$$Nt^N = (1 - t^N)\left(1 - \frac{1}{t}\right)\ln(1 - t^N) \ .$$

The monotony of the two sides of the equation can be examined if we let

$$\begin{cases} g(t) = Nt^N \ , \\ h(t) = (1 - t^N)\left(1 - \frac{1}{t}\right)\ln(1 - t^N) \ , \end{cases}$$

and take the derivatives of the two functions. Because $g(t)$ monotonically increases ($g'(t) > 0$), and $h(t)$ monotonically decreases ($h'(t) < 0$), there is only one root to the equation $g(t) = h(t)$. That is, Eq. (2.11) has only one root T in the form of N, which means that there is only one minimal point on each curve of the right diagram of Figure 2.6.

Assume that N is large, and $T \approx kN$, Eq. (2.11) becomes

$$\frac{1}{\ln(1 - e^{-1/k})} \cdot \frac{Ne^{-1/k}}{1 - e^{-1/k}} \cdot \frac{1}{kN} + 1 \approx 0 \ ,$$

which can be solved using numeric estimation, and gives $k \approx 1.44$. This reinforces the conjecture that $T \approx 1.44N$, meaning that when the signal transmission interval is 1.44 times the number of neighbors within two hops, the time required to statistically deliver a message to all one-hop neighbors becomes the shortest.

Applying $T \approx 1.44N$ ($N \gg 1$) to Eq. (2.9), n is derived as:

$$n = \frac{\ln(1 - p)}{\ln(1 - \left(1 - \frac{1}{1.44N}\right)^N)} \approx \frac{\ln(1 - p)}{\ln(1 - e^{-\frac{1}{1.44}})} \ ,$$

$$n = 1.45 \ln \frac{1}{1-p} \ , \tag{2.12}$$

and is dependent only on p. When $p = 0.99$, $n = 6.7$ according to Eq. (2.12).

When N is small, a more detailed linear relation between T and N has to be considered, which is $T = 1.44N + 1.55$, derived from the minimum points in the right diagram of Figure 2.6. Taking $T = 1.44N + 1.55$ into Eq. (2.9) and plotting n against N, it appears that n monotonically increases with N. In practice, n takes the derived value from Eq. (2.12) assuming N is large ($N \geq 20$), and T takes value $T = 1.44N + 1.55$ if N is small ($N < 20$) or $T = 1.44N$ otherwise, thus preserving the desired probability of message delivery.

For instance, using the above results, if a node has $N = 20$ neighbors within two hops, then the signal packet interval is set to $T = 1.44N = 29$ signal slots, and the same message has to be retransmitted for $n = 7$ times to achieve 0.99 delivery rate. Accordingly, the duration of the retransmissions is $f(T) = nT \approx 194$ signal slots, matching the result in the right diagram of Figure 2.6.

The interval values have been based on signal slots. As we stated in Section 2.2.1, every T_{sched} time slots for scheduled access are followed by T_{nbr} time slots for random access to send signals. Therefore, the latency of delivering a message with the desired probability depends on three factors: (a) the duration of regular time slots and signal slots, (b) the portion of time for random access, and (c) the channel bandwidth. Because the duration of regular time slots and signal slots are determined by the bandwidth and the sizes of packets carried in these slots, independent of neighbor protocol, we assume the signal-slot duration to be a constant and denote it by t_s.

Then, the portion of random-access sections for achieving a desired latency L for message delivery satisfies:

$$\frac{T_{nbr}}{T_{nbr} + T_{sched}} = \frac{Tnt_s}{L} \ .$$

The more neighbors a node has, the longer the interval value T is set for signal retransmissions and the more the portion of time needed for random access. For instance, if the neighbor protocol is to handle up to a moderate number of neighbors within two hops, such as $N = 20$, the signal slot lasts $t_s = 1ms$, the message delivery desires probability $p = 0.99$ and latency $L = 2s$, then the portion of time for random access overhead should be set in practice equal to

$$\frac{T_{nbr}}{T_{nbr} + T_{sched}} = \frac{1.44N \cdot 1.45 \ln \frac{1}{1-p} \cdot t_s}{L} = 9.6\% \ . \tag{2.13}$$

Chapter 3

Channel Access Scheduling in Ad Hoc Networks with Omnidirectional Antennas

In ad hoc networks, the contention for the channel happens among neighbors within two hops from each node. Using time-slotted channel access, the contention context corresponds to the time slot. Based on NCR and time-division multiple access scheme, four multiple access protocols are derived, which respectively schedule node activation (NAMA) suitable for broadcast communication, link activation (LAMA) and pair-wise link activation (PAMA) for unicast communication, and hybrid activation (HAMA) for both unicast and broadcast communications.

Section 3.1 describes the four scheduling protocols. Section 3.2 derives the channel access probabilities of the four protocols in randomly generated ad hoc networks, and compares the throughput attributes of NAMA and HAMA with those of ideal carrier sensing multiple access (CSMA) and carrier sensing multiple access with collision avoidance. Section 3.3 presents the results of simulations that provide further insight to the performance differences between the four scheduling protocols and the corresponding static scheduling based on UxDMA.

3.1 Channel Access Protocols

In this section, we apply the NCR algorithm to derive four channel access protocols in ad hoc networks with omnidirectional antennas.

3.1.1 Modeling of Network and Contention

We assume that each node is assigned a unique identifier, and is installed with an omnidirectional radio transceiver that is capable of communicating using DSSS (direct sequence spread spectrum) on a pool of well-chosen spreading codes. The radio of each node only works in half-duplex mode, *i.e.*, either transmitting or receiving data packet at a time, but not both.

In multihop wireless networks, signal collisions may be avoided if the received radio signals are spread over different codes or scattered onto different frequency bands. Because the same codes on certain different frequency bands can be equivalently considered to be on different codes, we only consider channel access based on a code division multiple access scheme.

Time is synchronized at each node, and nodes access the channel based on slotted time boundaries. Each time slot is long enough to transmit a complete data packet, and is numbered relative to a consensus starting point. Although global time synchronization is desirable, only limited-scope synchronization is necessary for scheduling conflict-free channel access in multihop ad hoc networks, as long as the consecutive transmissions in any part of the network do not overlap across time slot boundaries. Time synchronization has to depend on physical layer timing and labeling for accuracy, and is outside the scope of this dissertation.

The topology of a packet radio network is represented by a graph $G = (V, E)$, where V is the set of network nodes, and E is the set of links between nodes. Links are bi-directional. A link $(u, v) \in E$ means that node u and v are within the transmission range of each other, so that they can exchange packets via the wireless channel. In this case, node u and v are called *one-hop neighbors* of each other. The set of one-hop neighbors of a node i is denoted by N_i^1. Two nodes are called *two-hop neighbors* of each other if they are not adjacent, but have at least one common one-hop neighbor. The neighbor information of node i refers to the union of the one-hop neighbors of node i itself and the one-hop neighbors of node i's one-hop neighbors, which equals

$$N_i^1 \cup (\bigcup_{j \in N_i^1} N_j^1) \ .$$

In multihop wireless networks, a single radio channel is spatially reused at different parts of the network. The hidden-terminal problem is the main cause of interference and collision in ad hoc networks, and involves nodes within at most two hops. To ensure conflict-free transmissions, it is sufficient for nodes within *two hops* to not transmit on the same time, code and frequency coordinates. Therefore, the topology information within two hops provides the contender information required by the NCR algorithm. When describing the operation of the channel access protocols, we assume that each node already knows its neighbor information within two hops. Chapter 2 describes the neighbor protocol for acquiring this information in mobile ad hoc networks.

3.1.2 Code Assignment

We assume that the physical layer implements the direct sequence spread spectrum (DSSS) transmissions. In DSSS, the code assignments are categorized into transmitter-oriented, receiver-oriented or a per-link-oriented schemes, which are also referred to as TOCA, ROCA and POCA, respectively (*e.g.*, [42] [55]). The four channel access protocols described in this dissertation adopt different code assignment schemes, thus providing different features.

We assume that a pool of well-chosen orthogonal pseudo-noise codes, $C_{pn} = \{c_k \mid k = 0,\ 1, \cdots\}$, is available in the signal spreading function. The spreading code assigned to node i is denoted by $i.\texttt{code}$. During each time slot t, a new spreading code is assigned to node i derived from the priority of node i, using Eq. (3.1).

$$i.\texttt{code} = c_k,\ k = i.\texttt{prio}\ \mathbf{mod}\ |C_{pn}|\ . \tag{3.1}$$

Table 3.1 summarizes the notation used in the dissertation to describe the channel access protocols.

Table 3.1: Notation in NAMA, LAMA, PAMA and HAMA

i.`prio`	The priority of node i.
(u, v).`prio`	The priority of link (u, v).
i.`code`	The code assigned to node i for either reception or transmission.
i.`mode`	The activation mode of node i for either reception or transmission.
Tx	Transmission mode.
Rx	Reception mode.
i.`in`	The transmitter to node i.
i.`out`	The receiver set of node i.
i.`Q`(i.`out`)	The packet queues for the eligible receivers in i.`out`.
N_i^c	The set of one-hop neighbors assigned with code c at node i.
[statement]	A more complex and yet easy-to-implement operation than an atomic statement, such as a function call.

3.1.3 NAMA

The node-activation multiple access (NAMA) protocol requires that the transmission from a node is received by the one-hop neighbors of the node without collisions. That is, when a node is activated for channel access, the neighbors within two hops of the node should not transmit. Therefore, the contender set M_i of node i is the one-hop and two-hop neighbors of node i, which is $N_i^1 \cup (\bigcup_{j \in N_i^1} N_j^1) - \{i\}$.

```
NAMA(i, t)
{
    /* Initialize. */
    M_i = N_i^1 ∪ ( ⋃_{j∈N_i^1} N_j^1 ) − {i};
    for (k ∈ M_i ∪ {i}))
        k.prio = Hash(k ⊕ t) ⊕k;

    /* Resolve nodal mode. */
    if (∀k ∈ M_i,  i.prio > k.prio) {
        i.mode = Tx;
        i.out = N_i^1;
        [ Transmit the earliest packet in i.Q (i.out) ];
    }
    else {
        i.mode = Rx;
        [ Listen to the channel ];
    }
} /* End of NAMA. */
```

Figure 3.1: NAMA Specification.

Figure 3.1 specifies NAMA. Because only node i is able to transmit within its two-hop neighborhood when node i is activated, data transmissions from node i can be successfully received by all of its one-hop neighbor. Therefore, NAMA is capable of collision-free broadcast, and does not require code-division channelization for data transmissions.

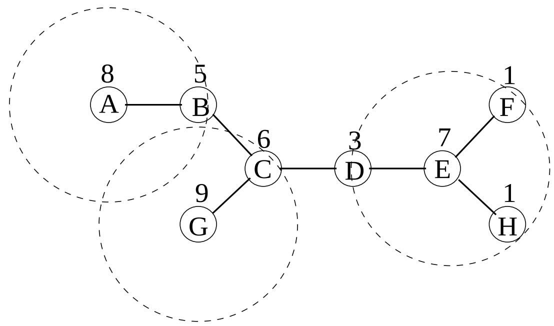


Figure 3.2: An example of NAMA operation.

Figure 3.2 provides an example of how NAMA operates in a multihop network. In the figure, the lines between nodes indicate the one-hop relationship, the dotted circles indicate the effective transmission ranges from nodes, and the node priorities in the current time slot are given beside each node. According to NAMA, there are three nodes A, G and E able to transmit, because their priorities are the highest in their corresponding two-hop neighborhood.

3.1.4 LAMA

In LAMA (link activation multiple access), the code assignment for data transmission is receiver-oriented, which is suitable for unicast using a link-activation scheme. The purpose of LAMA is to determine which node is eligible to transmit, and find out which outgoing link from the node can be activated in the current time slot.

Figure 3.3 specifies LAMA for activating a link from node i in time slot t. Node i first initializes the priority and code assignments of nodes within two hops (lines 1-5), and determines its eligibility to transmit (line 6). If eligible, node i examines each reception code c assigned to its one-hop neighbors, and decides whether node i

LAMA$(i,\ t)$
{
/ Initialize. */*
1 **for** $\left(k \in N_i^1 \cup \left(\bigcup_{j\in N_i^1} N_j^1\right)\right)$
2 $k.\texttt{prio} = \texttt{Hash}(k \oplus t) \oplus k$;
3 $n = k.\texttt{prio}$ **mod** $|C_{pn}|$;
4 $k.\texttt{code} = c_n$;
5 }

/ Resolve nodal mode. */*
6 **if** $(\forall k \in N_i^1,\ i.\texttt{prio} > k.\texttt{prio})$ {
7 $i.\texttt{mode} = \text{Tx}$;
8 $i.\texttt{out} = \emptyset$;
9 **for** $(c : \exists k \in N_i^1,\ c \equiv k.\texttt{code})$ {
10 $M_i = N_i^1 \cup \left(\bigcup_{j\in N_i^c} N_j^1\right) - \{i\}$;
11 **if** $(\forall j \in M_i,\ i.\texttt{prio} > j.\texttt{prio})$
12 $i.\texttt{out} = i.\texttt{out} \cup N_i^c$;
13 }

14 **if** $(\exists k :\ k \in i.\texttt{out}$ **and**
[k has the earliest packet in $i.\texttt{Q}$ $(i.\texttt{out})$])
15 [Transmit the packet in $i.\texttt{Q}$ $(\{k\})$ on $k.\texttt{code}$];
16 }
17 **else** {
18 $i.\texttt{mode} = \text{Rx}$;
19 [Listen to transmissions on $i.\texttt{code}$];
20 }
} */* End of* **LAMA**. **/*

Figure 3.3: LAMA Specification.

can activate links to the one-hop neighbor subset N_i^c, in which all nodes are assigned code c (lines 9-12). Here, the set of contenders of node i is N_i^c and one-hop neighbors of nodes in N_i^c, excluding node i (line 10). Then node i selects and transmits the earliest packet to one of the receivers in $i.\texttt{out}$ (lines 14-15 according to FIFO). If node i is not able to transmit, it listens on the code assigned to itself (lines 17-20).

Figure 3.4 illustrates a contention situation at node i in a time slot. The topology is represented by an undirected graph. The number beside each node repre-

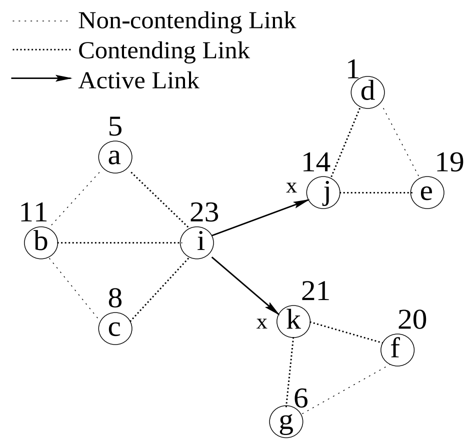


Figure 3.4: An example of LAMA operation.

sents the current priority of the node. Node j and k happen to have the same code x. To determine if node i can activate links on code x, we compare priorities of nodes according to LAMA. Node i has the highest priority among its one-hop neighbors, and higher priority than node j and k as well as their one-hop neighbors. Therefore, node i can activate either (i, j) or (i, k) in the current time slot t depending on the back-logged data flows at node i. In addition, node e may activate link (e, d) if node d is assigned a code other than code x.

3.1.5 PAMA

Figure 3.5 specifies PAMA. PAMA (Pairwise-link Activation Multiple Access) is different from NAMA and LAMA in that the link priorities are used in the contention resolution for channel access, instead of the node priorities. The priority of link (u, v) is computed according to Eq. (3.2), which is an adaptation of Eq. (2.1).

$$(u, v).\texttt{prio} = \texttt{Hash}(u \oplus v \oplus t) \oplus u \oplus v, \tag{3.2}$$

PAMA$(i,\ t)$
{
/* *Initialize.* */
1 **for** $(k \in N_i^1 \cup (\bigcup_{j \in N_i^1} N_j^1))$ {
2 k.`prio` = `Hash`$(k \oplus t) \oplus k$;
3 $n = k$.`prio` **mod** $|C_{pn}|$;
4 k.`code` $= c_n$;
5 }

6 **for** $(k \in N_i^1 \cup \{i\})$ {
/* *Link priorities.* */
7 **for** $(j \in N_k^1)$ {
8 (k, j).`prio` =
`Hash`$(k \oplus j \oplus t) \oplus k \oplus j$;
9 (j, k).`prio` =
`Hash`$(j \oplus k \oplus t) \oplus j \oplus k$;
10 }

11 k.`in` = -1;
12 k.`out` $= \emptyset$;
/* *Active income/outgo link.* */
13 **if** $(\exists j \in N_k^1, \forall u \in N_k^1,$
14 $((j,k)$.`prio` $> (u,k)$.`prio` $\mid u \neq j)$
15 **and** (j,k).`prio` $> (k,u)$.`prio`)
16 k.`in` $= j$;
17 **else if** $(\exists j \in N_k^1, \forall u \in N_k^1,$
18 $((k,j)$.`prio` $> (k,u)$.`prio` $\mid u \neq j)$
19 **and** (k,j).`prio` $> (u,k)$.`prio`)
20 k.`out` $= \{j\}$;

/* *Nodal modes.* */
21 **if** $(i$.`out` $\equiv \{k\}$ **and** k.`in` $\equiv i)$ {
22 i.`mode` = Tx;

/* *Hidden terminal avoidance.* */
23 **if** $(\exists u \in N_i^1 - \{k\},\ u$.`in` $\equiv v,\ v \neq i$
24 **and** i.`code` $\equiv v$.`code` **and**
25 $((v \in N_i^1$ **and** $u \in v$.`out`)
26 **or** $(v \notin N_i^1)))$
27 i.`out` $= \emptyset$;

28 **if** ([There is a packet in i.`Q` (i.`out`)])
29 [Transmit the packet on i.`code`];
30 }
31 **else if** $(i$.`in` $\equiv k)$ {
32 i.`mode` = Rx;
33 [Listen to transmissions on k.`code`];
34 }
} /* *End of* **PAMA**. */

Figure 3.5: PAMA Specification.

Based on the priorities of the incident links to a node, PAMA chooses the link with the highest priority for reception or transmission at the node. Hence, the set of contenders of a link includes all other links incident to the endpoints of the link. PAMA uses a POCA scheme, in which a code is assigned per transmitter-receiver pair. However, because a node can activate only one incident link for either transmission or reception in each time slot, the POCA is equivalent to the transmitter-oriented (TOCA) scheme.

In PAMA, lines 1-5 assign codes to the nodes in the two-hop neighborhood of node i. Then the priorities of the incident links at node i and its one-hop neighbors are computed (lines 7-10). The link with the highest priority at each node is marked for active incoming link (lines 13-16) or active outgoing link (lines 17-20). If node i has an active outgoing link, which is also an active incoming link at the receiver (line 21), node i may be able to transmit, and further examines the hidden terminal problem at other nodes (lines 23-27). If node i does not incur hidden-terminal interference, it selects the packet for the active outgoing link and transmits on i.`code` (lines 28-29). Otherwise, node i listens on the code assigned to the active incoming link (lines 31-34).

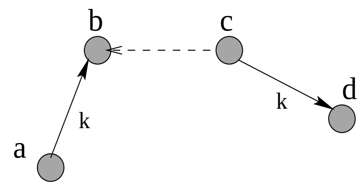


Figure 3.6: An example of hidden terminal problem in PAMA.

Figure 3.6 illustrates a simple 4-node network, in which arrows indicate the transmission directions from nodes. A collision happens at node b when link (a, b) and (c, d) are activated using the same code k. However, node c is able to know the possible collision and deactivate link (c, d) for the current time slot using PAMA lines 23-27 in Figure 3.5.

3.1.6 HAMA

Unlike the previous channel access scheduling protocols that activate either nodes or links only, HAMA (hybrid activation multiple access) is a node-activation channel access protocol that is capable of broadcast transmissions, while also maximizing the chance of link activations for unicast transmissions. The code assignment

Table 3.2: Nodal modes in HAMA

R	*Receiver*: The node has an intermediate priority among its one-hop neighbors.
D	*Drain*: The node has the lowest priority among its one-hop neighbors, and can only receive a packet in the time slot.
BT	*Broadcast Transmitter*: The node has the highest priority within its two-hop neighborhood, and can broadcast to its one-hop neighbors.
UT	*Unicast Transmitter*: The node has the highest priority among its one-hop neighbors, instead of two-hop. Therefore, the node can only transmit to a selected subset of its one-hop neighbors.
DT	*Drain Transmitter*: The node has the highest priority among the one-hop neighbors of a *Drain* neighbor.
Y	*Yield*: The node could have been in either UT- or DT-mode, but chooses to abandon channel access because its transmission may incur unwanted collisions due to potential hidden sources from its two-hop neighbors.

in HAMA is the TOCA scheme.

In each time slot, a node derives its mode by comparing its own priority with the priorities of its neighbors. We require that only nodes with higher priorities transmit to those with lower priorities. Accordingly, HAMA defines the nodal modes as shown in Table 3.2.

Figure 3.7 specifies HAMA. Lines 1-8 compute the priorities and code assignments of the nodes within the two-hop neighborhood of node i using Eq. (2.1) and Eq. (3.1), respectively. Depending on the one-hop neighbor information of node i and

```
HAMA(i, t)
{
    /* Initialized every node as receiver. */
    i.mode = R;
    i.in = -1;
    i.out = ∅;

    /* Priority and code assignments. */
    for (k ∈ N_i^1 ∪ (⋃_{j∈N_i^1} N_j^1)) {
        k.prio = Hash(t ⊕ k);
        n = k.prio mod |C_pn|;
        k.code = c_n;
    }

    /* Find UT and Drain. */
    for (∀j ∈ N_i^1 ∪ {i}) {
        if (∀k ∈ N_j^1, j.prio > k.prio)
            /* j may unicast. */
            j.mode = UT;
        elseif (∀k ∈ N_j^1, j.prio < k.prio)
            /* j is a drain-type node. */
            j.mode = D;
    }

    /* If i is UT, see if i can become BT */
    if (i.mode ≡ UT and (∀k ∈ ⋃_{j∈N_i^1} N_j^1,
        k ≠ i, i.prio > k.prio))
        i.mode = BT;

    /* If i is Receiver, i may become DT. */
    if (i.mode ≡ R and
        ∃j ∈ N_i^1, j.mode ≡ D and
        ∀k ∈ N_j^1, k ≠ i, i.prio > k.prio) {
        i.mode = DT;

        /* Check if i should listen instead. */
        if (∃j ∈ N_i^1, j.mode ≡ UT and
            ∀k ∈ N_i^1, k ≠ j, j.prio > k.prio)
            /* i has a UT neighbor j. */
            i.mode = R;
    }

    /* Find dest for Tx, src for Rx. */
    switch (i.mode) {
      case BT:
          i.out = {-1}; /* Broadcast. */
      case UT:
        for (j ∈ N_i^1)
          if (∀k ∈ N_j^1, i.prio > k.prio)
            i.out = i.out ∪{j};
      case DT:
        for (j ∈ N_i^1)
          if (j.mode ≡ D and (∀k ∈ N_j^1,
              k ≠ i, i.prio > k.prio))
            i.out = i.out ∪{j};
      case D, R:
        if (∃j ∈ N_i^1 and (∀k ∈ N_i^1,
              k ≠ j, j.prio > k.prio)) {
          i.in = j;
          i.code = j.code;
        }
    }

    /* Hidden Terminal Avoidance. */
    if (i.mode ∈ { UT, DT } and
      ∃j ∈ N_i^1, j.mode ≠ UT and
      ∃k ∈ N_j^1, k.prio > i.prio and
      k.code ≡ i.code)
      i.mode = Y;

    /* Ready to communicate. */
    switch (i.mode) { /* FIFO */
      case BT:
        if (i.Q(i.out) ≠ ∅)
          pkt = Earliest pkt in i.Q(i.out);
        else
          pkt = Earliest pkt in i.Q(N_i^1);
        Transmit pkt on i.code;
      case UT, DT:
        pkt = Earliest pkt in i.Q(i.out);
        Transmit pkt on i.code;
      case D, R:
        Receive pkt on i.code;
    }
} /* End of HAMA. */
```

Figure 3.7: HAMA Specification.

node $j \in N_i^1$, node i classifies the status of node j and itself into the receiver (R or D) or transmitter (UT) mode (lines 9-14).

If node i happens to be a unicast transmitter (UT), then i further checks whether it can broadcast by comparing its priority with those of its two-hop neighbors (lines 15-17). If node i is a *Receiver* (R), it checks whether it has a neighbor j in *Drain* mode (D) to which it can transmit instead (lines 18-21). If yes, node i needs to make sure that it is not receiving from any one-hop neighbor before it becomes the *drain* transmitter (DT) (lines 22-25).

After that, node i decides its receiver set if it is in transmitter mode (BT, UT or DT), or its sources if in receiver mode (R or D). A receiver i always listens to its one-hop neighbor with the highest priority by tuning its reception code into that neighbor's transmission code (lines 26-44).

If a transmitter i unicasts (UT or DT), the hidden-terminal problem should be avoided, in which case node i's one-hop receiver may be receiving from two transmitters on the same code (lines 45-49).

Finally, node i in transmission mode may send the earliest arrived packet (FIFO) to its receiver set i.`out`, or listens if it is a receiver (lines 50-62). In case of the broadcast mode (BT), node i may choose to send a unicast packet if its broadcast buffer is empty.

Figure 3.8 provides an example of how HAMA operates in a multihop network during a time slot. In the figure, the priorities are noted beside each node. Node A has the highest priority among its two-hop neighbors, and becomes a broadcast transmitter

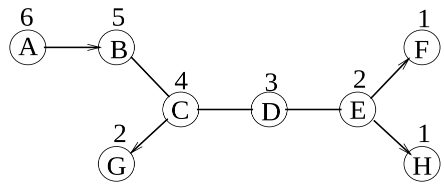


Figure 3.8: An example of HAMA operation.

(BT). Nodes F, G and H are receivers in the *drain* mode, because they have the lowest priorities among their one-hop neighbors. Nodes C and E become transmitters to *drain*s, because they have the highest priorities around their respective *drain*s. Nodes B and D stay in *receiver* mode because of their low priorities. Notice that in this example, only node A would be activated in NAMA, because node C would defer to node A, and node E would defer to node C. This illustrates that HAMA can provide better channel access opportunities over NAMA, although NAMA does not requires code-division channelization.

In contrast to NAMA, HAMA provides similar broadcasting capability, in addition to the extra opportunities for sending unicast traffic with only a little more processing required on the neighbor information.

3.2 Throughput Analyses

In a fully-connected network, it comes naturally that the channel bandwidth is evenly shared among all nodes using any of the above channel access protocols, because the priorities of nodes or links are uniformly distributed. However, in an ad hoc network model where nodes are randomly placed over an infinite plane, bandwidth allocation to a node is more generic, and much more complex. We first analyze the accurate channel access probabilities of HAMA and NAMA, then the upper bound of the channel access probability of PAMA and LAMA. The throughput of NAMA and HAMA is compared with that of the ideal CSMA and collision-avoidance protocols, which are analyzed in [80] and [76],

For simplicity, we assumed that infinitely many codes are available, such that hidden terminal collision on the same code was not considered.

3.2.1 Geometric Modeling

Similar to the network modeling in [80] and [76], the network topology is generated by randomly placing many nodes on an infinitely large two-dimensional area independently and uniformly, where the node density is denoted by ρ. The probability of having k nodes in an area of size S follows a Poisson distribution, namely:

$$p(k,S) = \frac{(\rho S)^k}{k!} e^{-\rho S} \ .$$

The mean of the number of nodes in the area of size S is ρS.

Based on this modeling, the channel access contention of each node, is related with the node density ρ and node transmission range r. Let N_1 be the average number of one-hop neighbors covered by the circular area under the radio transmission range of a node, we have $N_1 = \rho \pi r^2$.

Let N_2 be the average number of neighbors within two hops. As shown in Figure 3.9, two nodes become two-hop neighbors only if there is at least one common neighbor in the shaded area. The average number of nodes in the shaded area is:

$$B(t) = 2\rho r^2 a(t) \ ,$$

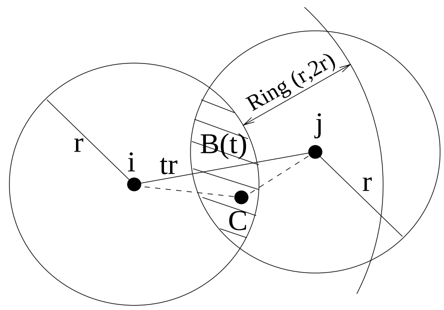


Figure 3.9: Becoming two-hop neighbors.

where

$$a(t) = \arccos\frac{t}{2} - \frac{t}{2}\sqrt{1-\left(\frac{t}{2}\right)^2}\ . \tag{3.3}$$

Thus, the probability of having at least one node in the shaded area is $1 - e^{-B(t)}$. Adding up all nodes covered by the ring $(r,\ 2r)$ around the node, multiplied by the corresponding probability of becoming two-hop neighbors, the average number of two-hop neighbors of a node is:

$$\rho\pi r^2 \int_1^2 2t\left(1-e^{-B(t)}\right) dt\ ,$$

which results in:

$$N_2 = N_1\left(1 + \int_1^2 2t\left(1-e^{-B(t)}\right) dt\right)\ .$$

For convenience, the symbols $T(N)$, $U(N)$ and $W(N)$ are introduced to denote three probabilities when the average number of contenders is N.

$T(N)$ denotes the probability of a node winning among its contenders. Because the number of contenders follows Poisson distribution with mean N, and that all nodes have equal chances of winning, the probability $T(N)$ is the average over all possible numbers of the contenders:

$$T(N) = \sum_{k=1}^{\infty} \frac{1}{k+1} \frac{N^k}{k!} e^{-N} = \frac{e^N - 1 - N}{Ne^N} \ .$$

Note that k starts from 1 in the expression for $T(N)$, because a node with no contenders does not win at all. $U(N)$ is the probability that a node has at least one contender, which is simply

$$U(N) = 1 - e^{-N}.$$

$W(N)$ is introduced to denote

$$W(N) = U(N) - T(N) \ .$$

3.2.2 NAMA

Because N_2 denotes the average number of two-hop neighbors, which is the number of contenders for each node in NAMA, it follows that the probability that the node broadcasts is $T(N_2)$. Therefore, the channel access probability of a node in NAMA is

$$q_{NAMA} = T(N_2) \ . \tag{3.4}$$

3.2.3 HAMA

HAMA includes the node activation cases in NAMA in the broadcast mode (BT). In addition, HAMA provides two more modes for a node to transmit in the unicast mode (UT and DT). Overall, if node i transmits in the unicast mode (UT and DT), node i must have at least one neighbor j, of which the probability is

$$p_u = U(N_1) \ .$$

In addition, the chances of unicast transmissions in either the UT or the DT modes depend on three factors: (a) the number of one-hop neighbors of the source, (b) the number of one-hop neighbors of the destination, and (c) the distance between the source and destination.

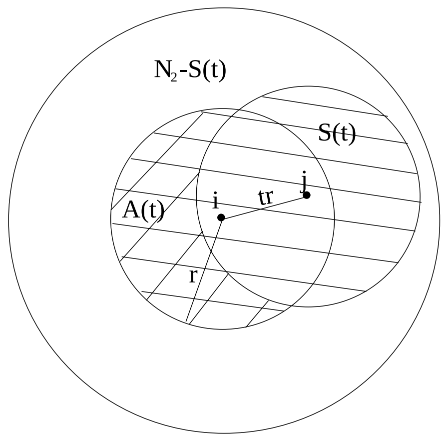


Figure 3.10: The unicast between two nodes.

First, we consider the probability of unicast transmissions from node i to node j in the UT mode, in which case, node i contend with nodes residing in the combined one-hop coverage of nodes i and j, as illustrated in Figure 3.10. Given that the transmission range is r and the distance between nodes i and j is tr $(0 < t < 1)$, we denote the number of nodes within the combined coverage by k_1 excluding nodes i and j, of which the average is

$$S(t) = 2\rho r^2 \left[\pi - a(t)\right] \ .$$

$a(t)$ is defined in Eq. (3.3). Therefore, the probability of node i winning in the combined one-hop coverage is:

$$p_1 = \sum_{k_1=0}^{\infty} \frac{1}{k_1+2} \frac{S(t)^{k_1}}{k_1!} e^{-S(t)} = \frac{W(S(t))}{S(t)} \ .$$

Furthermore, because node i cannot broadcast when it enters the UT mode, there has to be at least one two-hop neighbor with higher priority than node i outside the combined one-hop coverage in Figure 3.10. Denote the number of nodes outside the coverage by k_2, of which the average is $N_2 - S(t)$. The probability of node i losing outside the combined coverage is thus:

$$p_2 = \sum_{k_2=1}^{\infty} \frac{[N_2 - S(t)]^{k_2}}{k_2!} e^{-(N_2 - S(t))} \frac{k_2}{k_2+1} = W(N_2 - S(t)) \ .$$

In all, the probability of node i transmitting in the UT mode is:

$$p_3 = p_1 \cdot p_2 = \frac{W(N_2 - S(t)) \ W(S(t))}{S(t)} \ .$$

The probability density function (PDF) of node j at position t is $p(t) = 2t$. Therefore, integrating p_3 on t over the range $(0, 1)$ with PDF $p(t) = 2t$ gives the average probability of node i becoming a transmitter in the UT mode:

$$p_{UT} = \int_0^1 p_3 2t dt = \int_0^1 2t \frac{W(N_2 - S(t)) \ W(S(t))}{S(t)} \ dt \ .$$

Second, we consider the probability of unicast transmissions from node i to node j in the DT mode. We denote the number of one-hop neighbors of node j by k_3, excluding nodes i and j, of which the average is N_1. Then, node j requires the lowest priority among its k_3 neighbors to be a *drain*, and node i requires the highest priority to transmit to node j, of which the average probability over all possible values of k_3 is:

$$p_4 = \sum_{k_3=0}^{\infty} \frac{N_1^{k_3}}{k_3!} e^{-N_1} \frac{1}{k_3+2} \frac{1}{k_3+1} = \frac{T(N_1)}{N_1} \ .$$

In addition, node i has to lose to nodes residing in the side lobe, marked by $A(t)$ in Figure 3.10. Otherwise, node i would enter the UT mode. Denote the number of nodes in the side lobe by k_4, of which the average is

$$A(t) = 2\rho r^2 \left[\frac{\pi}{2} - a(t)\right] .$$

The probability of node i losing in the side lobe is thus

$$p_5 = \sum_{k_4=1}^{\infty} \frac{A(t)^{k_4}}{k_4!} e^{-A(t)} \frac{k_4}{k_4+1} = W(A(t)) .$$

In all, the probability of node i entering the DT mode for transmission to node j is the product of p_4 and p_5. That is:

$$p_6 = p_4 \cdot p_5 = \frac{T(N_1)}{N_1} W(A(t)) .$$

Using the PDF $p(t) = 2t$ for node j at position t, the integration of the above result over the range $(0, 1)$ gives the average probability of node i entering the DT mode, denoted by p_{DT}:

$$p_{DT} = \int_0^1 p_6 2t dt = \frac{T(N_1)}{N_1} \int_0^1 2t\ W(A(t))\ dt .$$

In summary, the average channel access probability of a node in the network is the chance of becoming a transmitter in the three mutually exclusive broadcast or unicast modes (BT, UT or DT), which is given by

$$\begin{aligned} q_{HAMA} &= q_{NAMA} + p_u(p_{UT} + p_{DT}) \\ &= T(N_2) + U(N_1) \cdot \left(\frac{T(N_1)}{N_1} \int_0^1 2tW(A(t))dt + \int_0^1 2t\frac{W(N_2 - S(t))W(S(t))}{S(t)}dt\right) . \end{aligned} \tag{3.5}$$

The above analysis for HAMA has made four simplifications. Firstly, we assumed that the number of two-hop neighbors also follows Poisson distribution, just like that of one-hop neighbors. Secondly, we let $N_2 - S(t) \geq 0$ even though N_2 may be smaller than $S(t)$ when the transmission range r is small. Thirdly, only one neighbor j is considered when making node i to become a unicast transmitter in the DT or the UT mode, although node i may have multiple chances to do so owning to other one-hop neighbors. The results of the simulation experiments reported in Section 3.3 validate these approximations.

3.2.4 PAMA

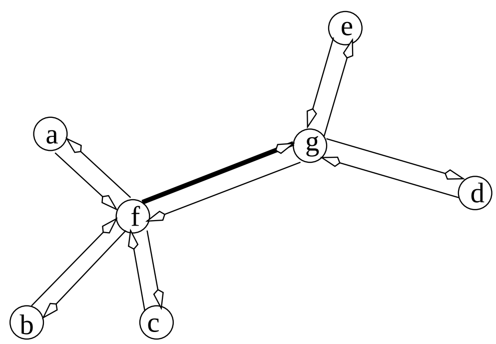


Figure 3.11: Link Activation in PAMA.

In PAMA, a link is activated only if the link has the highest priority among the incident links of the head and the tail of the link. For example, in Fig. 3.11, link (f, g) is activated only if it has the highest priority among the links with f and g as the heads or tails.

To analyze the channel access probability of a node in PAMA, we simplify the problem by assuming that the one-hop neighbor sets of the one-hop neighbors of a given node are disjoint (*i.e.*, and that any two-hop neighbor of a node is reachable through a single one-hop neighbor only). Using the simplification, the sizes of the two-hop neighbor sets become identical independent random variables following Poisson distribution with mean N_1.

Suppose that a node i has $k_1 \geq 1$ one-hop neighbors. The probability that the node is eligible for transmission is $k_1/2k_1 = 1/2$ because the node has $2k_1$ incident links, and k_1 of them are outgoing. Further suppose that link (i,j) out of the k_1 outgoing links has the highest priority, then node i is able to activate link (i,j) if link (i,j) also has the highest priority among the links incident to node j. Denote the number of one-hop neighbors of node j by k_2. Then the probability of link (i,j) having the highest priority among the incident links of node j is a conditional probability, based on the fact that link (i,j) already has the highest priority among the incident links of node i.

We denote the conditional probability of link (i,j) having the highest priority among the incident links of node j as $P\{A \mid B\}$, where A is the event that link (i,j) wins among the $2k_2$ incident links of node j, and B is the event that link (i,j) wins among the $2k_1$ incident links of node i. We have:

$$P\{B\} = \frac{1}{2k_1},\ P\{A \cap B\} = \frac{1}{2k_1 + 2k_2}\ ,$$

$$P\{A \mid B\} = \frac{P\{A \cap B\}}{P\{B\}} = \frac{k_1}{k_1 + k_2}\ .$$

Therefore, the condition of node i being able to transmit is that node i has an outgoing link (i,j) with the highest priority, of which the probability is $\frac{1}{2}$, and that link (i, j) has the highest priority among the incident links of node j, of which the probability is $\frac{k_1}{k_1+k_2}$. Considering all possible values of random variables k_1 and k_2, which follow a Poisson distribution, and we have:

$$
\begin{aligned}
q_{PAMA} &= \sum_{k_1=1}^{\infty} \frac{N_1^{k_1}}{k_1!} e^{-N_1} \frac{1}{2} \sum_{k_2=0}^{\infty} \frac{N_1^{k_2}}{k_2!} e^{-N_1} \frac{k_1}{k_1+k_2} \\
&= \frac{1}{2} e^{-2N_1} \sum_{k_1=1}^{\infty} \sum_{k_2=0}^{\infty} \frac{N_1^{k_1}}{k_1!} \frac{N_1^{k_2}}{k_2!} \frac{k_1}{k_1+k_2} \\
&= \frac{N_1}{2} e^{-2N_1} \sum_{k_1=1}^{\infty} \sum_{k_2=0}^{\infty} \frac{N_1^{k_1-1}}{(k_1-1)!} \frac{N_1^{k_2}}{k_2!} \frac{1}{k_1+k_2} \\
&= \frac{N_1}{2} e^{-2N_1} \sum_{k=0}^{\infty} \frac{(2N_1)^k}{k!} \frac{1}{k+1} \\
&= \frac{N_1}{2} (e^{-2N_1} + T(2N_1)).
\end{aligned} \tag{3.6}
$$

q_{PAMA} is the upper bound of the channel access probability of a node in PAMA, because we have assumed that the one-hop neighbor sets of the head and tail of a link are disjoint. If the one-hop neighbor sets of the head and tail of a link are overlapped, the number of one-hop neighbors of the tail of the activated link, k_2, could have started from a larger number than 0 in the expressions above, and the actual channel access probability in PAMA would be less than q_{PAMA}.

3.2.5 LAMA

In LAMA, a node can activate an outgoing link only if the node has the highest priority among its one-hop neighbors, as well as among its two-hop neighbors reachable through the tail of the outgoing link. For convenience, we make the same assumption as in the analysis of PAMA that the one-hop neighbor sets of the one-hop neighbors of a given node are disjoint.

Similarly, suppose a node i has k_1 one-hop neighbors, and the number of the two-hop neighbors reachable through a one-hop neighbor j is k_2. The probability of node i winning in its one-hop neighbor set N_i^1 is $1/(k_1+1)$. The probability of node i winning in the one-hop neighbor set of node j is $(k_1+1)/(k_1+k_2+1)$, which is conditional upon the fact that node i already wins in N_i^1, and is derived in the same way as in the PAMA analysis. Because k_2 is a random variable following the Poisson distribution,

$$p_7 = \sum_{k_2=0}^{\infty} \frac{N_1^{k_2}}{k_2!} e^{-N_1} \frac{k_1+1}{k_1+k_2+1}$$

is the average conditional probability of node i activating link (i,j). Besides node j, node i has other one-hop neighbors. If node i has the highest priority in any one-hop neighbor set of its one-hop neighbors, node i is able to transmit. Therefore, the probability of node i being able to transmit is

$$p_8 = 1 - (1-p_7)^{k_1}.$$

Because k_1 is also a random variable following the Poisson distribution, the channel access probability of node i in LAMA is:

$$p_9 = \sum_{k_1=1}^{\infty} \frac{N_1^{k_1}}{k_1!} e^{-N_1} \frac{1}{k_1+1} p_8 \ .$$

When k_1 increases, p_8 edges quickly towards the probability limit 1. Since we are only interested in the upper bound of channel access probability in LAMA, assuming $p_8 = 1$ simplifies the calculation of p_9 and provides a less tight upper bound. Let $p_8 = 1$, the upper bound of channel access probability in LAMA is thus:

$$q_{LAMA} = \sum_{k_1=1}^{\infty} \frac{N_1^{k_1}}{k_1!} e^{-N_1} \frac{1}{k_1+1} = T(N_1) \tag{3.7}$$

3.2.6 Comparison among NAMA, HAMA, PAMA and LAMA

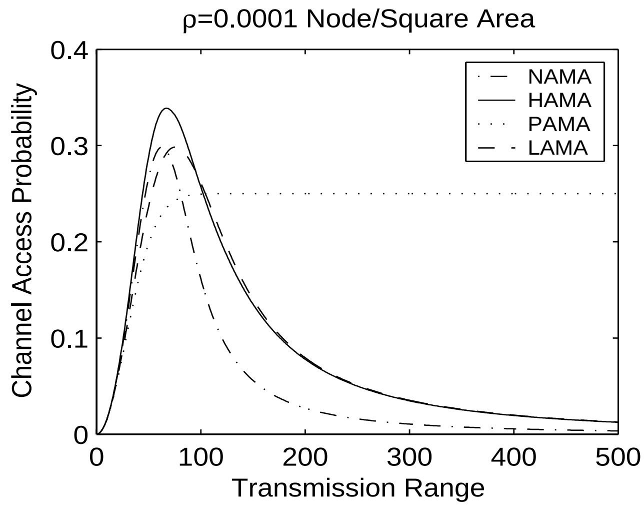


Figure 3.12: Channel access probability of NAMA, HAMA, PAMA and LAMA.

Assuming a network density of $\rho = 0.0001$, equivalent to placing 100 nodes on a 1000×1000 square plane, the relation between transmission range and the channel access probability of a node in NAMA, HAMA, PAMA and LAMA is shown in Figure 3.12, based on Eq. (3.4), Eq. (3.5), Eq. (3.6) and Eq. (3.7), respectively.

Because a node barely has any neighbor in a multihop network when the node transmission range is too short, Figure 3.12 shows that the system throughput is close to none at around zero transmission range, but it increases quickly to the peak when the transmission range covers around one neighbor on the average, except for that of PAMA, which is an upper bound. Then network throughput drops when more and more neighbors are contacted and the contention level increases.

Figure 3.13 shows the performance ratio of the channel access probabilities of HAMA, PAMA and LAMA to that NAMA. At shorter transmission ranges, HAMA,

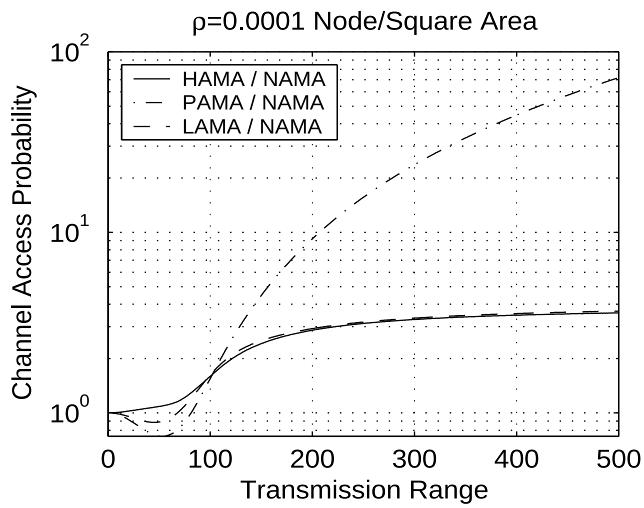


Figure 3.13: Channel access probability ratio of HAMA, PAMA and LAMA to NAMA.

PAMA and LAMA performs very similar to NAMA, because nodes are sparsely connected, and node or link activations are similar to broadcasting. When transmission range increases, HAMA, LAMA and PAMA obtain more and more opportunities to leverage its unicast capability and the relative throughput also increases more than three times that of NAMA. HAMA and LAMA exhibit very similar performance.

3.2.7 Comparison with CSMA and Collision-Avoidance

Because the analyses about NAMA and PAMA are more accurate than the analyses of PAMA and LAMA, which simply derive the upper bounds, we only compare the throughput of HAMA and NAMA to that of the idealized CSMA and collision-avoidance protocols, which are analyzed in [80] and [76]. We consider only unicast transmissions, because collision-avoidance does not support collision-free broadcast.

Scheduled access protocols are modeled differently from CSMA and collision-avoidance. In time-division scheduled channel access, a time slot can carry a complete data packet, while the time slot for CSMA and collision-avoidance only lasts for the duration of a channel round-trip propagation delay, and multiple time slots are used to transmit a data packet once the channel is successfully acquired. In addition, Wang *et al.* [76] and Wu *et al.* [80] assumed a heavily loaded scenario in which a node always has a data packet during the channel access, which is not true for the throughput analyses of HAMA and NAMA, because using the heavy load approximation would always result in the maximum network capacity according to Eq. (2.7).

The probability of channel access at each time slot in CSMA and collision-avoidance is parameterized by the symbol p'. For comparison purposes, we assume that *every attempt* to access the channel in CSMA or collision-avoidance is an *indication* of a packet arrival at the node. Though the attempt may not succeed in CSMA and collision-avoidance due to packet or RTS/CTS signal collisions in the common channel, and end up dropping the packet, conflict-free scheduling protocols can always deliver the packet if it is offered to the channel. In addition, we assume that no packet arrives during the packet transmission. Accordingly, the traffic load for a node is equivalent to the portion of time for transmissions at the node. Denote the average packet size as l_{data}. Because the interval between successive transmissions follows a Geometric distribution with parameter p', the traffic load for a node is given by

$$\lambda = \frac{l_{data}}{1/p' + l_{data}} = \frac{p' l_{data}}{1 + p' l_{data}} \ .$$

The network throughput is measured by the successful data packet transmission rate within the one-hop neighborhood of a node in [76] [80], instead of the whole network. Therefore, the comparable network throughput in HAMA and NAMA is the sum of the packet transmissions by each node and all of its one-hop neighbors. We reuse the symbol N in this section to represent the number of one-hop neighbors of a node, which is the same as N_1 defined in Section 3.2.1. Because every node is assigned the same load λ, and has the same channel access probability (q_{HAMA}, q_{NAMA}), the throughput of HAMA and NAMA becomes

$$S_{HAMA} = N \cdot \min(\lambda,\ q_{HAMA})\ .$$

$$S_{NAMA} = N \cdot \min(\lambda,\ q_{NAMA})\ .$$

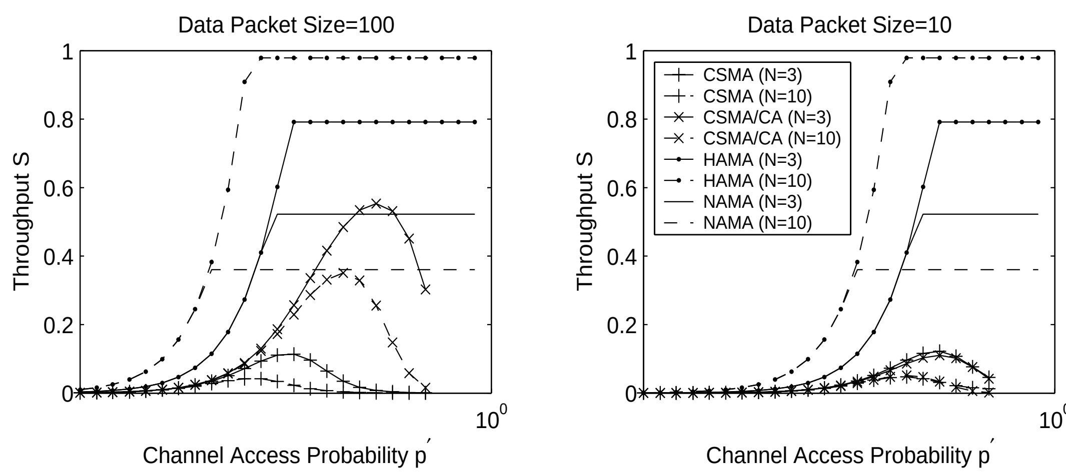


Figure 3.14: Comparison between HAMA, NAMA and CSMA, Collision-Avoidance.

Figure 3.14 compares the throughput attributes of HAMA, NAMA, the idealized CSMA [80], and collision-avoidance protocols [76] with different numbers of one-hop neighbors in two scenarios. The first scenario assumes that data packets last for $l_{data} = 100$ time slots in CSMA and collision-avoidance, and the second assumes a 10-time-slot average packet size.

The network throughput decreases when a node has more contenders in NAMA, CSMA and collision-avoidance protocols, which is not true for HAMA. HAMA and NAMA provide higher throughput than CSMA and collision-avoidance, because all transmissions are collision-free even when the network is heavily loaded. In contrast to the critical role of packet size in the throughput of CSMA and collision-avoidance, it is almost irrelevant in that of scheduled approaches, except for shifting the points of reaching the network capacity.

3.3 Simulations

The delay and throughput attributes of NAMA, LAMA, PAMA and HAMA are studied by comparing their performance with UxDMA [65] in two simulation scenarios: fully connected networks with different numbers of nodes, and multihop networks with different radio transmission ranges.

In the simulations, we use the normalized *packets per time slot* for both arrival rates and throughput. This metric can be translated into concrete throughput metrics, such as *Mbps* (megabits per second), if the time slot sizes and the channel bandwidth are instantiated.

Because the channel access protocols based on NCR have different capabilities regarding broadcast and unicast, we only simulate unicast traffic at each node in all protocols. All nodes have the same load, and the destinations of the unicast packets at each node are evenly distributed over all one-hop neighbors.

In addition, the simulations are guided by the following parameters and behavior:

- The network topologies remain static during the simulations to examine the performance of the scheduling algorithms only.
- Signal propagation in the channel follows the free-space model and the effective range of the radio is determined by the power level of the radio. Radiation energy outside the effective transmission range of the radio is considered negligible interference to other communications. All radios have the same transmission range.
- Each node has an unlimited buffer for data packets.
- 30 pseudo-noise codes are available for code assignments, *i.e.*, $|C_{pn}| = 30$.
- Packet arrivals are modeled as Poisson arrivals. Only one packet can be transmitted in a time slot.
- The duration of the simulation is 100,000 time slots, long enough to collect the metrics of interests.

We note that assuming static topologies does not favor NCR-based channel access protocols or UxDMA, because the same network topologies are used. Nonetheless, exchanging the full topology information required by UxDMA in a dynamic network would be far more challenging that exchanging the identifiers of nodes within two hops of each node.

Table 3.3: Constraint Sets For NCR-Based Protocols.

Protocol	Entity	Constraint Set
UxDMA-NAMA	Node	$\{V_{tr}^0, V_{tt}^1\}$
UxDMA-LAMA	Link	$\{E_{rr}^0, E_{tr}^0\}$
UxDMA-PAMA	Link	$\{E_{rr}^0, E_{tt}^0, E_{tr}^0, E_{tr}^1\}$

Except for HAMA, which schedules both node- and link-activations, UxDMA has respective constraint sets for NAMA, LAMA and PAMA. Table 3.3 gives the corresponding constraint sets for NAMA, LAMA and PAMA.

The meaning of each symbol is illustrated by Figure 3.15. Using the solid dots as transmitters, and the circles as receivers, node constraint V_{tr}^0 forbids a node from transmitting and receiving at the same time, and V_{tt}^1 eliminates hidden terminal problem and direct interference. Using wide lines to denote activated lines, and thin lines to denote interferences, link constraints E_{rr}^0, E_{tt}^0 and E_{tr}^0 restrict concurrent receptions, concurrent transmissions and simultaneous transmission and reception at a single node, respectively. Constraint E_{tr}^1 prevents the hidden-terminal problem in the link activation scheme.

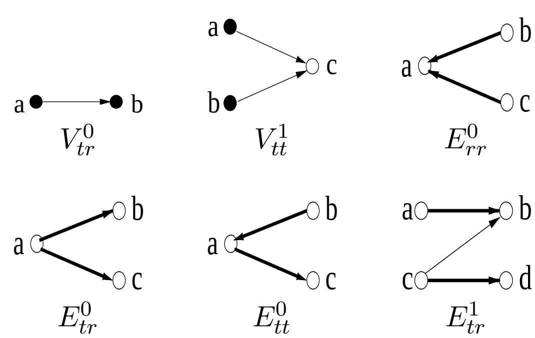


Figure 3.15: Constraints used by UxDMA for channel access scheduling.

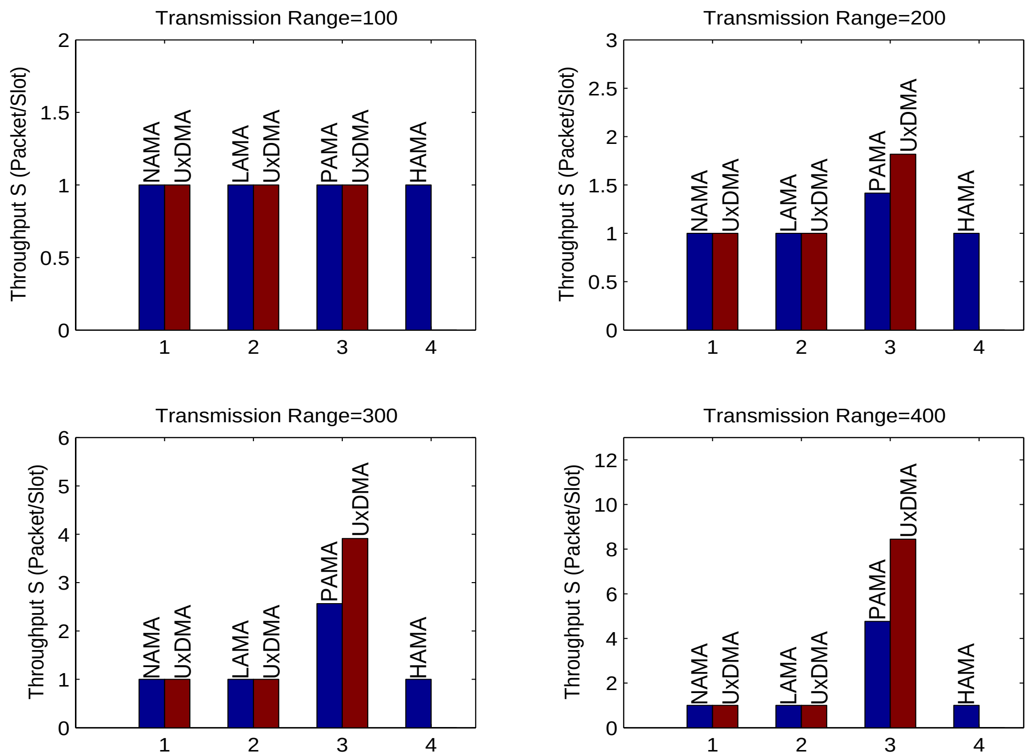


Figure 3.16: Packet throughput in fully-connected networks

Simulations were carried out in four configurations in the fully connected scenario: 2-, 5-, 10-, 20-node networks, to manifest the effects of different contention levels. Figure 3.16 shows the maximum throughput of each protocol in fully-connected networks. Except for PAMA and UxDMA-PAMA, the maximum throughput of every other protocol is one because their contention resolutions are based on the node priorities, and only one node is activated in each time slot. Because PAMA schedules link activations based on link priorities, multiple links can be activated on different codes in the fully-connected networks, and the channel capacity is greater in PAMA than in the other protocols.

Figure 3.17 shows the average delay of data packets in NAMA, LAMA and

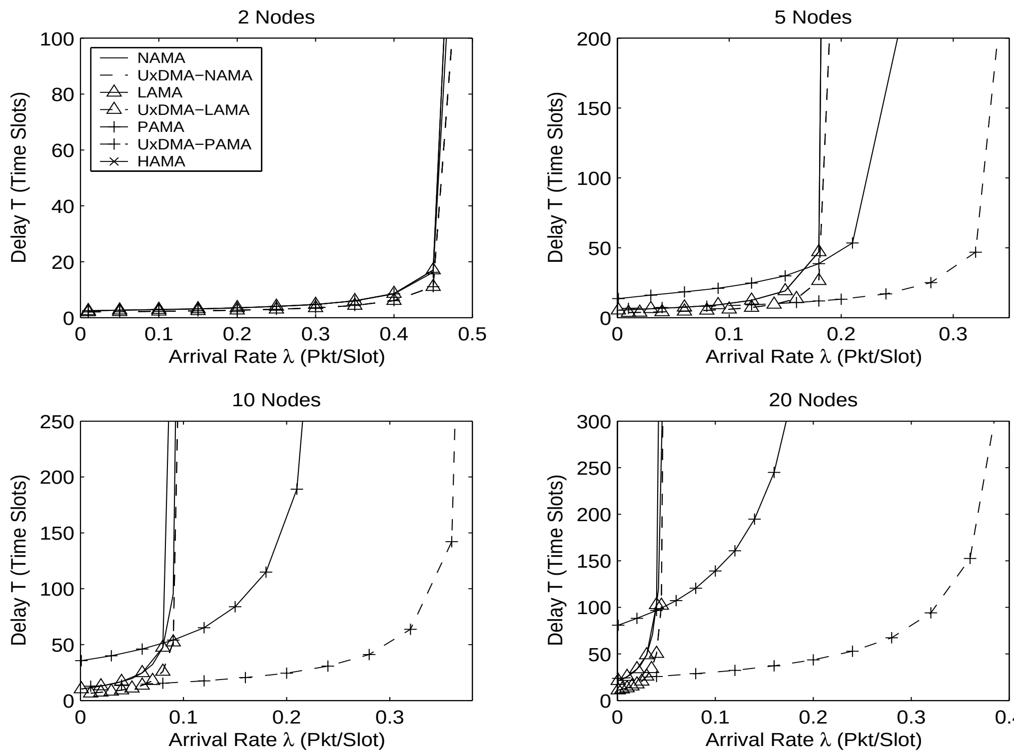


Figure 3.17: Average packet delays in fully-connected networks

PAMA with their corresponding UxDMA counterparts, and HAMA with regard to different loads on each node in fully-connected networks. NAMA, UxDMA-NAMA, LAMA, UxDMA-LAMA and HAMA have the same delay characteristic, because of the same throughput is achieved in these protocols. PAMA and UxDMA-PAMA can sustain higher loads and have longer "tails" in the delay curves. However, because the number of contenders for each link is more than the number of nodes, the contention level is higher for each link than for each node. Therefore, packets have higher starting delay in PAMA than other NCR-based protocols. Figure 3.18 and 3.19 show the throughput and the average packet delay of NAMA, LAMA, PAMA, HAMA and the UxDMA variations.

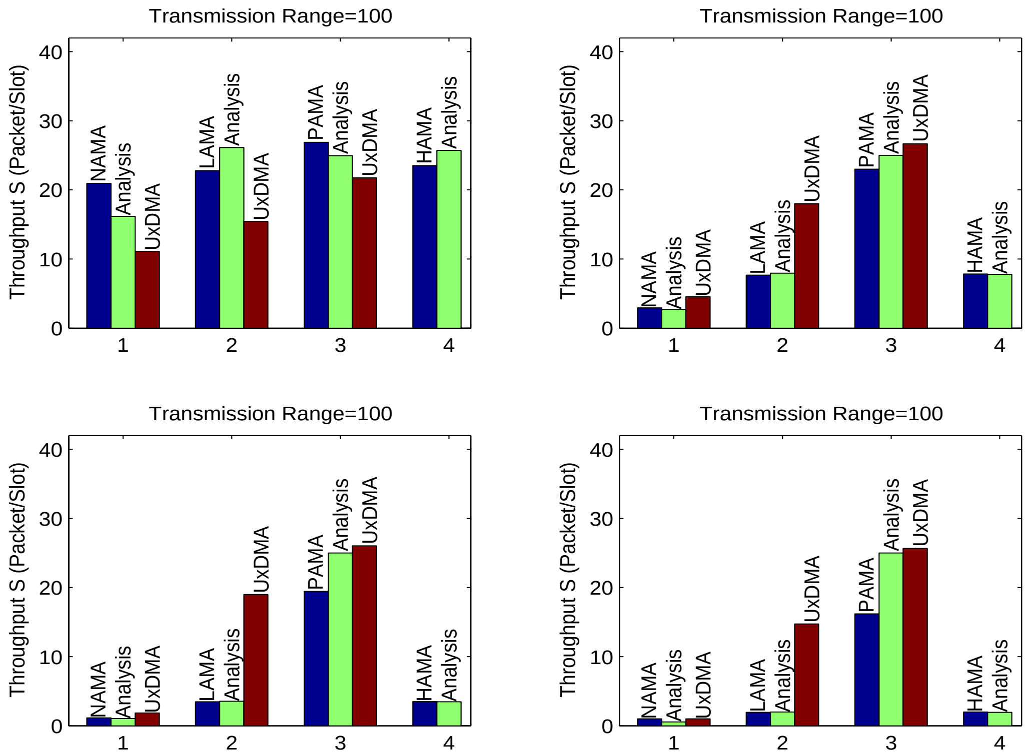


Figure 3.18: Packet throughput in multihop networks

Except for the ad hoc network generated using a transmission range of one hundred meters in Figure 3.18, UxDMA always outperforms its NCR-based counterparts — NAMA, LAMA and PAMA at various levels. For example, UxDMA-NAMA is only slightly better than NAMA in all cases, and UxDMA-PAMA is 10-30% better than PAMA. LAMA is comparatively the worst, with much lower throughput than its counterpart UxDMA-LAMA. One interesting point is the similarity between the throughput of LAMA and HAMA, which has been shown by Figure 3.14 as well, even though they have different code assignment schemes and transmission schedules. Especially, the network throughput of NAMA, LAMA, PAMA and HAMA based on Eq. (2.7) and the analyses in Section 3.2 is compared with the corresponding protocols in

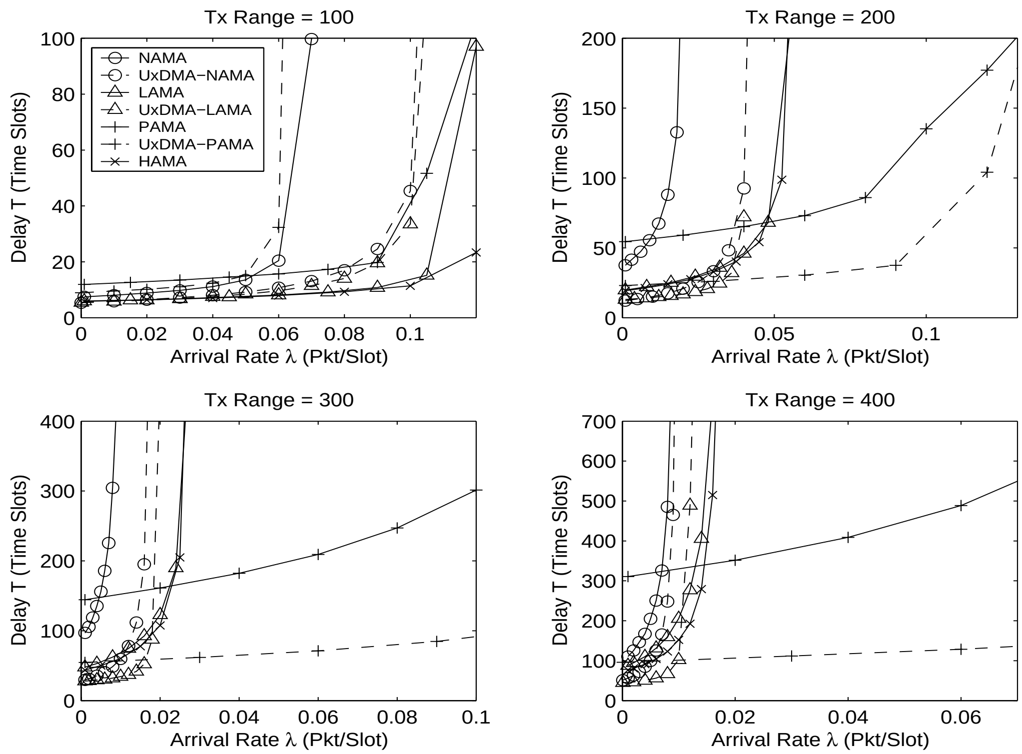


Figure 3.19: Average packet delays in multihop networks

the simulations. The analytical results fits well with the simulations results. Note that the analysis bars with regard to PAMA and LAMA are the upper bounds, although the analysis of LAMA is very close to the simulation results.

In Figure 3.19, PAMA still gives higher starting point to delays than the other two even when network load is low due to similar reasons as in fully connected scenario. However, PAMA appears to have slower increases when the network load goes larger, which explains the higher spectrum and spatial reuse of the common channel by pure link-oriented scheduling.

3.4 Conclusion

Based on NCR, time- and code-division channel access schemes, we have introduced four new approaches to channel access in ad hoc networks with omnidirectional antennas, which allow for both node-activation and link-activation channel access methods. The advantages of the protocols are that (a) they do not need the contention phases or schedule broadcasts, as adopted by many other channel access scheduling algorithms; (b) they only need the local topology information within two hops, as opposed to other schedule broadcasting algorithms that require the complete network topology. We have provided analyses to the four protocols and compared them with the random-access protocols. The performance of these protocols was also compared with the static scheduling algorithms, which require global topology information.

Chapter 4

Channel Access Scheduling in Ad Hoc Networks with Unidirectional Links

Unidirectional links can occur in wireless networks and mixed-media networks. Communication in ad hoc networks with unidirectional links encounters difficulties at the routing control layer and the channel access control layer, because the vast majority of routing algorithms and channel access protocols proposed to date require bidirectional links to operate. This chapter introduces a new family of collision-free channel access protocols for ad hoc networks with unidirectional links based on NCR, called PANAMA (Pair-wise link Activation and Node Activation Multiple Access). PANAMA supports collision-free broadcasting and unicasting, without repetitious schedule adjustments due to network topology changes or global topology information.

4.1 Topology Assumptions

The topology of a network with unidirectional links is modeled by a directed graph $G = (V, A)$, where V is the set of nodes, and A is the set of directional links between nodes, i.e., $A \subseteq V \times V$. We assume that each node in the multihop packet radio network has a unique identifier, and is installed an omnidirectional radio transceiver that works only in the half-duplex mode. We assume each node already knows the topology information within two hops through the neighbor protocol introduced in Chapter 2.

A link $(u, v) \in A$ means that node v is within the radio transmission range of node u and that a possible data transmission channel exists from node u to node v. Link $(u, v) \in A$ does not necessarily mean that link $(v, u) \in A$ in unidirectional networks.

If link $(u, v) \in A$, nodes u and v are called the *head* and *tail* of the link, respectively, and node u is called the *upstream neighbor* of node v, while node v is the *downstream neighbor* of node u. In addition, link (u, v) is called an *outgoing link* of node u, and an *incoming link* of node v. We denote the set of upstream and downstream neighbors of node i as U_i and D_i, respectively. Two non-adjacent nodes sharing the same one-hop neighbor are called *two-hop neighbor* of each other.

A unidirectional link is first detected by the tail of the link, and its existence is propagated back to the head of the link. Hence, there is causal asymmetric relation about the knowledge of a unidirectional link at the head and tail of the link. For instance, if link $(u, v) \in A$, then node $v \in D_u$ implies node $u \in U_v$, but not the opposite.

The existence of inclusive cycle for a link indicates the usability of a unidirectional link for data forwarding purpose [8] [10]. The *inclusive cycle* of a link is defined as the smallest cycle that includes the link in the network.

4.2 PANAMA

PANAMA is a distributed multiple access control protocol that combines two channel access scheduling algorithms based on a time-slotted code-division multiple access scheme using direct sequence spread spectrum (DSSS) transmission techniques. The first scheduling algorithm used in PANAMA is NAMA-UN (Node Activation Multiple Access for Unidirectional Networks), which is a node-activation oriented channel access algorithm suitable for broadcasting in wireless networks with unidirectional links. The second scheduling algorithm in PANAMA is PAMA-UN (Pair-wise link Activation Multiple Access for Unidirectional Networks), which is a link-activation oriented channel access control algorithm suitable for unicasting in wireless networks with unidirectional links.

In both NAMA-UN and PAMA-UN, a node is in the *receiving mode* when it does not win the contention to access the channel during a time slot. It listens to the traffic in the channel by tuning its reception code to the potential transmitter. In NAMA-UN, the potential transmitter is an upstream neighbor that has the highest node-priority among the upstream neighbor set. In PAMA-UN, the potential transmitter is the head of an incoming link with the highest link-priority among all the incoming links.

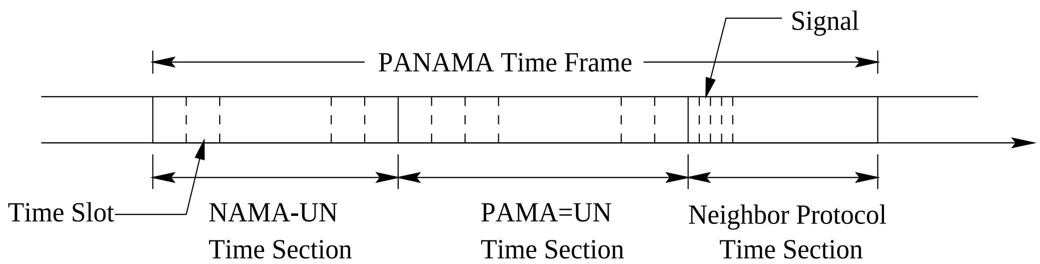


Figure 4.1: NAMA-UN and PAMA-UN time section allocations in PANAMA

PANAMA combines NAMA-UN and PAMA-UN to support unicast and broadcast traffic efficiently. Deciding on what portion of the channel to assign to each protocol is a pragmatic decision that depends on expected traffic patterns in the network. In this chapter, NAMA-UN and PAMA-UN are allocated fixed portion of the time slots as illustrated in Figure 4.1. The time section dedicated to NAMA-UN is denoted by T_{nama}, and that to PAMA-UN is denoted by T_{pama}, each represents the number of assigned time slots, respectively. Accordingly, broadcast traffic always waits for the NAMA-UN section, while unicast traffic is sent during the PAMA-UN section. If broadcast traffic is not present when a NAMA-UN time slot is assigned, unicast packets may be sent, instead. Figure 4.1 also indicated the addition time slot allocation for the neighbor protocol.

We use the same notation as in Table 3.1 to describe PANAMA.

4.2.1 Network Assumptions

As we have stated in previous chapters, code assignment in a network using DSSS (direct sequence spread spectrum) can be based on a transmitter-oriented code assignment (also known as TOCA), receiver-oriented code assignment (ROCA) or a per-link-oriented code assignment (POCA). Both NAMA-UN and PAMA-UN use TOCA for channel coding.

POCA and ROCA schemes are not chosen because of following reasons.

- Because a node can only transmit or receive at one time on a single code, it is unnecessary to assign different codes to links incident to a single node as in a POCA scheme.

- As Figure 4.2 illustrates pathologically, simply using two-hop topology information at each node is insufficient to resolve collisions in a network with unidirectional links using a ROCA scheme. In Figure 4.2, the number beside each node gives the current receive-code assigned to the node. A unidirectional link (b, c) partitions the network. If links (b, a) and (e, c) is activated, neighbor node b nor node e is able to detect possible collision at node c because b is unaware of node c, while node e does not know about node a.

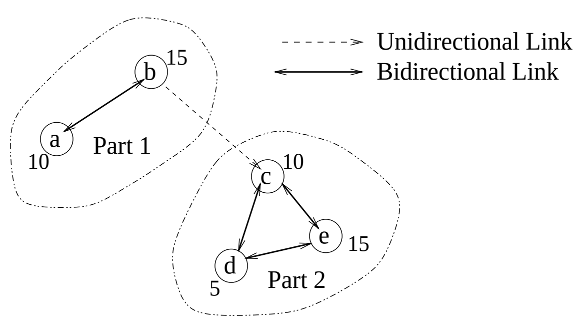


Figure 4.2: Irresolvable Situation in ROCA

We assume that a pool of quasi-orthogonal pseudo-noise codes, $C_{pn} = \{c_k\}$, are available for code assignment to the nodes. The code for each node is computed in each time slot so that the contention situation is different from time slot to time slot. A pseudo-noise code $i.\texttt{code}$ from C_{pn} is assigned to a node i in time slot t according to the following algorithm:

$$i.\texttt{code} = c_k, k = \texttt{Hash}(i \oplus t) \ \texttt{mod} \ |C_{pn}|. \tag{4.1}$$

where $\texttt{Hash}(x)$ is an integer pseudo-random number generator that produces a random integer using input x as the random seed.

In PANAMA, links are assigned a bandwidth property that indicates if the link can be used for data forwarding purpose. The bandwidth property of a link is chosen from value 0 or 1. When a link is initially detected by its tail, the bandwidth of the link is set to 0 by the tail. After the inclusive cycle of the link is found at the head of the link, the bandwidth property is set to 1. This implies that the bandwidth property of a link remains 0 if a link has no inclusive cycle in the network. We denote the bandwidth of link (u, v) by $(u, v).\texttt{bw}$.

The goal of channel access at node i is to send data packets to a subset of downstream neighbors to which node i has positive outgoing link bandwidth, and is defined as:

$$R_i = \{k \mid k \in D_i, (i, k).\texttt{bw} \equiv 1\} \ .$$

R_i is called the *receiver set* of node i.

4.2.2 NAMA-UN

Figure 4.3 specifies NAMA-UN, which decides whether a node i can transmit in a time slot t, such that its receiver set receives the data packet without collisions.

```
NAMA-UN(i, t)
{
    /* Initialize. */
    for (k ∈ U_i ∪ D_i ∪ (⋃_{j∈U_i∪D_i} U_i ∪ D_i)) {
        k.prio = Hash(k ⊕ t) ⊕k;
        n = k.prio mod |C_pn|;
        k.code = c_n;
    }

    M_i = U_i ∪ R_i ∪ (⋃_{k∈R_i} U_k);

    /* Resolve nodal mode. */
    if (∀k ∈ M_i, i.prio > k.prio and
            ∄u, v : v ∈ R_i, u ∈ U_v, (u, v).bw ≡ 0, i.code ≡ u.code) {
        i.mode = Tx;
        i.out = R_i;
        [ Transmit the earliest packet in i.Q (i.out) ];
    }
    else {
        i.mode = Rx;
        if (∃j ∈ U_i, ∀k ∈ U_i, j.prio > k.prio)
            [ Listen to the channel on k.code ];
    }
} /* End of NAMA-UN. */
```

Figure 4.3: NAMA-UN Specification.

Accordingly, the contenders for node i are of the following three kinds:

1. The receiver set of node i, R_i;

2. All of i's upstream neighbors, U_i;

3. All upstream neighbors of nodes in i's receiver set, i.e., $\bigcup_{k\in R_i} U_k$

Therefore, the set of contenders for node i is:

$$M_i = U_i \cup R_i \cup \left(\bigcup_{k\in R_i} U_k \right) .$$

NAMA-UN is similar to NAMA, which was designed for networks without unidirectional link. Unlike NAMA, NAMA-UN requires code division channelization.

First, NAMA-UN initializes the priorities and code assignments to the known one- and two-hop neighbors (lines 1-5). NAMA-UN line 6 finds the contender set of node i. Lines 7-8 in NAMA-UN avoids possible hidden terminal conflicts at the receiving node v from node v's upstream neighbor u when node u is assigned the same code as node i's and does not know about link (u, v) due to the asymmetric properties of unidirectional links.

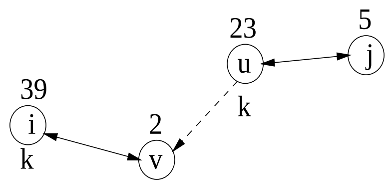


Figure 4.4: Contention Resolution in NAMA-UN

Figure 4.4 illustrates an example of collision avoidance in NAMA-UN. The numbers beside each node are the current priorities of the nodes, and k is the code assigned to node u and node i. Though both node i and node u can transmit on code k according to the priority comparisons in NAMA-UN, node i will be deactivated in order to avoid collisions at node v because node u does not know the existence of link (u, v) and can inadvertently collide with node i.

4.2.3 PAMA-UN

Eq. 4.2 provides the formula for computing the priority of link (u, v).

$$(u, v).\texttt{prio} = (u, v).\texttt{bw} \cdot (\texttt{Hash}(u \oplus v \oplus t) \oplus u \oplus v), \tag{4.2}$$

```
PAMA-UN(i, t)
{
    /* Initialize. */
    for (k ∈ U_i ∪ D_i ∪ (⋃_{j∈U_i∪D_i} U_i ∪ D_i)) {
        k.prio = Hash(k ⊕ t) ⊕k;
        n = k.prio mod |C_pn|;
        k.code = c_n;
    }

    for (k ∈ U_i ∪ D_i ∪ {i}) {
        /* Link priorities. */
        for (j ∈ D_k)
            (k, j).prio = (k, j).bw·(Hash(k ⊕ j ⊕ t) ⊕k ⊕ j);

        for (j ∈ U_k)
            (k, j).prio = (j, k).bw·(Hash(j ⊕ k ⊕ t) ⊕j ⊕ k);

        k.in = -1;
        k.out = ∅;

        /* Active incoming or outgoing link. */
        if (∃j ∈ U_k, (∀u : u ∈ U_k, u ≠ j, (j, k).prio > (u, k).prio) and
            (∀u ∈ D_k, (j, k).prio > (k, u).prio))
            k.in = j;
        else if (∃j ∈ D_k, (∀u : u ∈ D_k, u ≠ j, (k, j).prio > (k, u).prio) and
            (∀u ∈ U_k, (k, j).prio > (u, k).prio))
            k.out = {j};
    }

    /* Nodal modes. */
    if (i.out ≡ {k} and k.in ≡ i) {
        i.mode = Tx;

        /* Hidden terminal avoidance. */
        if ((∃u, v : u ∈ R_i − {k}, v ∈ U_u, (v, u).bw ≡ 0, v.code ≡ i.code) or
            (∃u, v : u ∈ R_i − {k}, v ∈ U_u, (v, u).bw ≡ 1, v.code ≡ i.code,
                v.prio > i.prio))
            i.out = ∅;

        if ([ There is a packet in i.Q (i.out) ])
            [ Transmit the packet on i.code ];
    }
    else if (i.in ≡ k) {
        i.mode = Rx;
        [ Listen to transmissions on k.code ];
    }
} /* End of PAMA-UN. */
```

Figure 4.5: PAMA Specification.

With synchronized information about local topologies and bandwidth allocations, PAMA-UN decides whether a node i can activate one of its outgoing links in a time slot t. Figure 4.5 specifies PAMA-UN.

In PAMA-UN, a node i initializes the priority and code assignments for the known downstream and upstream one- and two-hop neighbors in time slot t (PAMA-UN lines 1-5). Then node i computes the priorities for the incident links of the known one-hop neighbors (PAMA-UN lines 6-10). According to the link priorities, node i chooses the incident link that has the highest link priority to each one-hop neighbor and itself (PAMA-UN 11-19). If the link with the highest priority among the incident links of node i is an outgoing link, and the tail of the link also has the link as the highest-priority link incoming, node i may activate the link (PAMA-UN lines 20-21). However, node i needs to avoid hidden-terminal problem from upstream neighbors of the receiver, which is the tail of the selected link (PAMA-UN lines 22-25). If node i can still transmit, and the packet queue for the receiver is not empty, node i transmits a packet (PAMA-UN lines 26-28). If node i can not transmit, it needs to listen to the head of the incoming link with the highest priority among node i's incident links.

PAMA-UN encounters similar hidden-terminal problems as NAMA-UN. Using the sample network in Figure 4.4 with the same transmission code assignments as an example, collision happens at node v if link (i, v) and (u, j) are activated simultaneously on code k. PAMA-UN deactivates link (i, v) for the current time slot.

4.2.4 Performance

The delay and throughput attributes of PANAMA are studied by simulations in static network topologies with unidirectional links. Many configurable parameters in the protocol are simplified, such as the durations of different sections.

The simulations are guided by the following parameters and behaviors. The parameter values used are simply examples.

- The networks are generated by randomly placing 100 nodes within an area of 1000×1000 square meters. To simulate an infinite plane that has constant node placement density, the opposite sides of the square are seamed together, which visually turns the square area into a torus.

- Signal propagation in the channel follows the free-space model and the effective range of the radio is determined by the power level of the radio. The transmission range of the radios in each simulation scenario is randomly chosen from the range $[150m, 300m]$, which creates unidirectional links.

- To simulate deactivated or unusable unidirectional links in the network, we assume that 10% of the network links are assigned 0 bandwidth, while all other links and all nodes are assigned bandwidth 1.

- The data rate of the radio channel is 2 Mbps. A time unit in the simulation equals one time slot. A time slot lasts 8 milliseconds, including guard time, long enough to transmit a 2KB packet.

- The time section for NAMA-UN T_{nama} contains 25 time slots, and the time section for PAMA-UN T_{pama} contains 95 time slots. The neighbor protocols takes 5 time slots for coordinating neighborhood information. Therefore, the time frame for PANAMA is equal to 1 second.

- In NAMA-UN and PAMA-UN, 30 pseudo-noise codes are available for code assignments, *i.e.*, $|C_{pn}| = 30$.

- All nodes have the same packet arrival rate λ, and p_b is the proportion of the broadcast traffic. The destinations of the unicast packets in PAMA-UN are distributed on all outgoing links with positive bandwidth, proportional to the probability of activating the link.

- Packets are served in First-In First-Out (FIFO) order.

- The duration of the simulation is 800 seconds (equal to 100000 time slots), long enough to compute the metrics of interest.

Four main factors influence the system delay and throughput attributes of PANAMA, namely:

- λ: the data traffic load on each node;

- p_b: the portion of broadcast traffic in the overall traffic;

- p_i: the portion of inactive unusable directional links that only interfere with other transmissions;

Table 4.1: Four Scenarios and Their Parameters

Scenario	Values of Fixed Parameters	Variable
1	p_b=0.05, p_i=0.1, r=100	λ
2	λ=0.06, p_i=0.1, r=100	p_b
3	λ=0.06, p_b=0.05, p_i=0.1	r
4	λ=0.06, p_b=0.05, r=100	p_i

- r: the radio transmission ranges that affect the contention levels at each node.

To manifest the effects of these different parameters on the system delay and throughput, we carry out four sets of simulations by fixing three of the four factors and variate the remaining factor in each simulation to examine the operations of PANAMA. The fixed factors of each simulation do not overly stress the network delays or throughput so as to manifest the effects of the variable factor. Thus, we obtain four scenarios described in Table 4.1.

Scenario 1 simulates a unidirectional network where the broadcast traffic is $p_b = 5\%$ of the overall packet arrivals, unusable links are $p_i = 10\%$ of the network links, nodal transmission range is randomly selected over $r = 100$ to $r = 2 \times 100 = 200$ meters, and the packet arrival rate is variable over certain range. Similarly, scenario 2 varies the broadcast traffic percentage while fixing the other three factors, *i.e.*, the packet arrival rate λ, unusable link ratio p_i and transmission range r. Scenario 3 simulates the transmission range r variations, and scenario 4 simulates the variations of unusable link ratio p_i. These scenarios examine the effects of packet arrival rate, broadcast traffic ratio, transmission range and unusable link ratio on the packet delay and network throughput performance, respectively.

Table 4.2: Constraint sets in UxDMA for NAMA-UN and PAMA-UN

Section	Colored Object	Constraint Set
Broadcast	Node	$\{V_{tr}^0, V_{tt}^1\}$
Unicast	Link	$\{E_{rr}^0, E_{tt}^0, E_{tr}^0, E_{tr}^1\}$

In addition, we simulate the static scheduling algorithm, UxDMA, specified in [65], for comparison with PANAMA in the same simulation scenarios with as many similar parameters as possible, such as percentage of inactive links and channel divisions into NAMA-UN, PAMA-UN and signal sections, where UxDMA uses a different coloring schemes for the corresponding NAMA-UN and PAMA-UN section. The criteria for coloring nodes in the broadcast section and links in the unicast section are given by Table 4.2.

The meaning of each symbol is referred to the original paper on UxDMA [65]. The constraint E_{tr}^1 in UxDMA eliminates hidden terminal problem. Furthermore, to make a fair comparison between UxDMA and PANAMA, nodes in UxDMA are assigned transmission codes so that constraint E_{tr}^1 is allowed when transmitters have different transmission codes.

In UxDMA, because the coloring on nodes and links is closely coupled with code assignments, the code assignments are carried out only once at the beginning of each simulation, and remain static throughout the simulation as well as the color assignments. Nodes are assigned codes randomly chosen from the code base C_{pn}. In addition, inactive links in the network topology are not colored but are taken into account as possible interference sources when links and nodes are colored in UxDMA.

The number of colors used by UxDMA determines the time frame during which every entity is able to access the channel once. Because UxDMA interleaves broadcast section and unicast section similarly as in PANAMA, a time frame for UxDMA-NAMA or UxDMA-PAMA may not fit evenly into one time-slot section. If a time frame is not finished in either NAMA-UN or PAMA-UN section, it continues in the upcoming section of the same type (broadcast or unicast).

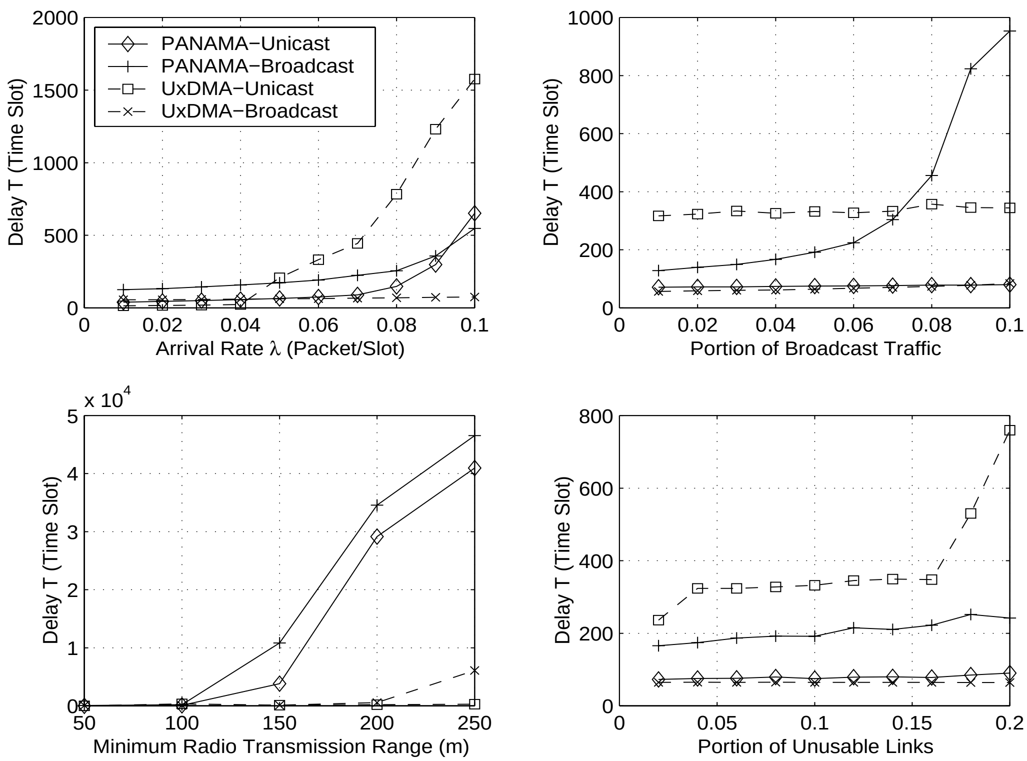


Figure 4.6: Average Packet Delays In Multihop Networks

Figure 4.6 and 4.7 show the delay and throughput attributes of PANAMA and UxDMA in multihop networks of the four scenarios, described by the parameters provided in Table 4.1.

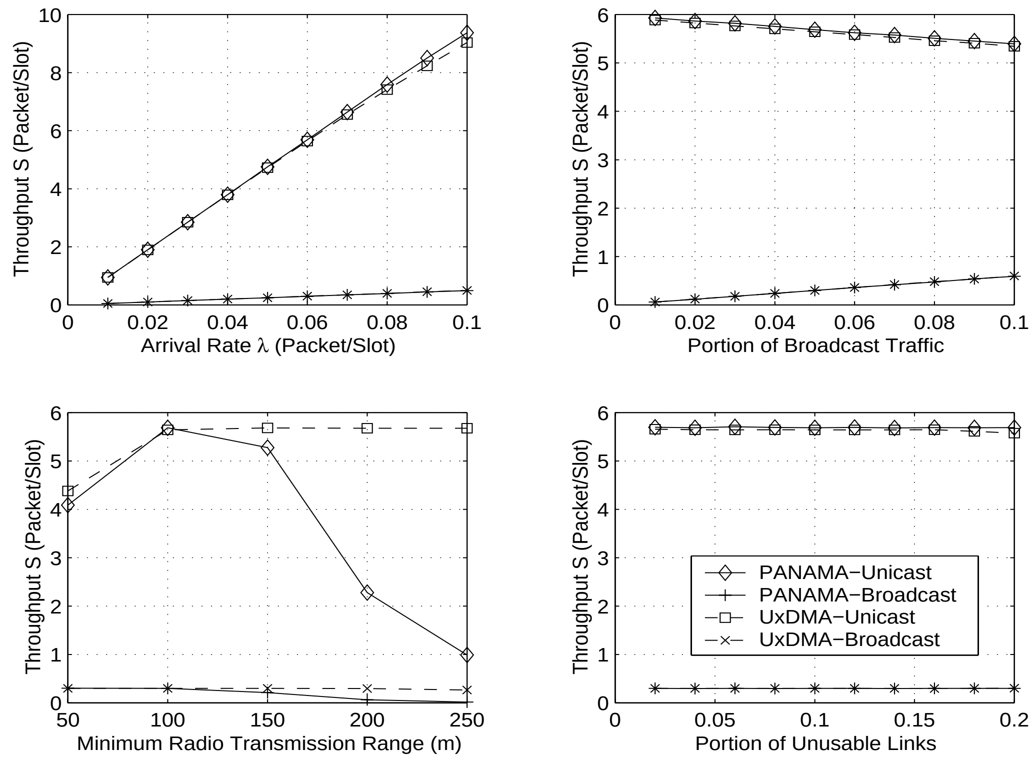


Figure 4.7: Packet Throughput Of Multihop Networks

In all scenarios except scenario 3, unicast traffic has lower delays in PANAMA than in UxDMA; however, PANAMA performs worse than UxDMA for broadcast traffic. Likewise, the network throughput is almost the same in both PANAMA and UxDMA when the network load remains at sustainable levels. In scenario 3, however, both unicast and broadcast traffics suffer much longer delay and worse network throughput when the transmission ranges increase in PANAMA. This is because of higher contention levels from longer radio transmission ranges that increase the probability of code assignment conflicts between two-hop neighbors, which lead to many aborted transmission due to collision avoidance.

Overall, NAMA-UN performs worse than its counterpart of static scheduling, while PAMA-UN is better than the static link scheduling algorithm in UxDMA. However, PAMA-UN cannot sustain its performance under high contention levels in multihop networks due to conflicts in code assignment. With power control and topology control algorithms that modulate the number of one-hop neighbors of each node, PAMA-UN gives the best mechanisms for data transmission in mobile environments.

PANAMA does have disadvantages in that the intervals between successive transmissions by a single entity is governed by a geometric distribution, which introduces positive probability of long delays to deliver packets.

4.3 Conclusion

We have introduced PANAMA for channel access scheduling in networks with unidirectional links, based on the neighbor-aware contention resolution (NCR) algorithm, introduced in Chapter 2. PANAMA is the first channel access protocol that provides collision-free multiple access to the shared channel in ad hoc networks with unidirectional links. PANAMA incorporates both node-activation and link-activation scheduling in the channel access. It was shown that PANAMA is also capable dynamic bandwidth allocation for topology control and resource management in ad hoc networks that contain unidirectional links.

Chapter 5

Channel Access Scheduling in Ad Hoc Networks with Directional Antennas

This chapter presents a new channel access protocol based on a link activation scheme, which we call Receiver-Oriented Multiple Access (ROMA), to fully utilize the concurrent transmission or reception capability of multi-beam adaptive array (MBAA) antennas. Section 5.1 introduces assumptions and relevant terminologies for ad hoc networks with MBAA antennas. Section 5.2 specifies ROMA. Unlike most random access protocols for directional antennas that form only a single beam, both transmissions and receptions are carried out in the directional mode of the antennas in ROMA. ROMA adopts the neighbor-aware contention resolution algorithm (NCR) introduced in Chapter 2 to derive channel access schedules for a node. Section 5.3 addresses the performance of ROMA and compares it against UxDMA by simulation experiments.

ROMA offers four key advantages over prior approaches to the channel access problem. First, ROMA allows both transmitters and receivers to use the directional mode of the antenna, instead of requiring one end of the communication to stay in omnidirectional mode, as adopted by prior random access schemes. Second, ROMA relies on the local topology information within two hops for computing the channel access schedules, in contrast to the reliance on global topology information in UxDMA. Third, ROMA evenly splits nodes in the network into transmitters and receivers in each time slot, which are then paired together for the maximum throughput. Whereas UxDMA allocates each link only one time slot for activation per time frame, ROMA may activate the link multiple times during the same period. Fourth, ROMA is the most efficient distributed channel access control mechanism for ad hoc networks with directional antennas that are capable of forming multiple antenna beams.

5.1 Network Assumptions

5.1.1 Directional Antenna System

Dipole or isotropic antennas propagate radio frequency (RF) energy equally in horizontal or spherical directions. In contrast, directional antennas use multiple antenna elements so that individual omnidirectional RF radiations from these elements interfere constructively or destructively with each other in space, and the signal strength is increased in one or multiple directions. Antenna gain measures the increase of signal strength in those directions in decibels over either a dipole (dBd) or a theoretical isotropic (dBi) antenna. Relative to the center of the antenna pattern, the angle of the directions where the radiated power drops to one-half the maximum value of the lobe is defined as the antenna beamwidth, denoted by β in this chapter. The beamwidth can be as narrow as 5° to 10° [75] depending on the number of elements and their spacing.

With the advance of silicon and DSP technologies, DSP modules in real-life directional antenna systems with multiple antenna elements can combine more than one set of weights to form several antenna patterns simultaneously [77]. Because radio reception and transmission are reciprocal, any directivity pattern achievable for reception is also achievable for transmission.

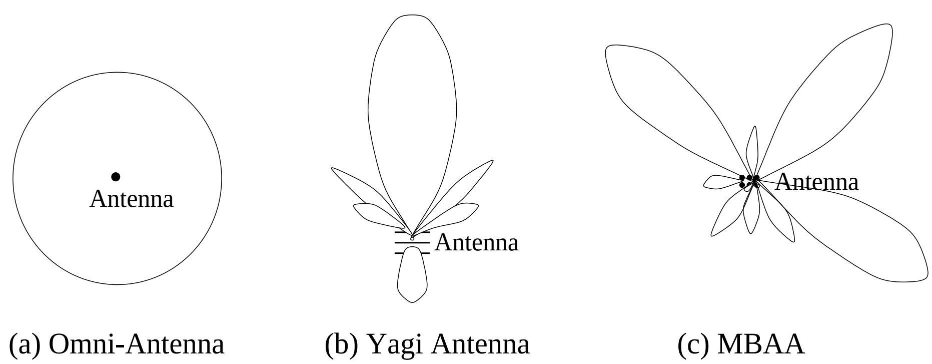


Figure 5.1: Antenna radiation patterns.

Figure 5.1 (a) and (b) illustrate the RF radiation patterns of the idealized omnidirectional antenna and the Yagi antenna for comparison purposes. Figure 5.1 (c) shows the radiation pattern of an MBAA antenna that is capable of dynamically forming three independent antenna beams in separate directions. The smaller parasitic lobes in Figure 5.1 (b) and (c) are called "side lobes" that can cause harmful interference to other receivers in the nearby vicinity of a radiating directional antenna. However, the side lobes can be avoided at the receivers by steering nulls toward the spurious side lobes, owing to the adaptability of the directional antenna beams. For simplicity, side lobes are omitted from discussions for the rest of the chapter.

We consider the use of multi-beam adaptive array (MBAA) antennas in ad hoc networks. When used in an ad hoc network, an MBAA antenna can successfully receive and transmit one or more overlapping packets at the same time by pointing its beams toward individual packet directions, while annulling all other unwanted directions. The number of beams that an MBAA antenna is capable of forming is denoted by K.

On the other hand, we assume that an MBAA antenna is also capable of broadcast by using the omnidirectional mode of the antenna at a lower frequency band or higher transmission power so that the coverage areas of both directional and omnidirectional transmissions are the same. Broadcasting capability is useful in mobile ad hoc networks for control information propagation and neighbor-direction findings. Using an electrically steerable switched-parasitic antenna array, Preston [64] presented three operational modes of directional antennas for finding (a) the coarse angular location of single source, (b) the precise angular location of single source and (c) the precise angular locations of multiple sources. Depending on the signal processing speed and the mode, the angular position of a radiating source can be decided within one or two hundred microseconds. We assume that an MBAA antenna system is capable of the second mode that detects the precise angular position of a single source for one-hop neighbor locating and tracking purposes.

A directional antenna works only in the half-duplex mode.

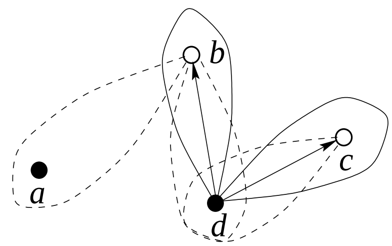


Figure 5.2: Communications using MBAA antennas.

Figure 5.2 illustrates two data communication sessions using MBAA antennas. The solid lines indicate the RF radiation beams, while the dotted lines indicate reception beams of the receivers. The arrows point to the directions of the data flows. Node d is transmitting two separate data packets to node b and c, respectively. In scheduled channel access protocols, node b may still orient its reception beam to node a, even though node a has no packet to transmit.

5.1.2 Network Topology

We assume that each node in the network is assigned a unique ID number, and equipped with an MBAA antenna. The topology of a packet radio network is represented by a directed graph $G = (V, E)$, where V is the set of nodes, and E is the set of directional links between nodes, $E \subseteq V \times V$. If a link (u, v) belongs to E, it can be activated when node u directs its transmission beam toward node v, and node v points its reception beam toward node u. Node u and v are called *one-hop neighbors*

Table 5.1: Additional notation in ROMA

K	The maximum number of beams formed by an MBAA antenna.
(u, v).`state`	The activation state of link (u, v).
ACT	Active state.
INACT	Inactive state.
i.`income`	The set of active incoming links to node i in reception mode.
i.`outgo`	The set of active outgoing links from node i in transmission mode.

to each other. Regarding link (u, v), node u is called the *head* of the link, while node v is the *tail*. A link (u, v) always has a companion link (v, u) in the opposite direction. The set of one-hop neighbors of a node u is denoted as N_u^1.

Every link of the network has a weight that reflects the data flow demand over the link, and is determined dynamically by the head of the link, which monitors traffic demands or receives bandwidth requests from the upper-layer applications. The weight of a link (u, v) is denoted by $w_{(u,v)}$. To prevent instability in the channel access schedules due to frequent link weight changes, the weight values are limited to the values in the set $\{0,\ 1,\ 2,\ 3\}$. A link with weight 0 can never be activated, whereas a link with weight 3 gets the most share of the channel as we will discuss in the specification of ROMA.

We use the notation in Table 5.1 in addition to the notation in Table 3.1

for the specification of ROMA. At any time slot t, the antenna at node i is either in the transmission mode (Tx) or the reception mode (Rx). The state of a link (u, v) is chosen from ACT or INACT to indicate the activation status. If (u, v).`state` is ACT, u may transmit a data packet u to v using through the main lobe of the directional antenna.

Each node $i \in V$ maintains angular profiles of its one-hop neighbors for antenna-beam orientation purposes. For simplicity, the nodes in the network are assumed to be placed on a flat plane. The horizon seen by a node is evenly divided into $360°/\frac{\beta}{2} = 720°/\beta$ segments, and every two continuous segments define one group. A group corresponds to the coverage of a directional beam from the node, and a segment determines the minimum angular separation of two neighbors for receiving non-interfering individual antenna beams. Consequently, $720°/\beta$ groups are identified. Each one-hop neighbor j of a node i belongs to two groups that overlap at j. The set of angular groups that a one-hop neighbor j of node i belongs to is denoted by A_i^j.

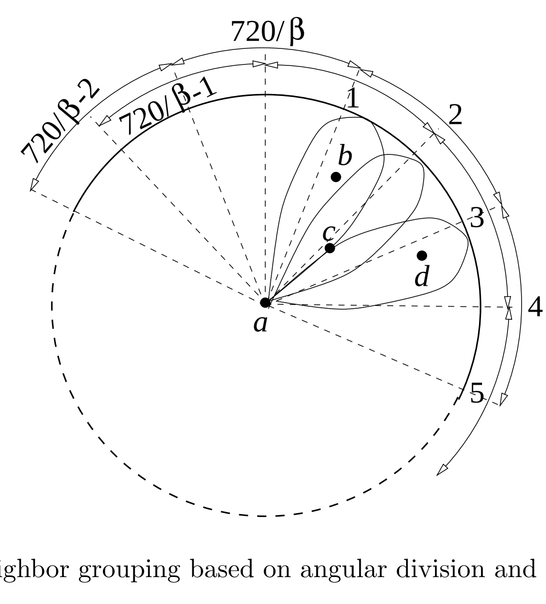


Figure 5.3: Neighbor grouping based on angular division and antenna patterns.

The inclusion of the angular group profiling is to avoid excessive communication overhead for updating neighbor locations between one-hop neighbors in ROMA. Angular group profiling also provides an easy tool in algorithm descriptions. For example, in Figure 5.3, the set of the angular groups for link (a, b) is $A_a^b = \{1,\ 2\}$, for link (a, c) is $A_a^c = \{2,\ 3\}$, and for link (a, d) is $A_a^d = \{3,\ 4\}$. The decision about whether two beams from node a can be activated for transmissions to b and c is to simply examine whether $A_a^b \cap A_a^c$ is an empty set.

Based on the above definitions, the attributes of a one-hop neighbor j of a node i can now be represented by the tuple: $(j,\ w_{(i,j)},\ w_{(j,i)},\ A_i^j)$. The attributes of a neighbor is used for contention resolution. Every node is required to promptly propagate its one-hop neighbor information to all of its one-hop neighbors whenever the attributes of a neighbor change, which is handled by the neighbor protocol described in Chapter 2.

Last but not least, we assume that time is synchronized on all mobile nodes to such a precision that the time difference between any pair of one-hop neighbors does not exceed the maximum signal propagation delay between the one-hop neighbors. Time synchronization can be achieved by a physical-layer protocol attaching the real-time clock information to data packets before transmissions, and aligning time slots to the latest starting point of a complete packet received [50].

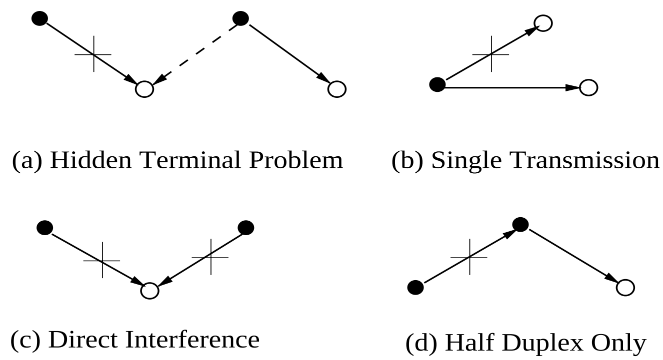


Figure 5.4: Contention types.

5.2 ROMA

As Figure 5.4 illustrates, a channel access protocol has to consider four types of contentions in multihop wireless networks: transmissions should not cause interference to other communication sessions (Figure 5.4 (a)); each transmission can convey only one packet (Figure 5.4 (b)); each reception accepts only one packet (Figure 5.4 (c)); and a node cannot transmit and receive at the same time (Figure 5.4 (d)).

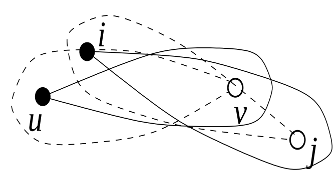


Figure 5.5: Hidden-terminal problem in directional antenna systems.

The hidden-terminal problem in networks with directional antennas is illustrated in Figure 5.5, in which link (i, j) and (u, v) are simultaneously activated. Interference happens at node v because both radiation lobes from node i and node u cover node v. When node v orients its reception lobe to node u, it accidently becomes sensitive to the signals from node i as well.

The other type of hidden-terminal interference comes from the side lobes of irrelevant communication sessions at the receivers. However, because the receivers can adaptively adjust their reception beams to nullify the sources of the side lobes, we do not consider the harmful effect from side lobes. Therefore, Figure 5.5 illustrates the only situation where the hidden-terminal problem happens, in which case node i is responsible for avoiding the problem, because both nodes u and v are in the one-hop neighborhood of node i, and node i has enough information to avoid the problem.

Nodes and links are assigned priorities based on their identifiers and the current time slot. When the current time slot is t, the priority of a node i is computed by

$$i.\texttt{prio} = \texttt{Hash}(i \oplus t) \oplus i \ , \tag{5.1}$$

where the sign $\oplus$ is designated to carry out the bit-wise concatenation operation on its operands, and has lower order than other operations. Function $\texttt{Hash}(x)$ is a fast pseudo-random number generator that produces an unsigned integer message digest of the input bit stream x. The identifier of node i is appended to the result to distinguish the priority from those of other nodes.

The priority of a link $(u, v) \in E$ is computed by

$$(u, v).\texttt{prio} = (i.\texttt{prio} \ \textbf{mod} \ 2) \oplus (\texttt{Hash}(u \oplus v \oplus t) \cdot w_{(u,v)}) \oplus u \oplus v \ , \tag{5.2}$$

which uses the same hashing function and distinguishing feature as that of the node-priority computation. The variable $w_{(u,v)}$ denotes the weight of link (u, v), and is discussed subsequently.

```
ROMA(i, t)
{
    /* Priority and Tx/Rx mode assignments. */
    for (k ∈ N_i^1 ∪ (⋃_{j∈N_i^1} N_j^1)) {
        k.prio = Hash(t ⊕ k);
        if (k.prio mod 2 ≡ 1)
            k.mode = Tx; /* Transmit mode. */
        else
            k.mode = Rx; /* Receive mode. */
    }

    /* Break unanimous Tx/Rx tie
        in one-hop neighborhood. */
    for (j ∈ N_i^1 ∪ {i}) {
        if ((∀k ∈ N_j^1 ∪ {j}, k.mode ≡ Tx and
             (∀k ∈ N_j^1, j.prio > k.prio))
              j.mode = Rx; /* All transmitters? */
        else if ((∀k ∈ N_j^1 ∪ {j}, k.mode ≡ Rx and
             (∀k ∈ N_j^1, j.prio > k.prio))
              j.mode = Tx; /* All receivers? */
    }

    /* Compute active incoming links for
        one-hop neighbors in Rx mode. */
    for (j ∈ N_i^1 ∪ {i}, and j.mode ≡ Rx) {
        /* Initialization. */
        for (k ∈ N_j^1) {
            (k, j).state = ACT;
            (k, j).prio = (k.prio mod 2)⊕
                  (Hash(k ⊕ j ⊕ t)· w_(k,j))⊕k ⊕ j;
        }

        /* Hidden-terminal avoidance. */
        for (k ∈ N_j^1 and (k, j).state ≡ ACT) {
            if (∃m ∈ N_j^1, A_j^k ∩ A_j^m ≠ ∅ and
                (m, j).prio > (k, j).prio)
                (k, j).state = INACT;
```

Figure 5.6: ROMA Specification (Part 1).

```
    /* Select up to K active incoming links. */
    [ Sort (k, j) according to (k, j).prio
      in descending order, where k \in N_j^1 and
      (k, j).state \equiv ACT ];

    j.income =
      {k | k belongs to top K of the sorted list};
  }

  /* Collect active outgoing links. */
  if (i.mode \equiv Tx) {
    i.outgo = \emptyset;

    /* Active outgoing links are the active
      incoming links at one-hop neighbors. */
    for (j \in N_i^1)
      if (j.mode \equiv Rx and i \in j.income)
        i.outgo = i.outgo \cup {j};

    for (j \in i.outgo)
      if ( /* Figure 5.4 (b). */
        \exists k \in i.outgo and A_i^k \cap A_i^j \neq \emptyset and
        (Packet to k is earlier than to j))
        i.outgo = i.outgo - {j};
      else if ( /* Figure 5.5. */
        \exists v \in N_i^1, v.mode \equiv Rx and A_i^j \cap A_i^v \neq \emptyset and
        (\exists u \in N_i^1 \cap N_v^1, u.mode \equiv Tx and A_v^i \cap A_v^u \neq \emptyset))
        i.outgo = i.outgo - {j};
  }

  if (i.mode \equiv Rx)
    [ Tune antenna beams to members
      in i.income for reception ];
  else if (i.outgo \neq \emptyset) {
    [ Select up to K members from i.outgo
      with the earliest packets, and tune
      antenna beams for transmission ];
} /* End of ROMA. */
```

Figure 5.7: ROMA Specification (Part 2).

ROMA is a link-activation receiver-oriented multiple access protocol that exploits the multi-beam forming capability of MBAA antennas. Given up-to-date information about the two-hop neighborhood of a node and link bandwidth allocations, ROMA decides whether a node i is a receiver or a transmitter, and which corresponding links can be activated for reception or transmission during time slot t.

In essence, node i separates its neighbors within two hops including i itself into receivers and transmitters randomly. Then node i chooses up to K active incoming links with the highest priorities for receivers in i's one-hop neighborhood using only the corresponding one-hop neighbor information. If node i is a transmitter, node i is allowed to choose up to K of its active outgoing links for transmissions. The value K indicates the number of antenna beams provided by an MBAA antenna. Figures 5.6 and 5.7 specify ROMA using C-style pseudo code.

Using the current time slot number t, ROMA determines the priority, and subsequently the mode of each node k in node i's two-hop neighborhood according to whether the node priority k.`prio` is odd or even (lines 1-7). If k.`prio` is odd, node k is a transmitter for the current time slot; otherwise, node k is in reception mode. As a result, nodes are randomly separated into two classes. It is possible that a node and all its one-hop neighbors are put into the same class, such that the node can neither transmit or receive. Lines 8-15 break the stalemate by converting the mode of the node into the opposite state if the node has the highest priority among its one-hop neighborhood.

Although it is not necessary when node i is in reception mode, lines 16-28 compute up to K active incoming links for every node in reception mode in the one-hop neighborhood of node i for uniformity. Lines 16-21 initialize the state and priority of each incoming link to the receivers according to the link identifiers, current time slot and whether the head priorities are odd. For each node in reception mode, lines 23-25 deactivate some of its incoming links according their priorities if the links cause direct interference, as shown in Figure 5.4 (c). Afterward, up to K incoming links with the highest priorities at the receiver are chosen for activation (lines 26-27).

If node i is in transmission mode, it needs to collect the active outgoing links to its one-hop neighbors (lines 31-33) according to the results of lines 16-28. Furthermore, node i needs to avoid activating multiple links in the same angular group (lines 35-38), and avoid causing any hidden-terminal problem to its one-hop neighbors (lines 39-42).

If node i is a receiver, node i may orient its antenna beams toward the one-hop neighbors in the incoming link set (lines 44-45). Otherwise, node i may select up to K outgoing links for transmissions using MBAA antennas according to traffic scheduling criteria (lines 46-47).

Overall, ROMA has to decide the active incoming links of each node in reception mode before the actual link activations at the transmitters.

Figure 5.8 illustrates the operation of ROMA in a sample network with MBAA antennas capable of forming up to three antenna beams. Nodes denoted by solid circles indicate the nodes are in Tx mode (transmitter), and nodes denoted by

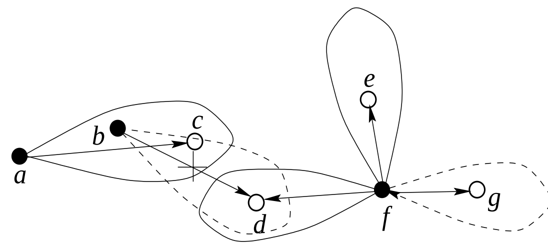


Figure 5.8: Example of ROMA operation.

empty circles indicate that they are in Rx mode (receiver). Arrows leading into each receiver are the incoming links chosen by the receiver for activation. Lobes depicted by solid lines indicates the traffic needs from the transmitters. However, because node b detects hidden-terminal contention at node c incurred from node a and node b itself, link (b, d) is not activated (dashed lobe). On the other hand, node g is ready to receive from node f, but node f has no traffic for node g, and link (f, g) is not activated, either (dashed lobe).

The computation of link priorities is carried out as follows:

- The oddity of a node is prepended to the link priority (term (k.`prio` **mod** 2) in Eq. (5.2), and **ROMA** line 19). This operation differentiates the transmitters converted from reception-mode nodes (**ROMA** lines 12-14) against transmitters computed by regular means (**ROMA** lines 3-4), such that incoming links in the latter case always have higher priorities than those in the former case. The converted transmitters may join the active incoming links of the receiver (**ROMA** lines 26-27), only when the transmitters derived from regular means cannot fulfill the reception capacity of the receive MBAA antennas.

- The priority of a link (u, v) is proportional to its weight, $w^t_{(u,v)}$ (Eq. (5.2), and

ROMA line 20). Even though the weight of a link ranges over only four integer values, the bandwidth allocations change dramatically according to the different weight values. For instance, given that three links (x, i), (y, i) and (z, i) have weight $w_{(x,i)} = 1$, $w_{(y,i)} = 2$ and $w_{(z,i)} = 3$, and only one incoming link of node i can be activated at a time, the bandwidth allocations to the three links are $\frac{1}{3} \cdot \frac{1}{3} = 11\%$, $\frac{1}{3} \cdot \frac{1}{3} + \frac{1}{3} \cdot \frac{1}{2} = 28\%$ and $\frac{1}{3} \cdot \frac{1}{3} + \frac{1}{3} \cdot \frac{1}{2} + \frac{1}{3} = 61\%$ of the total incoming bandwidth at node i, respectively, because of the differences in the link priority ranges. When carefully chosen, the limited number of weight values in ROMA can satisfy wide ranges of the bandwidth demands. However, the choice of link weights depends on the traffic requirements in the application layer, and is outside the scope of this dissertation.

5.2.1 Mobility Handling

In the neighbor protocol, we use the omnidirectional mode of the directional antennas to propagate neighbor updates as well as to find out the current angular locations of one-hop neighbors in mobile networks. However, the antenna gain of directional antennas operating in directional mode can be significantly different from that in the omnidirectional mode, making the coverage of the two transmissions significantly different. The difference depends on the frequency and the output power of the antennas in the two modes. We assume that the directional antennas work at different frequencies and suitable output power levels in the two operation modes, such that the coverage of the two transmissions are approximately the same.

Because of the direction sensitivity in ROMA, the neighbor protocol needs to promptly update the one-hop neighbor locations, so that the next round of channel access scheduling is free of errors. Therefore, the random access section should be allocated as frequently as possible for better responsiveness of the neighbor protocol. For example, if the random access section is allocated every second, the neighbor protocol needs $100ms$ for neighbor information update purpose, using the result in Eq. (2.13). Because L in Eq. (2.13) is an upper bound of the latency in delivering a message to all one-hop neighbors at once, the real latency in delivering the neighbor updates can be much lower if we consider that the message can also arrive asynchronously at one-hop neighbors during the process of retransmissions.

5.3 Performance

5.3.1 Static Multiple Access Scheduling

Channel assignment problems in the time, frequency and code domains have traditionally been treated as graph coloring problems. The basic characteristic of these channel access schemes is that the schedule is static as long as network topology remains unchanged. Inherently, topology information needs to be collected and frequent schedule broadcasts have to be carried out in mobile networks.

We compare ROMA with the best-known static schedule approximation algorithms that are summarized in a unified framework by Ramanathan [65]. Assuming the global topology of the network, Ramanathan [65] provided a unified algorithm for coloring the nodes or links of the graph in polynomial time.

The constraints on nodes or edges of the graph are represented by eleven atomic relations between nodes or edges. A constraint set characterizes a channel assignment problem on the graph using various technologies, such as TDMA, FDMA or CDMA. However, it did not specify the modeling of constraints in spatial division multiple access (SDMA) scheme. It happens that the only change necessary in UxDMA for SDMA is the procedure for choosing the first available least color. For comparison purposes, we modify the algorithm for searching the first available color in SDMA scheme such that the color selection process considers angular profiles of one-hop neighbors as well as the maximum number of incident links in the same color.

The number of colors used by UxDMA determines the time frame during which every link is able to access the channel once. When computing the colorings on the graphs in UxDMA, an optimal ordering, PMNF (Progressive Minimum Neighbors First) heuristic, has been applied in each computation so that the colorings "perform quite close to optimum" [65].

An MBAA antenna may only activate K incoming or K outgoing links simultaneously. Therefore, the constraint set in UxDMA in networks with directional antennas is:

$$\{E^0_{tr}, E^0_{rr}, E^0_{tt}, E^1_{tr}\},$$

where E^0_{tr} denotes the self-interference case in Figure 5.4, and E^1_{tr} represents the hidden-terminal case. E^0_{rr} and E^0_{tt} constrain multiple simultaneous transmission or reception sessions from a node. However, because of the multi-beam capability of the antenna systems, the constraints E^0_{rr} and E^0_{tt} are allowed as long as the number of instances at an antenna does not exceed K.

5.3.2 Simulation Assumptions

We study the performance of ROMA by running simulations in two scenarios: fully connected networks and multihop networks. Fully connected networks exhibit homogeneous contention situations for each link, while links in multihop networks encounter different levels of contentions because of the variations in node density. The fully connected networks are generated by setting the size of the square plane to 100×100 square meters, and tuning the transmission range of directional antennas to 100 meters, so that every node is reachable from all other nodes. The contention level in fully connected networks is affected only by the number of nodes. We study the performance differences when the network has 5 and 20 nodes. In multihop networks, contention levels for each link are determined not only by the number of nodes in the network, but also by the antenna coverage. We generate the multihop networks by randomly placing 100 nodes within a square plane of 1000×1000 square meters, and set the antenna transmission ranges to 200 and 400 meters, respectively. Because both ROMA and UxDMA can support channel access scheduling with multiple antenna beam activations, directional antennas with one, two and four beams are simulated, respectively, as well. The performance is measured in terms of the delay, throughput and packet drop-rates of the protocols in each simulation case.

UxDMA is simulated in each scenario with the corresponding constraint parameters as well. Because UxDMA is a static scheduling algorithm, the coloring of links in each scenario is carried out at the beginning of each simulation.

We model the packets arrivals at each node as a Poisson process (packet inter-arrival intervals are exponentially distributed with parameter λ), and packets are served in first-in-first-out (FIFO) order. All nodes have the same packet arrival rate λ. Because every node has equal probability of being activated in UxDMA, the data packets are evenly dispatched onto each outgoing link. In ROMA, each link has different probability of activation depending on the number of contenders of each link, thus the traffic is proportionally distributed to outgoing links according to the activation probability of that link. The simulations are guided by the following parameters and assumptions:

- The beamwidth of directional antennas is 30°.
- Because UxDMA is not capable of dynamic bandwidth allocations, ROMA has the weight of each link fixed to one.
- Antenna beams always have the same transmission range in each simulation scenario. We do not consider power management for communicating with one-hop neighbors at different distances.
- Signal propagation in the channel follows the free-space model and the effective transmission range is determined by the power level of the antenna alone.
- The bandwidth of the radio channel is 2 Mbps. In all simulations, the bandwidths of all links are assigned 1 for simplicity.

- A time unit in the simulation equals to one time slot. A time slot last for 8 milliseconds, including guard time, which is long enough to transmit a 2KB packet.

- Only static networks are considered in the simulations, so that the two-hop neighbor information or the entire topology is known beforehand in the corresponding protocols. The networks are generated by randomly placing a number of nodes onto a square plane. To simulate an infinite plane that has constant node placement density, the opposite sides of the square are seamed together, which visually turns the square area into a torus.

- At each node, the number of the memory buffers holding packets for each neighbor is 20. Generally, dropping packets has very minor influence on the system throughput because there are most likely other fresher data packets waiting when the older packets are dropped, and channel access chances are not likely to be wasted. However, we assume an infinite buffer size for simulations using single beam-forming antennas.

- The duration of the simulation is 800 seconds (equal to 100000 time slots), which is long enough to compute the metrics of interests.

In addition to the delay and throughput examinations of these protocols, we also study the packet drop rates due to memory overflow in the corresponding simulations.

5.3.3 Analysis of Results

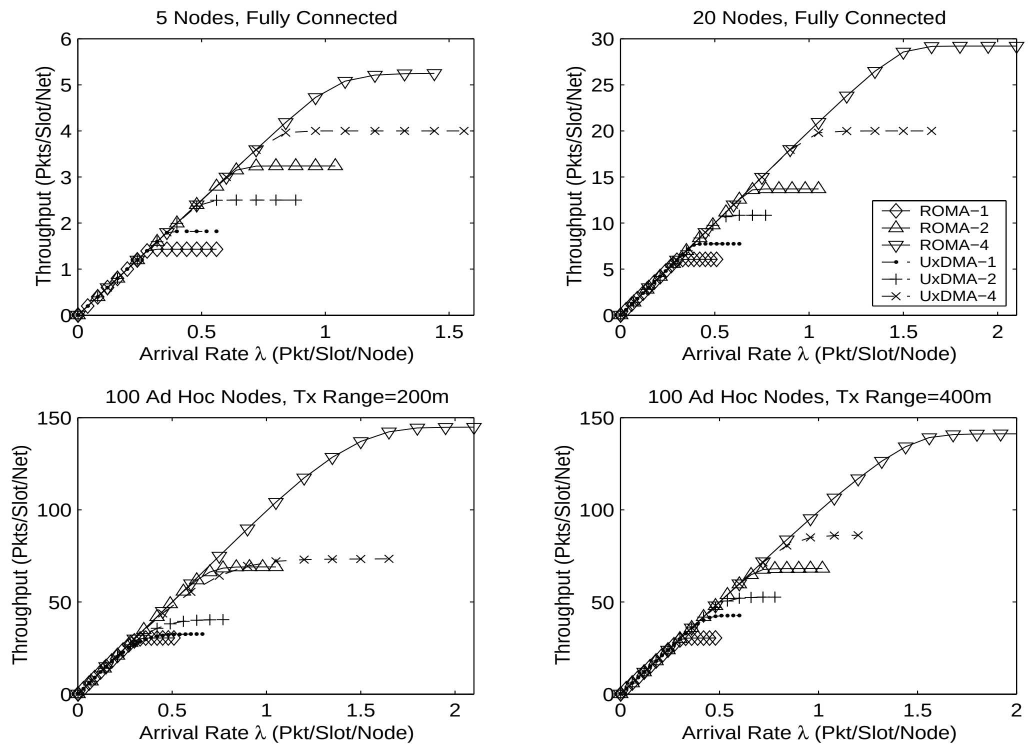


Figure 5.9: Average packet throughput in networks with MBAA antennas capable of different numbers of beams.

Figures 5.9 and 5.10 show the throughput and delay attributes of ROMA and UxDMA in fully-connected and multihop networks when the number of active antenna beams is one, two and four. The appended numbers in the legends represent the number of beams that each antenna can form.

Figure 5.9 shows that networks with MBAA antennas capable of forming four beams have higher throughput than those of forming two beams, and networks with two-beam antennas have more capacity that those with a single beam-forming capability. UxDMA displays better performance characteristics than ROMA when

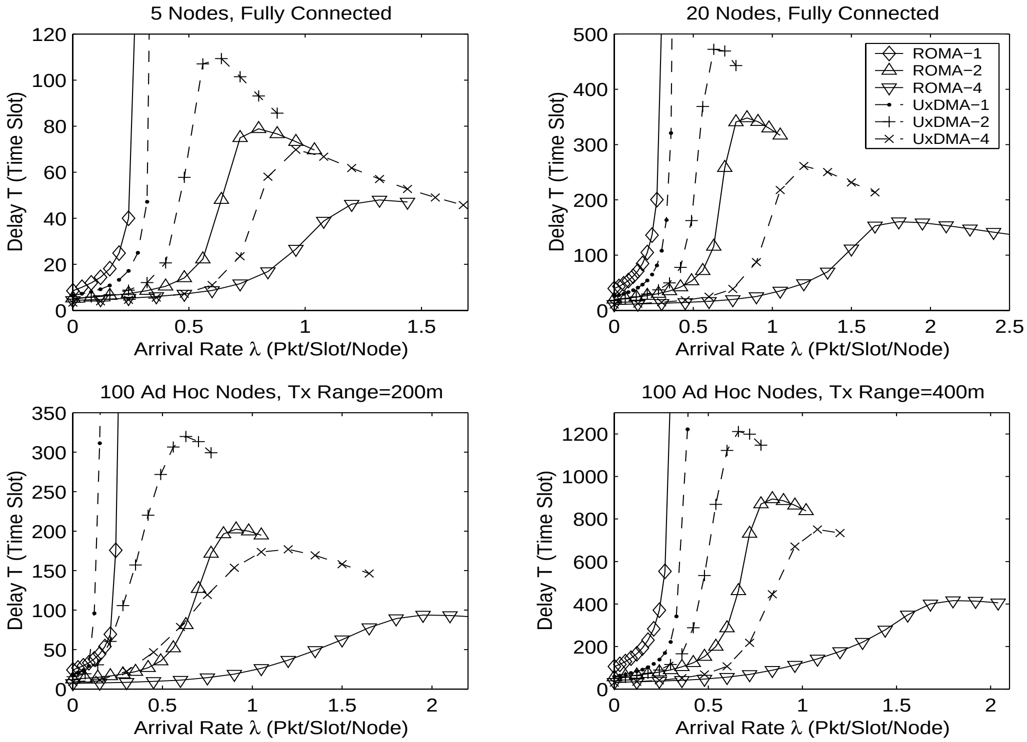


Figure 5.10: Average packet delays in networks with MBAA antennas capable of different numbers of beams.

the number of active antenna beams is one in the fully connected networks and the multihop network constructed with 400m transmission range. This is due to the fact that the contention situations are more homogeneous, and UxDMA can take advantage of global topology information. However, UxDMA under-performs ROMA in multihop network in the lower-left plots of Figure 5.9 and 5.10. This is because ROMA is more adaptive to the local topology of each node. When the network is randomly generated with a lower transmission range (200 meters), contentions are more heterogeneous in different parts of the network due to node density variations on the plane, and link activations are more frequent at network regions with less node or link density than those with higher node or link density. In contrast, UxDMA assigns a link only one activation chance per time frame, and has to apply the same time frame throughput the network.

Figure 5.10 demonstrates that the more beams that an MBAA antenna is capable of forming, the lower the packet delays. Except for the case of MBAA antennas with a single beam, ROMA has better performance than UxDMA in other scenarios. Figure 5.10 also shows the effects of memory resource limitations on the delay attributes of ROMA and UxDMA when the antenna system has two and four beam-forming capabilities. When the packet arrival rates become higher than what the network can serve, the packet delay begins to drop at the verge of the network capacity, then drops because only fresher packets are kept in the buffers as packets arrive faster. In contrast, when the MBAA antennas support only a single beam with infinite buffer, the delay increases to infinity when the packet arrival rate exceeds the network capacity.

In general, when the MBAA antennas of the nodes in the networks have two or four beams, ROMA always outperforms UxDMA in network throughput and packet delay. The average packet delays are lower and the network throughput is higher in ROMA than in UxDMA in each simulation scenario.

We also examined the drop rates of different drop policies under various traffic loads in Figure 5.11. ROMA provides lower packet drop rates than UxDMA under the same traffic loads due to its high throughput capability.

ROMA demonstrated superior adaptiveness over the link scheduling algorithm (UxDMA) in all scenario simulations of multihop networks, because two-hop

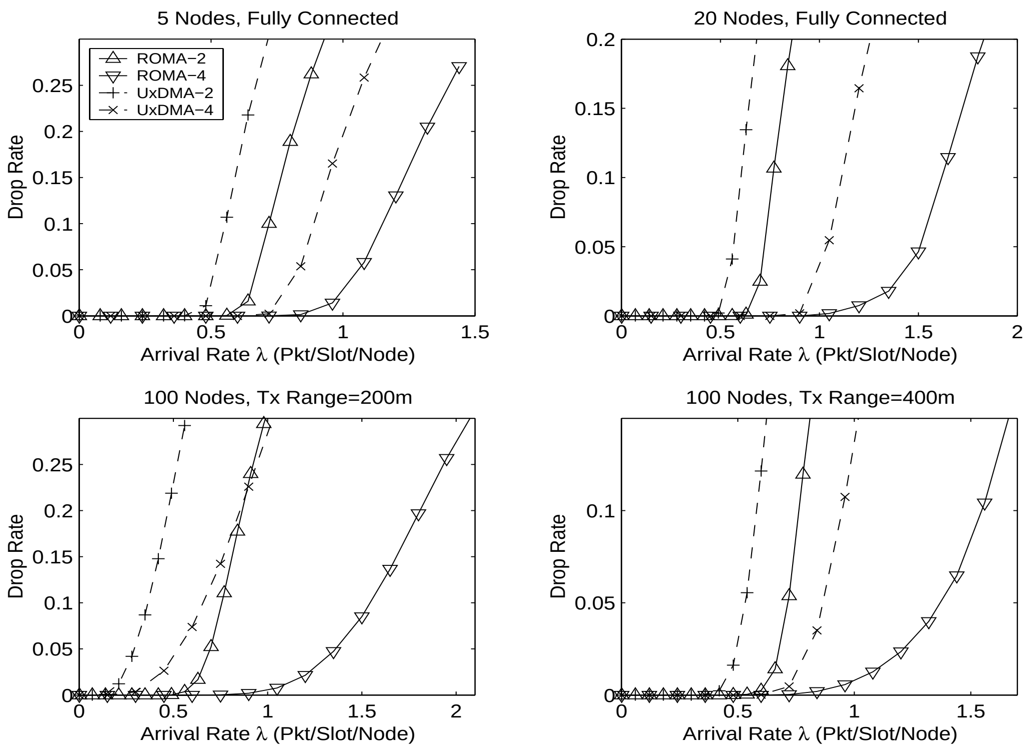


Figure 5.11: Packet drop rates in networks with MBAA antennas capable of forming two and four beams.

neighbor information is necessary and sufficient to insure collision-freedom in multihop packet radio networks. Applying the capacity at some parts of the network topology with the worst contentions to the other less contended parts results in inefficient channel utilization in UxDMA. In addition, ROMA tries to evenly separate networks nodes into transmitters and receivers, so that link activations are maximized in each time slot. UxDMA lacks a mechanism to balance transmissions and receptions.

However, ROMA does have some disadvantages, in that the intervals between successive activations of a single link is non-deterministic, and is governed by a geometric distribution, which gives uncertainty about the delays. This is an inherent property of channel access schemes when random functions are involved, as seen in any other on-demand channel access protocol. Only global and relatively static scheduling can guarantee bounds on packet delays.

5.4 Conclusion

We have introduced ROMA, a very efficient distributed channel access scheduling protocol for ad hoc networks with directional antennas that are capable of forming multiple beams to carry out several simultaneous data communication sessions. ROMA shows superior performance over the best-known polynomial time approximation algorithm (UxDMA) for scheduling in ad hoc networks with multi-beam adaptive array (MBAA) antennas in terms of the network throughput and packet delay. The ability of ROMA to achieve collision-freedom for channel access using only two-hop topology information is more efficient than in UxDMA with respect to the control overhead incurred by the two approaches.

Chapter 6

Stable Energy-Aware Topology Management in Ad Hoc Networks

This chapter introduces new algorithms for computing the minimal dominating set (MDS) and the connected dominating set (CDS) of ad hoc networks using NCR, which we call Topology Management by Priority Ordering (**TMPO**). **TMPO** requires only two-hop neighbor information to assign dynamic priorities to nodes according to the node identifiers and their "willingness" values. The willingness of a node is a function of the mobility and battery life of the node. Using the node priorities, **TMPO** elects the clusterheads comprising the MDS, and consequently determines the nodes needed to connect these clusterheads to obtain the CDS in the corresponding network topology.

The rest of the chapter is organized as follows. Section 6.1 describes the network topology assumptions. Section 6.2 specifies the minimal dominating set election algorithm based on node priorities. Section 6.3 extends the minimal dominating set (MDS) to a connected dominating set (CDS) in ad hoc networks. The correctness of MDS and CDS election algorithms is proved in Section 6.4. Section 6.5 analyzes the size of the MDS elected by TMPO. Section 6.6 compares **TMPO** with other CDS computation algorithms using simulations. Section 6.7 summarizes the chapter and key contribution of the work.

6.1 Network Assumptions

The topology of a wireless ad hoc network is represented by an undirected graph $G = (V, E)$, where V is the set of network nodes, and $E \subseteq V \times V$ is the set of links between nodes. The existence of a link $(u, v) \in E$ also means $(v, u) \in E$, and that nodes u and v are within the transmission range of each other, so that they can exchange packets via the common channel, in which case u and v are called *one-hop neighbors* to each other. The set of one-hop neighbors of a node i is denoted as N_i^1. Two nodes that are outside the transmission range of each other but share at least one common one-hop neighbor are called *two-hop neighbor* of each other.

We assume that each node in the network has an omnidirectional antenna, and is assigned a unique identifier (ID). Time is synchronized at each node to the time slot boundaries, and time t is defined by the corresponding time slot number, starting from a consensus temporal point in the past. In addition, we assume that there is a reliable neighbor protocol that propagates one-hop neighbor update information and updates two-hop neighbor information at each node. The mechanisms for acquiring and synchronizing two-hop neighbor information were described in Chapter 2.

6.2 Minimal Dominating Set

The dominating set problem in graph theory consists of finding a subset of nodes with the following property: each node is either in the dominating set, or is adjacent to a node in the dominating set. The members of the dominating set are often called *clusterhead*s, whereas the other nodes without special functionalities are called *host*s.

6.2.1 Clusterhead Election Criteria

The minimum dominating set problem is known to be NP-hard [3] [36] even when the complete network topology is available. In ad hoc networks, the difficulty of acquiring complete network topology makes it impossible to compute the "minimum" dominating sets. Instead, a *minimal* dominating set (MDS) is usually pursued based on various heuristics that can guarantee a local minimum election of the dominators in polynomial steps.

We propose a distributed algorithm for determining the candidacy of a node in the MDS by a single step using the available neighbor information within two hops. The neighbor information includes the identifiers of the neighbors and their willingness to become clusterheads. Based on the information, a priority is computed and assigned to each node for the current time. Then a node makes a local decision about whether it becomes a clusterhead according to one of the following criteria:

1. The node has the highest priority in its one-hop neighborhood.

2. The node has the highest priority in the one-hop neighborhood of one of its one-hop neighbors.

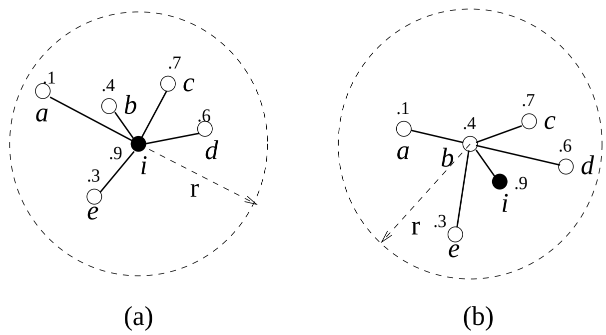


Figure 6.1: Two cases that enable node i becoming a clusterhead.

For convenience in the discussion, we represent node priorities using floating point numbers throughout this chapter. The principles in the algorithms can be easily converted to integer operations. Figure 6.1 illustrates the two criteria that make node i a clusterhead. The number next to each node is the sample priority at a particular moment. In Figure 6.1 (a), node i has the highest priority among its one-hop neighbors. In Figure 6.1 (b), node i has the highest priority among node b's one-hop neighbors. We discuss the computation of node priorities in the following sections.

6.2.2 Willingness to Become A Clusterhead

Clusterheads in the MDS are used to build the connected dominating set and provide the backbone of an ad hoc network for data forwarding services. Therefore, the energy of the clusterheads is consumed more rapidly than that of ordinary hosts, which can stand by or doze off to save energy. In addition, because draining the energy of certain nodes may disable some part of the network, we need to avoid electing low-energy nodes as the clusterheads to save their energy. We also need to prevent nodes in high moving speed from becoming clusterheads, so that the MDS experiences fewer structural changes.

It is preferable that clusterheads stay relatively static and robust, and have a high amount of energy. A composite willingness parameter is introduced to each node in the network and is directly related with the probability of the node being elected as a clusterhead as we see in the following section.

Denote the willingness value of node i by w_i, the speed of node i by $s_i \in (0, \infty)$ in terms of meters per second, and the remaining energy on node i as $e_i \in [0, 1]$. The willingness value w_i is derived using following formula:

$$w_i = e_i^{\lfloor \log_2(1+s_i) \rfloor} \tag{6.1}$$

where the *floor* operation is given to the speed logarithm for larger granularity of the willingness values. The logarithmic operation on the speed value makes the willingness value more sensitive to the speed variations at low speed level. When the speed is high, the variations in speed do not affect the willingness value much because the willingness value is already low at that speed. On the other hand, because the energy level is represented by a fractional number, the high-speed and the low-energy properties of a node give the node a very low willingness value to become a clusterhead.

At node i, the energy level e_i and speed s_i are both local parameters, only used to derive the willingness value. The willingness value is an attribute of node i and is propagated to the one- and two-hop neighbors of node i.

6.2.3 Clusterhead Election Algorithm

The priority of a node is a pseudo random number generated from the message digest derived from the node identifier and the current time, and multiplied by the willingness value of the node. The priorities of nodes in the network are periodically recomputed so that the nodes load-balance clusterhead roles in a timely manner.

The recomputation period, denoted by T in terms of time slots, lasts from seconds to minutes depending on the deployments of ad hoc networks, and is the same for all nodes in the network. However, to avoid the synchronization problem during clusterhead role changes, each node has a time-slot offset when the priorities are recomputed. The time-slot offset with regard to node i is denoted as $i.\texttt{off}$, and is computed in Eq. (6.2).

$$i.\texttt{off} = \lfloor \texttt{Hash}(i) \cdot T \rfloor \tag{6.2}$$

where the function $\texttt{Hash}(x)$ is a pseudo-random floating-point number generator that produces a uniformly distributed random number over range $(0, 1]$ based on the input bit-stream x. The trailing fraction of the result is truncated to get an integer offset.

Suppose the current time slot is t, t can be represented in the following way:

$$t = kT + r$$

The point when a node needs to recompute its priority is when r becomes equal to the time-offset of the node. At the moment $r \equiv i.\texttt{off}$, the priority of node i, denoted by $i.\texttt{prio}$, is recomputed according to the following formula:

$$i.\texttt{prio} = \texttt{Hash}(k \oplus i) \cdot w_i \oplus i \tag{6.3}$$

where the function `Hash` is defined in Eq. (6.2), and the sign "$\oplus$" is designated to carry out bit-concatenation operation on its operands, and has lower order than other operations. The last concatenation with i in the final result is to differentiate the priorities for different nodes. The priority of a node remains the same for the duration of T.

The algorithms for determining the clusterhead status of node i are described in the following. Each node has a `type` field, indicating the clusterhead or other states of the node. Nodes are initially all categorized as *host*s.

/ Initialize */*
Init(i, t)
{
    $i.\texttt{workfor} = \emptyset$; */* Explained later. */*
    $w_i = e_i^{\lfloor \log_2(1+s_i) \rfloor}$;
    $i.\texttt{type} = \text{Host}$;

    / Recompute priorities. */*
    for ($j \in N_i^1 \cup (\bigcup_{k \in N_i^1} N_k^1)$) {
        if ($t \equiv 0$)
            $j.\texttt{prio} = \texttt{Hash}(j) \cdot j.\texttt{will} \oplus j$;
        else if ($t - j.\texttt{off} \bmod T \equiv 0$)
            $j.\texttt{prio} = \texttt{Hash}(\frac{t-j.\texttt{off}}{T} \oplus j) \cdot j.\texttt{will} \oplus j$;
    }
} */* End of* **Init**. **/*

Figure 6.2: TMPO Function for initialization.

Function **Init** in Figure 6.2 initializes the data structures at node i. Data members that are self-explanatory are added on line. In particular, node i has a data structure `workfor` that will be used in the connected dominating set election algorithm to store the clusterheads that node i may connect to provide a connected backbone. The set i.`workfor` is initialized to be empty (**Init** line 1). The willingness value of node i is computed according to Eq. (6.1) and the nodal type is initialized to `Host` (**Init** lines 2-3). Node i also computes the priority for each two-hop neighbor according to the recomputation period of the neighbor (**Init** lines 4-9).

```
isClusterhead(i, t)
{
  for (j ∈ N_i^1 ∪ {i}) {
     j.ch = j;

     for (k ∈ N_j^1)
        if (k.prio > j.ch.prio)
           j.ch = k;

     if (j.ch ≡ i)
        i.type = Clusterhead;
  }
} /* End of isClusterhead. */
```

Figure 6.3: TMPO Function for determining clusterhead.

After invoking **Init**, the node priorities are computed and stored in the neighbor data structures, and function **isClusterhead** in Figure 6.3 decides the clusterhead of a node i, which is indicated by the field i.`ch` in the neighbor data structure.

If node i becomes a clusterhead after computing the MDS using function **isClusterhead**, its clusterhead-type attribute needs to be propagated to its two-hop neighbors for further computations.

6.3 Connected Dominating Set

Topology management requires a minimum and sufficient number of network connections to work as the communication backbone of the network while reducing network maintenance and control overhead. The minimum connected dominating set (MCDS) problem provides a best model for this purpose. The MCDS problem consists of obtaining a minimum subset of nodes in the original graph such that the nodes compose a dominating set of the graph, and the induced subgraph has the same number of connected components as the original graph. However, MCDS is a well-known NP-complete problem in graph theory. Instead, we have to approach the MCDS problem with a sub-optimum solution that involves only local computations. We call such sub-optimum MCDS the connected dominating set (CDS) in this chapter.

6.3.1 CDS Election

Because the maximum distance from a clusterhead in the minimal dominating set (MDS) to the closest clusterhead is three, which we will prove in Theorem 2, we can derive the CDS by adding some nodes to the MDS such that clusterheads within two or three hops are connected, using only two-hop neighbor information. Two other types of nodes, called *doorways* and *gateways*, are added upon MDS to derive the CDS.

The CDS of a network topology based on the MDS is constructed in two steps. In the first step, if two clusterheads in the MDS are separated by three hops and there are no other clusterheads between them, a node on the shortest path between the two clusterheads becomes a *doorway*, and is added to the CDS. Therefore, the addition of a doorway brings the connected components where the two clusterheads reside one hop closer. In the second step, if two clusterheads or one clusterhead and one doorway node are only two hops away and there are no other clusterheads between them, one of the nodes between them becomes a *gateway* to connect clusterhead to clusterhead or doorway to clusterhead.

The decision of a node becoming a doorway or gateway depends on the priorities of neighbors within two hops. Figures 6.4 and 6.5 specify the functions for determining the eligibility of a node i to become a doorway or a gateway in order to compose the CDS. Function **TMPO** in Figure 6.6 is the entry point that encloses the MDS and CDS election function calls.

Function **isDoorway** in Figure 6.4 determines whether node i can become a doorway for other clusterheads at time slot t. To decide whether node i becomes a doorway for clusterhead n and j, node i needs to assert that

1. Clusterheads n and j are not two hops away (**isDoorway** lines 3-7 illustrated by case (a) in Figure 6.7).
2. There is no other clusterhead m on the shortest path between clusterhead n and j (**isDoorway** lines 8-11 illustrated by case (b) and (c) in Figure 6.7).
3. There is no other node m with higher priority than node i on the three-hop path between clusterhead n and j (**isDoorway** lines 12-16 illustrated by case (d) in Figure 6.7).

If the three assertions are satisfied, node i becomes a doorway (**isDoorway**

isDoorway(i, t)
{
if (i.`type` $\equiv$ Clusterhead)
  return;

for ($j \in N_i^1$ **and** j.`type` $\equiv$ Clusterhead) {
  for ($k \in N_i^1$ **and** $k \neq j$
    and k.`type` $\neq$ Clusterhead) {
    for ($n \in N_k^1$, n.`type`$\equiv$Clusterhead **and** $n \notin N_i^1 \cup N_j^1$) {
      if ($\exists m \in N_i^1$, $\{j,\ n\} \subseteq N_m^1$) */* Case (a) in Figure 6.7. */*
        continue n;

      for ($m \in N_i^1$) {
        / Case (b) or (c) in Figure 6.7. */*
        if ($n \in N_m^1$ **and** ((m.`type` $\equiv$ Clusterhead) **or**
          ($\exists p \in N_i^1 \cap N_m^1$, p.`type` $\equiv$ Clusterhead)))
            continue n;
      } */* m */*

      for ($m \in N_i^1 \cap N_n^1$) {
        / Case (d) in Figure 6.7. */*
        if ((m.`prio` $>$ i.`prio`) **or**
          ($\exists p \in N_j^1 \cap N_m^1$, p.`prio` $>$ i.`prio`))
          continue n;
      } */* m */*

      i.`type` = Doorway;
      i.`workfor` = i.`workfor` $\cup \{j, n\}$;
    } */* n */*
  } */* k */*
} */* j */*
} */* End of* **isDoorway**. **/*

Figure 6.4: TMPO Function for determining doorway.

```
isGateway(i, t)
{
 1  if (i.type ∈ {Clusterhead, Doorway})
 2     return;

 3  for (j ∈ N_i^1
 4     and j.type ∈ {Clusterhead, Doorway}) {
 5     for (k ∈ N_i^1 and k ≠ j and k ∉ N_j^1 and
 6           k.type ∈ {Clusterhead, Doorway} and
 7           (k.type ≢ Doorway or
 8              j.type ≢ Doorway) {
 9        if (∃n ∈ N_j^1 ∪ N_k^1, n ≠ i and
              (/* Case (a) in Figure 6.8. */
10            n.type ∈ {Clusterhead, Doorway} or
              /* Case (b) in Figure 6.8. */
11            n.prio > i.prio))
12           continue k;
13        else {
14           i.type = Gateway;
15           i.workfor = i.workfor ∪{j, k};
16        }
17     } /* k */
18  } /* j */
} /* End of isGateway. */
```

Figure 6.5: TMPO Function for determining gateway.

line 17). The attribute i.`workfor` is the set of clusterheads that make node i become a doorway (**isDoorway** line 18).

Function **isGateway** in Figure 6.5 determines whether node i becomes a gateway to connect two clusterheads or one clusterhead and another doorway, k and j (**isGateway** lines 3-8). According to Figure 6.8, if there is another clusterhead or doorway between node k and j (**isGateway** line 10), or there is another node with higher priority than node i between node k and j (**isGateway** line 11), node i cannot become a gateway. Otherwise, node i becomes a gateway, and the attribute i.`workfor` is the set of nodes that make i become a gateway (**isGateway** lines 13-16).

```
TMPO(i, t)
{
   i.oldType = i.type;
   Init(i, t);
   isClusterhead(i, t);
   isDoorway(i, t);
   isGateway(i, t);

   /* i's status changes? */
   if (i.type ≠ i.oldType)
      Propagate i.type;
} /* End of TMPO. */
```

Figure 6.6: TMPO Overall Function.

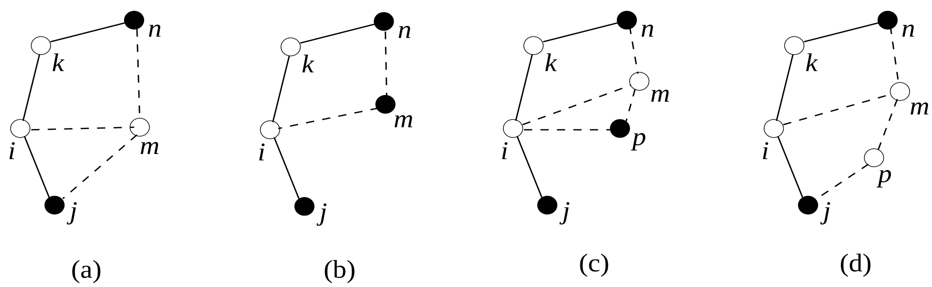


Figure 6.7: Four cases that may disable i becoming a doorway

Function **TMPO** is called after every neighbor information update. After calling **TMPO**, if node i changes its role between clusterhead, doorway, gateway and host, then node i needs to propagate its new status to its neighbors. Note that the status changes from/to doorway and gateway are propagated only to one-hop neighbors, while the status changes from/to clusterheads are required to propagate within two hops, so as to inform other nodes when constructing the CDS. It is the responsibility of a neighbor protocol to propagate the changes in node types to the one- and two-hop neighbors in time.

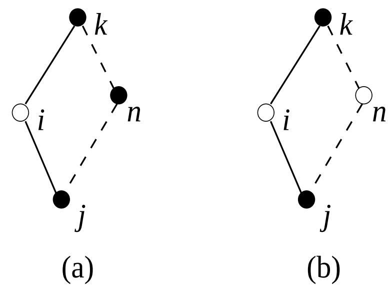


Figure 6.8: Two cases that disable i becoming a gateway.

6.3.2 CDS Connections

Based on the elected clusterheads, doorways and gateways, the backbone topology is therefore constructed between these specialized nodes. Links of the backbone topology are selected according to the following rules:

R1: All links between clusterheads are kept in the backbone topology.

R2: The one-hop links from doorways and gateways to the nodes in their respective sets `workfor` are kept in the backbone topology. Some nodes in the `workfor` set may be outside of the one-hop neighborhood for doorways, and there are no links from the doorways to these nodes.

The links of the original topology between gateways or between doorways are not kept in the CDS. Doorways are alway attached to a clusterhead on one end and a gateway on the other end. Gateways are alway attached with clusterheads or doorways.

For communication purposes, clusterheads are responsible for receiving or delivering data packets to hosts under their dominance, while gateways and doorways forward data packet between clusterheads for which they work, as indicated by the data member `workfor` in function **isDoorway** and **isGateway**.

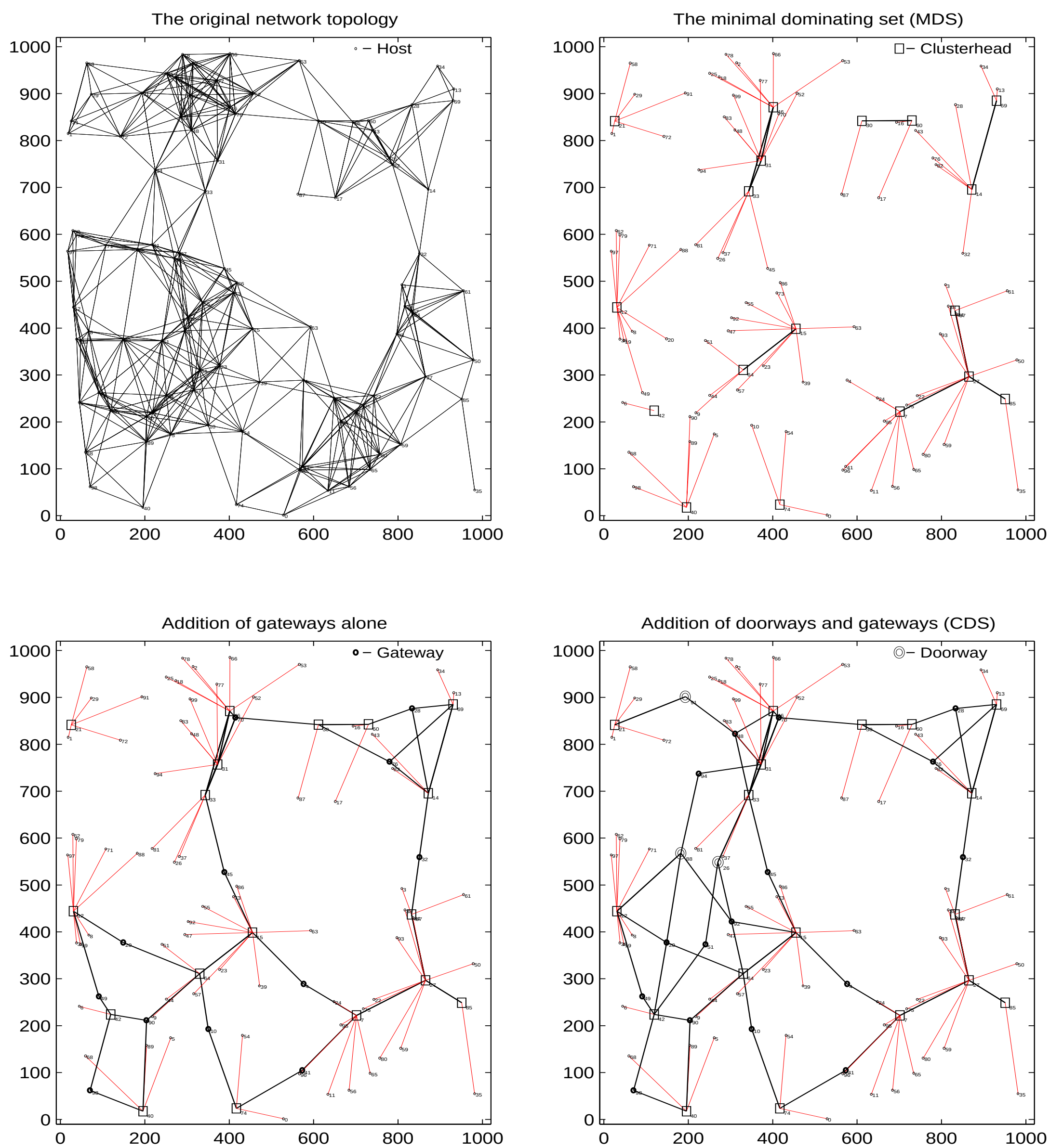


Figure 6.9: A network topology control example.

Figure 6.9 illustrates the backbone topology construction process using **TMPO** in four steps on a network graph, which is generated by randomly placing 100 nodes over a 1000×1000 square meter area. The radio transmission range is 200 meter, the same for all nodes. In Figure 6.9 (a), every node is a host, and network connections are dense at some parts of the network, which can be big problem to the routing control overheads if the routing protocols pro-actively exchange routing information between all neighboring nodes. In Figure 6.9 (b), clusterheads are elected, but disconnected from each other. Hosts are attached to their corresponding clusterheads in their one-hop neighborhood. In Figure 6.9 (c), gateways are elected. Without adding doorways for extending the coverage of clusterheads, we can see that adding gateways alone cannot guarantee the connectivity of the network. Once the doorways are added after the clusterheads election in Figure 6.9 (d), gateways are again inserted, and the CDS is formed over the original network topology using the CDS construction rules.

The stability of the backbone topology is an important issue in **TMPO**, and will be achieved by means of a few mechanisms when applying **TMPO**. First, the period of willingness adjustment is long to allow enough time to adjust to clusterhead changes. Second, the willingness value is directly related with the speed of node movement. Fast moving nodes get fewer chances to become a clusterhead than lowly moving or static nodes, thus decreasing the possibility of topology changes due to mobility. Third, the willingness value is also related with the remaining energy of the node, so that the clusterhead role is sustained longer and fewer topology changes happen. Forth, except for clusterheads, doorways and gateways are not put in the routing tables built over the CDS, but only provide links between clusterheads. When doorways or gateways fail, there can be other hosts taking in the role and sustaining the topology without any routing updates. The transient period between clusterhead connection re-establishment is equal to the delay of the neighbor protocol for propagating one-hop neighbor updates. Lastly, nodes change their priorities aperiodically, thus avoiding synchronization problem from clusterhead changes and routing updates.

6.3.3 Application of CDS

The CDS obtained from **TMPO** reduces the topology information at each node with just enough links to maintain network connectivity.

Efficient Routing

Routing information requires only enough active nodes in the network for connectivity and data forwarding purposes among hosts and routers. The number of active nodes can be dramatically smaller than the total number of nodes in the network. An extreme case is a fully connected network, in which only one active node is needed to provide the routing functionality. The CDS of a multihop wireless network provides the backbone of the network for this purpose, where clusterheads, doorways and gateways in the CDS work as routers in the network, forwarding data packets, and the hosts are simplified wireless routers that only originate or receive data packets.

Another application for the CDS problem is to carry out reliable broadcasts in ad hoc networks. In broadcast operations, each message from a source node needs to reach every other network node. There is no need for path maintenance, and traditional broadcast may flood and involve every node to replay the messages, unnecessarily incurring high overhead for broadcast operations. Utilizing CDS in the network, network hosts are attached to their clusterheads. Because of the network coverage of the MDS, broadcast messages can be delivered to the end-points reliably by delegating the broadcast responsibility to the corresponding clusterheads. The propagation of broadcast messages from one clusterhead to others is achieved by the intermediate doorways and gateway using reliable mechanisms by acknowledgments.

Power Conservation

Ad hoc networks are characterized by multi-hop wireless connectivity, frequently topology changes and the need for efficient dynamic routing protocols. Power consumption not only comes from data flows generated or forwarded by each node, but also from routing information maintenance. In flat network topology management, nodes maintain routing information exchanges with every neighbor, and consume large amount of energy.

To balance power consumption as well as to maintain network connectivity, **TMPO** elects backbone routers (CDS) to serve the network routing functionalities based on their power levels and mobility. Only clusterheads, doorways and gateways stay awake all the time. Hosts that are not serving data forwarding can be put in sleeping mode so as to conserve energy, and to prolong network lifetime by later switching the clusterhead role to other hosts in **TMPO**. Routers in the CDS buffer information for sleeping hosts until they wake up and receive the packets. Hosts attached to core routers wake up at times when core routers recompute their priority to receive data. If a backbone router becomes host router, and the buffered data are not delivered to the destination host, the backbone router keeps holding the data and deliver or delegate them later.

6.4 Correctness

Theorem 1 *The set of clusterheads elected by the algorithm is a dominating set.*

Proof: By the definition of a dominating set, a node is either a dominator itself, or is a one-hop neighbor of a dominator. Because a node either has the highest priority among its one-hop neighbors such that it becomes a dominator itself, or has a neighbor with the highest priority among the one-hop neighbors of the node, which elects the neighbor as a dominator, the network always has a dominating set elected after function **isClusterhead** is called at each node. □

Theorem 2 *In a dominating set, the maximum distance to another closest clusterhead from any clusterhead is three.*

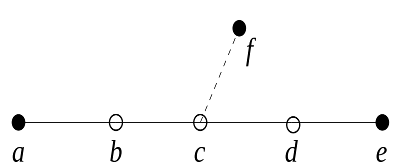


Figure 6.10: The maximum distance between the closest clusterheads.

Proof: We prove the theorem by contradiction. Assume that the maximum distance from a clusterhead a to the closest clusterhead e is four, as illustrated in Figure 6.10, according to Theorem 1, node c must have been covered by a clusterhead f, which is one hop closer to a than e, thus contradicting the assumption that e is the closest clusterhead to a. □

Theorem 3 *Two clusterheads that are within three hops from each other are connected by a path in the CDS.*

Proof: There are three cases to consider for the proof according to the number of hops between the two clusterheads.

1. The two clusterheads are one hop away. They are directly connected according to rule R1 in the CDS.

2. The two clusterheads are two hops away. One of the intermediate nodes between the clusterheads either is a clusterhead itself, or become a gateway according to **isGateway** algorithm. Because the link between gateway and clusterhead is kept in the backbone topology by rule R2, the two clusterheads are connected in the CDS.

3. The two clusterheads are three hops away. If there is any clusterhead on the shortest paths between the two clusterheads, then the connectivity problem is converted to the previous two cases. Otherwise, one of the nodes on the shortest paths has to become a doorway according to function **isDoorway**. Because the doorway is treated similarly as clusterhead when electing gateways (function **isGateway**), one of the nodes between the newly elected doorway and the other clusterhead becomes a gateway. Because the link between the doorway and the

clusterhead for which it works is kept in the backbone topology as well as the links from the elected gateway to the doorway and the clusterhead (rule R2), the path between the two clusterheads is preserved in the backbone topology. That is, the aforementioned two clusterheads are still connected via clusterheads, doorways or gateways in the CDS. □

Theorem 4 *After* **TMPO** *terminates, the CDS has the same number of connected components as the original graph.*

Proof: We prove that any pair of clusterheads that are connected in the original graph is still connected via a path in the CDS after **TMPO** terminates.

Suppose that the two clusterheads are v_0 and v_n, and the path between them is $p = v_0 \cdot v_1 \cdot v_2 \cdots v_{n-1} \cdot v_n$ in the original graph. For the endpoints of any link $v_i \cdot v_{i+1}$ on the path p, where $i = 0, 1, \cdots, n-1$, there exist one or two clusterheads that cover node v_i and v_{i+1}. For the case of one clusterhead, the clusterheads of v_i and v_{i+1} are trivially connected. For the case of two clusterheads, the distance between the clusterheads is less than three. From Theorem 3, it follows that the two clusterheads are still connected in the backbone topology. Therefore, there is a path between v_0 and v_n that is composed of clusterheads of the nodes on the path p and other clusterheads, doorways or gateways that connect them. That is, v_0 and v_n are still connected in the CDS. □

6.5 Performance Analysis of the MDS Algorithm

6.5.1 Clusterhead Probability

Guha and Khuller [39] and Jia *et al.* [41] evaluated the performance of algorithms for constructing dominating sets based on the *performance ratio*, which is the approximate ratio of the cost of a solution from using an algorithm to the optimal one. In contrast, we evaluate the performance of an MDS election algorithm by the percentage of nodes being elected as clusterheads.

Despite the fact that we have actually provided a *node-weighted* MDS election algorithm based on the willingness parameter, the performance of the specified algorithm can be evaluated regardless of such weights because of the randomness and the consideration of the network topology as a whole. Therefore, we consider the case in which all nodes have the same willingness to become a clusterhead, that is, $w_i = 1, \forall i \in V$. Furthermore, we analyze the probability of a node being elected as a clusterhead with the following simplified assumptions: all nodes have the same effective transmission range r to communicate with each other, and the network is created by uniformly placing an infinite number of nodes on an infinite 2-dimensional plane with average node density ρ. The number of nodes with an area of size S is a random variable following a Poisson distribution as given in Eq. (6.4).

$$p(k, S) = \frac{(\rho S)^k}{k!} e^{-\rho S} \ . \tag{6.4}$$

Because node priorities are evenly distributed over $(0, 1]$ according to Eq.

(6.3), nodes have equal opportunities to become clusterheads using the clusterhead election algorithm. That is, the probability of a node winning over k other contenders is $\frac{1}{k+1}$.

For convenience, the variable $T(N)$ and $U(N)$ are introduced to denote two probabilities when the number of contenders k follows Poisson distribution with mean N. $T(N)$ denotes the probability of a node winning among its contenders. Because the number of contenders follows Poisson distribution with mean N, and that all nodes have equal chances of winning, the probability $T(N)$ is the average over all possible numbers of the contenders:

$$T(N) = \sum_{k=1}^{\infty} \frac{1}{k+1} \frac{N^k}{k!} e^{-N} = \frac{e^N - 1 - N}{N e^N} \; .$$

Note that k starts from 1 in the expression for $T(N)$, because a node with no contenders does not win at all. $U(N)$ is the probability that a node has at least one contender, which is simply $1 - e^{-N}$.

In addition, N_1 is introduced to denote the average number of one-hop neighbors of a node, which is $N_1 = \rho \pi r^2$ according to the assumptions we have made.

As mentioned before, a node i becomes a clusterhead if either of the following two conditions holds:

1. Node i has the highest priority among its one-hop neighbors;

2. Node i does not have the highest priority in its one-hop neighbors, but has the highest priority among the one-hop neighbors of one of i's own one-hop neighbors.

For the first condition, the probability is:

$$p_1 = \sum_{k=0}^{\infty} \frac{N_1^k}{k!} e^{-N_1} \cdot \frac{1}{k+1} = \frac{U(N_1)}{N_1} \ .$$

For the second condition, there are many situations can render node i as the clusterhead, depending on the way in which one-hop neighbors are placed around node i. However, due to the high complexity of various geometric relations and correlations between multiple one-hop neighbors, we only consider the lower bound of the probability of i becoming a clusterhead by focusing on a single one-hop neighbor j that makes node i a clusterhead.

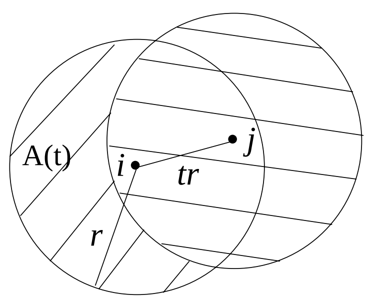


Figure 6.11: Clusterhead election.

Hence, node i needs at least one neighbor j among whose one-hop neighbors node i has the highest priority, and node i and j are separated by distance tr as shown in Figure 6.11. For node i to become a clusterhead in node j's one-hop neighborhood alone, there should be at least another node with higher priority than node i in the side lobe area, in which the number of nodes is denoted by $A(t)$. We have

$$A(t) = 2\rho r^2 \left[\frac{\pi}{2} - a(t) \right] \ ,$$

where $a(t) = \arccos\frac{t}{2} - \frac{t}{2}\sqrt{1 - \left(\frac{t}{2}\right)^2}$. Therefore, the probability of node i losing to the nodes in the side lobe is

$$p_2 = \sum_{k=1}^{\infty} \frac{A(t)^k}{k!} e^{-A(t)} \frac{k}{k+1} = U(A(t)) - T(A(t)) \ .$$

In addition, node i should have the highest priority among node j's one-hop neighborhood, of which the probability is:

$$p_3 = \sum_{k=0}^{\infty} \frac{N_1^k}{k!} e^{-N_1} \frac{1}{k+2} = \frac{N_1 - 1}{N_1} \cdot T(N_1) + e^{-N_1} \ .$$

Since the probability density function of parameter t is $p(t) = 2t$, the probability that node i becomes a clusterhead can be obtained by multiplying the above two probabilities and integrating over the range $t \in (0, 1]$:

$$p_4 = \int_0^1 p_2 \cdot p_3 \cdot 2tdt = \left(\frac{N_1 - 1}{N_1} \cdot T(N_1) + e^{-N_1} \right) \cdot \int_0^1 [U(A(t)) - T(A(t))] 2tdt \ .$$

As the two conditions are mutually exclusive, the probability of node i becoming a clusterhead is thus:

$$\begin{aligned} p_{ch} &= p_1 + (1 - p_1) \cdot p_4 \\ &= \frac{U(N_1)}{N_1} + \left(1 - \frac{U(N_1)}{N_1}\right) \left(\frac{N_1 - 1}{N_1} \cdot T(N_1) + e^{-N_1} \right) \cdot \int_0^1 [U(A(t)) - T(A(t))] 2tdt \ . \end{aligned}$$

Nodes are homogeneous in the randomly generated network with regard to their priority generations and one-hop neighbor information. Therefore, the probability of becoming a clusterhead is also the same for all nodes. Given a segment with N nodes from an infinitely large network with uniform node density, the expected size of the MDS in the segment is:

$$|MDS_N| = N \cdot p_{ch} \ . \tag{6.5}$$

To validate the analysis in Eq. (6.5) with the performance of the MDS election function **isClusterhead**, we carried out a number of simulations on a randomly generated network by placing 100 mobile nodes onto a 1000×1000 square meter plane. The opposite sides of the square are seamed together so as to emulate the infinite plane. All nodes have the same transmission range, which increases from 1 to 400 meter in separate simulations so as to evaluate the performance of the algorithms in different node densities. Each node moves in a random direction at a random speeds ranging from 0 to 50 meter/second in all simulations.

A near-optimum MDS election is also carried out for comparison purposes in the same network topology by computing the MDS with the aforementioned **Max Degree** algorithm, and then eliminating redundant clusterheads in the MDS.

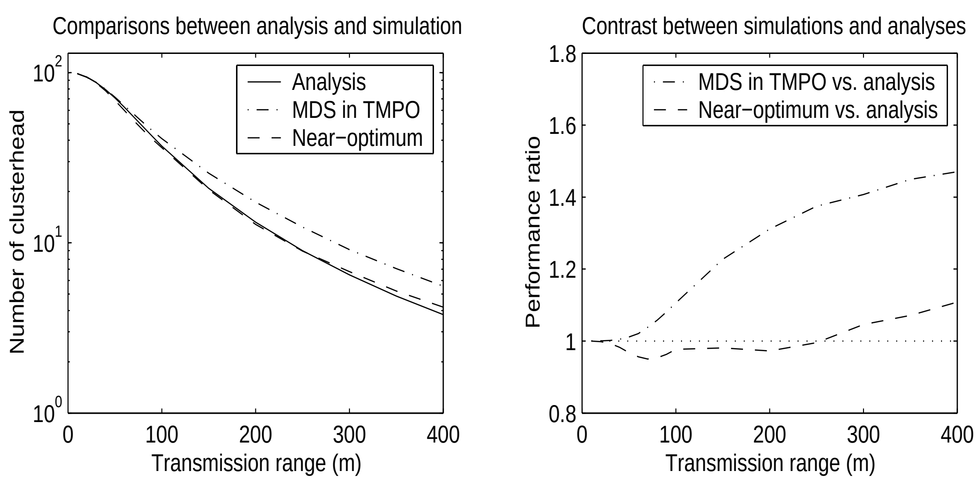


Figure 6.12: Comparison between theoretical analysis and simulations.

Because we ran the simulation in small time steps, and recomputed the MDS at each step, we achieve the average performances of both algorithms after running the simulations for a certain period. The theoretical curve and the results from the simulation are plotted in Figure 6.12.

The size of the MDS drops quickly as the node transmission range increases in the left diagram of Figure 6.12, while the performance ratios of both algorithms over the analyses digress from one gradually in the right diagram. Even though the analyses approximate the simulations closely in the left diagram, the sizes of the MDSes in **TMPO** offset the theoretical analysis as far as 48% as shown in the right diagram. This is because a node is more likely to be elected as a clusterhead when it considers more one-hop neighbors than simply considering only one neighbor, which was adopted in our analysis.

6.5.2 Effect of Willingness Value in Clusterhead Election

We analyze the effect of the willingness value in the probability of a node becoming a clusterhead.

We denote the priority of node n_k by p_k and its willingness value by w_k in the following discussion. According to Eq. (6.3), the priority p_k is a random variable uniformly distributed over the range $(0, w_k]$. Therefore, the probability of priority p_k being less than x is $P\{p_k < x\} = \frac{x}{w_k}$.

Suppose that we have a node with other $n-1$ contenders in the clusterhead election, where each is assigned a willingness value over the range $(0, 1]$. To derive the probability of a node being elected as a clusterhead in the group, we sort nodes by their willingness values in decreasing order, and denote the sorted nodes as n_1, n_2, $\cdots$, n_n, accordingly. The node of our interest is located at the k-th position, n_k. As a result, the willingness value range is divided into n ranges where the priority values of these nodes may reside: $(w_{n+1}, w_n]$, $(w_n, w_{n-1}]$, $\cdots$, $(w_2, w_1]$. The constant $w_{n+1} = 0$ is added to normalize the expressions.

Therefore, the probability of node n_k being elected as clusterhead with other $n-1$ contenders is:

$$P\{n_k \text{ becoming clusterhead}\} = \sum_{i=k}^{n} \frac{w_i^i - w_{i+1}^i}{i \cdot \prod_{j=1}^{i} w_j} .$$

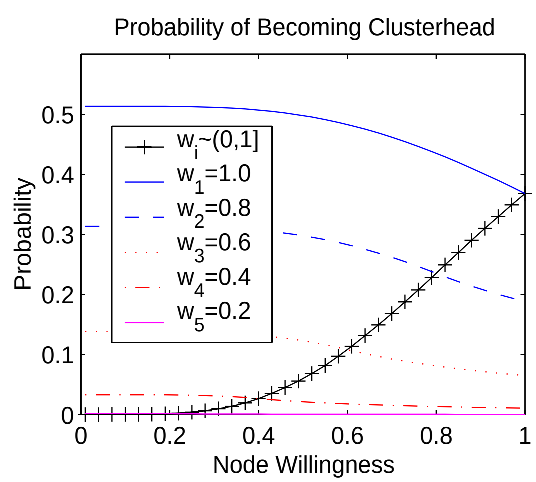


Figure 6.13: Probability becoming a clusterhead when node i increases w_i from 0 to 1, and other nodes remain constant.

Figure 6.13 shows the variation of the probability of node i being elected as the clusterhead when it has five other nodes contending for the clusterhead role. The other five nodes have fixed willingness values as indicated in the figure, while node i's willingness increases from 0.01 to 1. As the probability of node i being elected as clusterhead increases in response to the increments in its willingness, those of the other nodes decrease. When the willingness value becomes larger, the probability increases faster. Node n_5's probability of becoming clusterhead is almost negligible in the figure.

6.6 Simulations

We compare the performance of **TMPO** with the optimum topology management algorithm and four other topology management algorithms based on different heuristics using simulations. For comparability between these algorithms, some MDS stability optimizations and clusterhead negotiation procedures presented in the original papers are omitted. The algorithms differ from one another only in the clusterhead election process, and use the same procedure to connect clusterheads using doorways and gateways to form the CDS, which does not need negotiation packets and can actually improve the performance of the original algorithms based on other heuristics.

- **OPTIMUM**: In this near-optimum approach that uses global topology information, the MDS is constructed by selecting nodes with the highest degree one by one until all nodes outside the MDS are covered by the MDS. Individual nodes with low degree in the MDS are inspected and eliminated from the MDS if the node and its dominated nodes are covered by other clusterheads in the MDS.

- **Lowest ID** [3] [6] [32] [38] [54]: In this approach, the node identifier is used to elect MDS members. A node is elected into the MDS if it has the lowest identifier in the one-hop neighborhood of itself or one of its one-hop neighbors.

- **Max Degree** [39] [41] [72]: In this approach, a node is elected into the MDS if it has the highest degree in the one-hop neighborhood of itself or one of its one-hop neighbors.

- **MOBIC** [18]: In this approach, each node computes a mobility metric based on the received-signal strength variations from its one-hop neighbors. A node becomes a clusterhead if it has the lowest mobility metric in the one-hop neighborhood of itself or one of its one-hop neighbors.

- **Load Balance** [4]: This approach is similar to **Lowest ID** except that it is based on a virtual identifier (VID) assigned to each node. The VID of a node increases every time slot if the node is a host, or remains constant if the node is a clusterhead. Each clusterhead runs a budget that decreases every time slot. A clusterhead resets its VID to 0 and returns to host status when the budget runs out, thus providing load balancing between network nodes.

The simulations are carried out in ad hoc networks, generated over a 1000×1000 square meter area with 100 nodes moving in random directions at random speeds selected from 0 to 5 meters/second in low mobility scenarios, or from 0 to 50 meters/second in high mobility scenarios. In each type of the mobility scenarios, the radio transmission range is set at different values, chosen from 100 to 500 meters, so as to demonstrate the effects of the one-hop neighborhood density in the clusterhead elections. In **TMPO**, the node priority recomputation period is 1 minute.

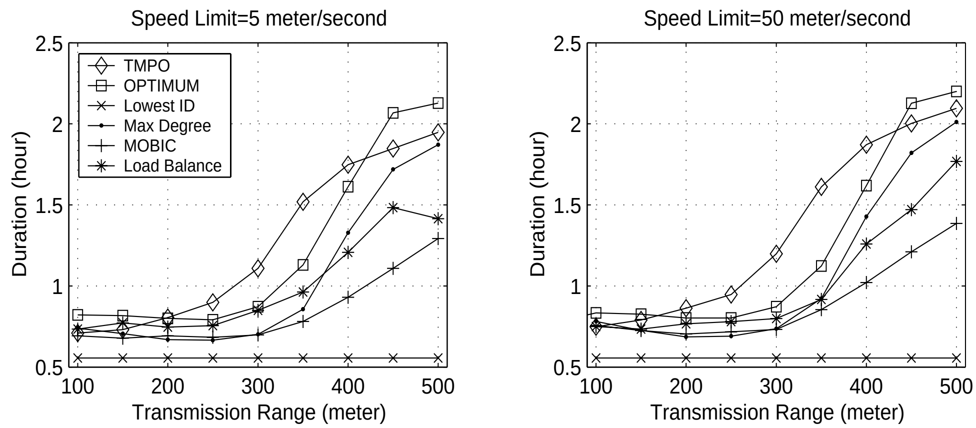
Different types of nodes consume energy at different rates. We ignore the power consumed due to local computations, but assume that the power consumption rate is only dependent on the type of the node. A host consumes 0.6% of the total energy per minute in these algorithms, a clusterhead consumes 3%, and a doorway or a gateway consumes 2.4%. Every node starts with a power level of 1 at the beginning of each simulation.

The algorithms are compared using the following metrics, and illustrated respectively in Figure 6.14-6.16:

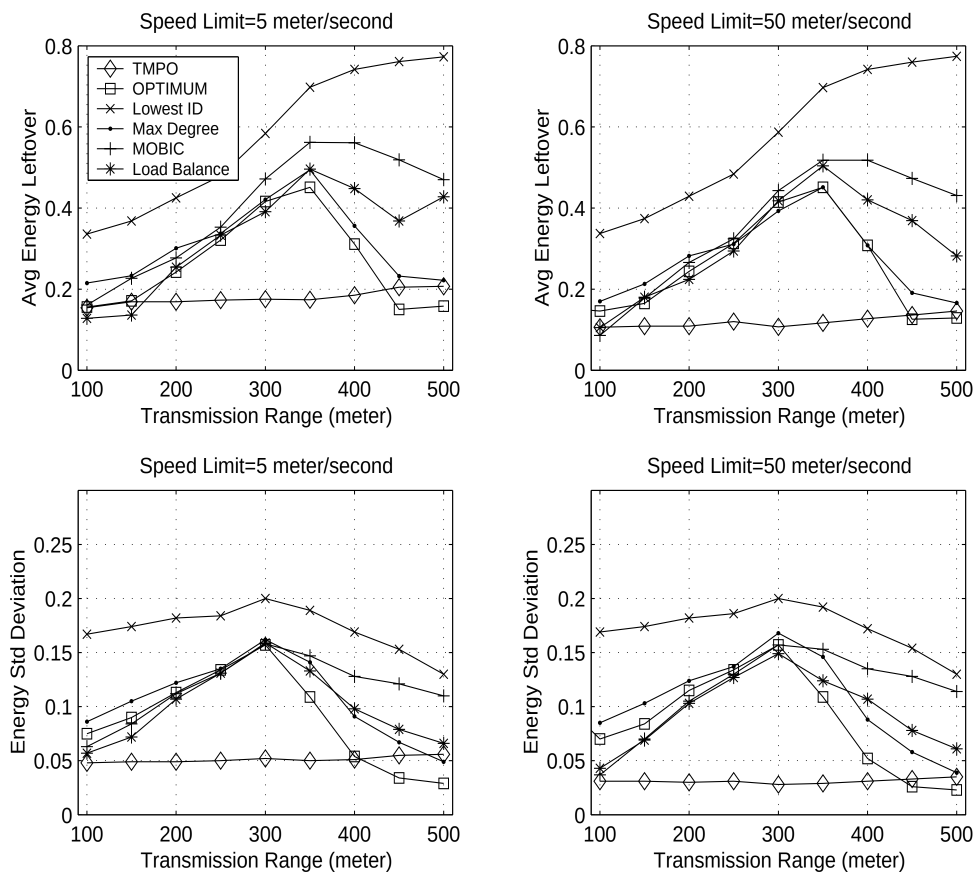
Simulation duration (Figure 6.14 (a)): The simulations stop once any node runs out of energy in the network. This metric measures the load balancing capability of the heuristics for prolonging the system lifetime by rotating the clusterhead roles between network nodes. **MOBIC** and **Lowest ID** perform the worst, because the clusterheads are mostly fixed over certain nodes throughout the simulations. In **Lowest ID**, the fact that the node with the lowest identifier is always in the MDS terminates the simulations in fixed time. **TMPO** is one of the best heuristics.

The mean and the standard deviation of the energy left per node when the simulation is over (Figure 6.14 (b)): This also indicates the load balancing capability of the heuristics. After each simulation, **TMPO** leaves the network nodes with the least energy and the lowest standard deviation because of its energy-awareness of when selecting the MDS, while **Lowest ID** performs the worst because it always has nodes that are always or never elected to the MDS. The curves peak at transmission range 300- or 350-meter because the two-hop neighborhood of each node begins to overlap in the opposite directions on the plain, which renders less clusterheads and more opportunities to rotate clusterhead roles for some heuristics.

Average number of clusterheads (Figure 6.15 (a)): which is measured each time slot when clusterhead recomputation happens. As we see, all heuristics perform almost the same.

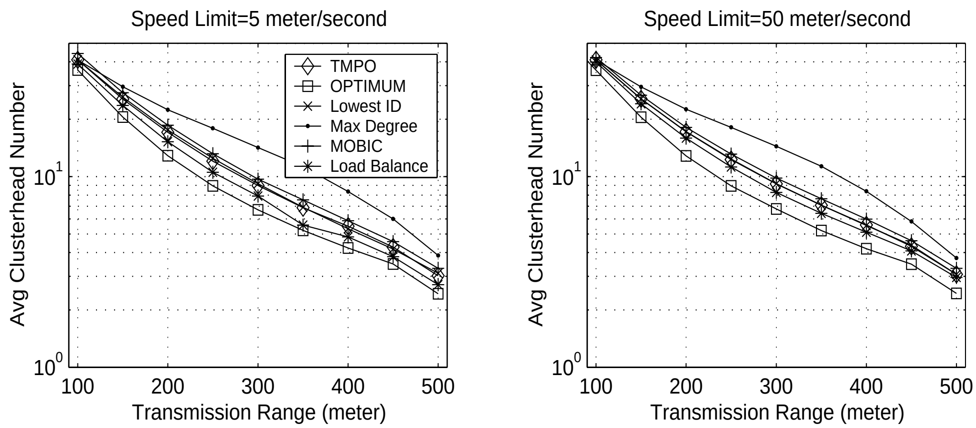


(a) Simulation duration.

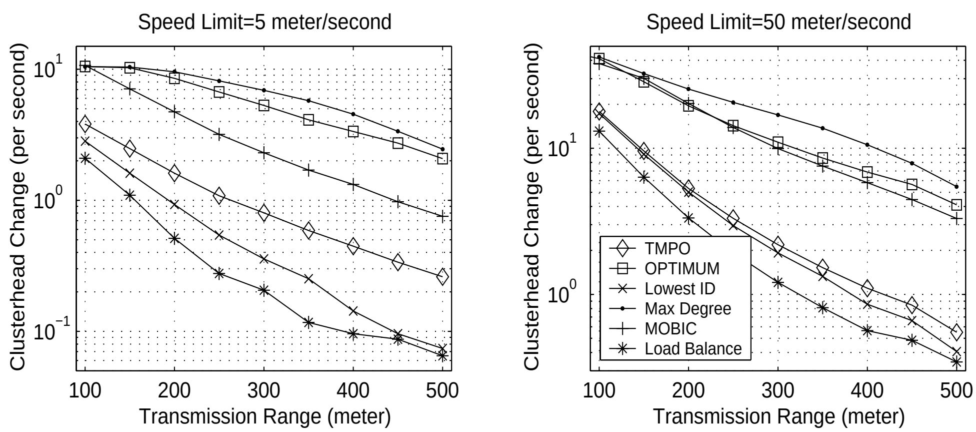


(b) The mean energy left and its standard deviation.

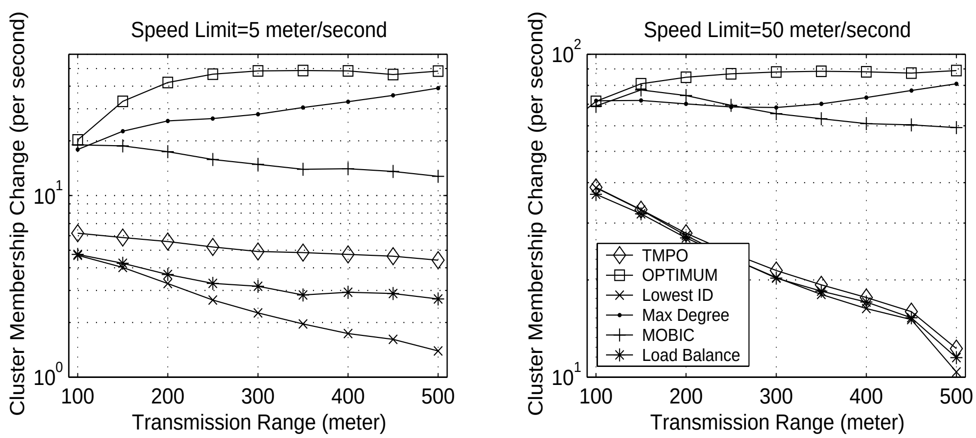
Figure 6.14: Simulation duration and energy left.



(a) Average number of clusterheads.



(b) Clusterhead change rate.



(c) Cluster membership change rate.

Figure 6.15: Statistics about clusterheads.

Clusterhead change rate (Figure 6.15 (b)): This metric measures the stability of the MDS in a mobile network. **TMPO**, **Lowest ID** and **Load Balance** perform the best because they depends on relatively static attributes for clusterhead election, such as node identifiers and priorities which change less frequently than node locations. **OPTIMUM**, **Max Degree** and **MOBIC** perform the worst because the MDS elections depend on the relative mobility and network topology.

Cluster membership change rate (Figure 6.15 (c)): This metric measures the stability of network connections in the presence of mobility. The attachments to the clusterheads vary accordingly to the clusterhead change rate corresponding to the heuristics. **TMPO**, **Lowest ID** and **Load Balance** still perform the best in both high and low mobility scenarios.

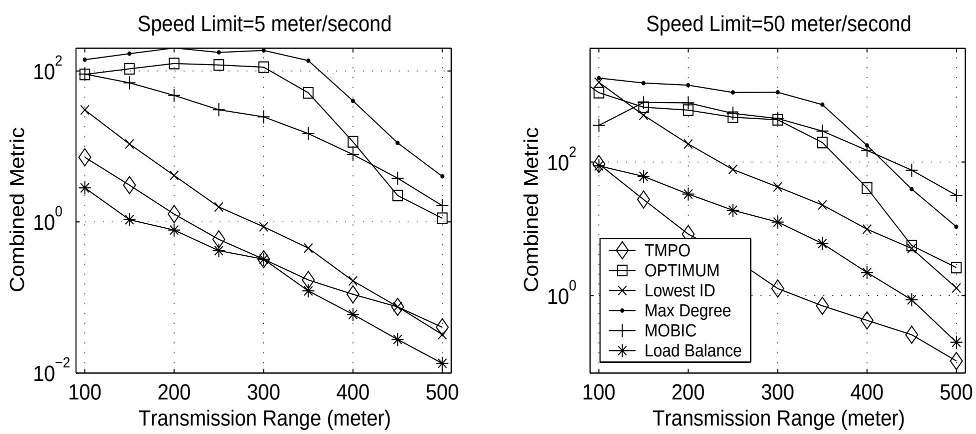


Figure 6.16: Combined evaluations.

Combined metric (Figure 6.16): Given the aforementioned metrics, it is not easy to see the advantages of different heuristics. The combined metric is the product of the average energy, its standard deviation, the average number of clusterheads, the clusterhead change rate and the cluster membership change rate of the respective simulations. It measures the overall performance derived with the metrics given above. The lower the combined metric of a heuristic, the better the heuristic performs in terms of the clusterhead load-balancing capability and the MDS/CDS stability. As shown in Figure 6.16, **TMPO** performs near, if not always, the best among all heuristics. **Load balance** is the second best in general. Note that **Load balance** assumes that all nodes start with the same energy level [4].

Overall, when the mobility increases from 5 meter/second to 50 meter/second, **TMPO** shows better load balancing capability (Figure 6.14) and higher topology maintenance stability (Figure 6.16).

6.7 Conclusion

We have presented **TMPO**, a novel energy-aware topology management algorithm that is based on dynamic node priorities in ad hoc networks. **TMPO** consists of two parts that implement the MDS and CDS elections, respectively. **TMPO** builds a stable and energy-aware CDS from the MDS to simplify the topology information for sufficient network connectivity and efficient data communications. Comparing with prior five heuristics of MDS and CDS elections in ad hoc networks, **TMPO** offers three key advantages. First, **TMPO** obtains the MDS and CDS of the network without any negotiation stage; only two-hop neighbor information is needed. Second, **TMPO** allows nodes in the network to periodically recompute their priorities, so as to balance the clusterhead role and prolong the battery life of each node. Third, **TMPO** introduces the willingness value of a node, which decides the probability of the node being elected into the MDS according to the battery life and mobility of the node.

A key contribution of the work consists of converting the static attributes of a node, such as node identifier, into a dynamic control mechanism that incorporates the three key factors for topology management in ad hoc networks — the nodal battery life, mobility, and load balancing.

Chapter 7

Contributions and Future Work

7.1 Contributions

The goal of this dissertation has been to explore efficient control mechanisms for communication in ad hoc networks. We presented the neighbor-aware contention resolution (NCR) algorithm. In contrast to other contention resolution mechanisms, NCR resolves contention among a group of contenders without frequently exchanging control messages or depending on a node knowing the complete topology of the network. Instead, the only required information for the operation of NCR is the set of contenders to the shared resource, and the following resource-access schedule is automatically derived from the contender information in each contention context (*e.g.* a time slot). The key idea behind NCR is to assign a dynamic priority number to each entity, *i.e.* the message digest of the entity's identifier and the contention context, and subsequently to determine the resource-access based on the priorities. In essence, NCR creates a random permutation of the set of contenders, and the precedence of an entity in the sequence determines the resource-access right of the entity among its contenders.

Several methods were proposed to dynamically allocate the shared resource according to the demand of an entity. Because the entity priorities are pseudo-random numbers, the distribution of the priorities can be adjusted according to the demand of each contender. If an entity requires more resource, the range of its priority is increased by the requested demand, thus giving the entity higher probability to win the contention. The increments for dynamic allocation of the resource can be linear or non-linear, resulting in several allocation methods with different levels of computational complexities.

Assuming that the work-load arrival events follow a Poisson distribution, we derived the delay and throughput attributes of NCR when utilizing NCR as the resource access control function. NCR provides steady throughput over the shared resource even when the work-load surpasses the capacity of the resource. In contrast, the degradation of throughput in face of high work-load is a common phenomenon in contention resolution mechanisms based on hand-shaking signaling.

In ad hoc networks, contention to the shared channel is limited to the two-hop neighborhood of each node. To acquire the neighborhood information, we proposed a new neighbor protocol that is based on slotted ALOHA and uses retransmission scheme to improve message-delivery reliability. We derived the number of, and the interval between retransmissions for safely broadcasting a piece of information to the one-hop neighbors of a node with high probability. The neighbor protocols eliminates the dependency on carrier-sensing or collision-avoidance handshakes, used by other random channel access protocols.

Based on the NCR algorithm, four channel access protocols were proposed for ad hoc networks with omnidirectional antennas, using different activation schemes. The node-activation multiple access (NAMA) protocol is suitable for collision-free broadcast in ad hoc networks with the simplest antenna technologies. In contrast, the following channel access protocols require more complicated physical layer technologies for code-division and frequency-division channelizations. The link-activation multiple access (LAMA) and pair-wise activation multiple access (PAMA) are suitable for scheduling unicast transmissions. The hybrid activation multiple access (HAMA) is capable of scheduling broadcast transmissions while maximizing the unicast transmission opportunities. We provided a probabilistic model for creating random ad hoc networks, and analyzed the respective average channel access probabilities of a node in the four channel access protocols. The analytical method is useful for comparing the protocol efficiency without carrying out simulations with concrete traffic patterns. We have shown that NAMA and HAMA can provide much higher throughput and better channel utilization than the ideal random access protocols, CSMA and collision-avoidance protocols.

We applied the NCR algorithm in the channel access scheduling in ad hoc networks with unidirectional links. A family of channel access protocols, PANAMA (the combination of pair-wise link activation and node activation multiple access schemes) is proposed, which provides collision-free channel access in ad hoc networks with unidirectional links. PANAMA is the first family of protocols that address efficient and collision-free communication over unidirectional links without the complete topology information.

We applied the NCR algorithm for channel access scheduling in ad hoc networks with directional antennas. A novel angular position profiling method was presented for tracking neighbor positions, and a new channel access protocol, the receiver-oriented multiple access (ROMA), was proposed for utilizing multi-beam adaptive arrays (MBAA) in ad hoc networks. ROMA offers four key advantages over prior approaches to the channel access problem. First, ROMA allows both transmitters and receivers to use the directional mode of the antennas, instead of requiring one end of the communication to stay in omnidirectional mode, as adopted by random access schemes. Second, ROMA relies on the local topology information within two hops for computing the channel access schedules, in contrast to the reliance on global topology information in the unified framework for channel access protocols (UxDMA). Third, ROMA evenly splits nodes in the network into transmitters and receivers in each time slot, which are then paired together for the maximum throughput. In contrast to UxDMA which allocates only one time slot per time frame to each link for activation, ROMA may activate a link multiple times during the same period.

We presented a new topology management algorithm for constructing backbone topologies in ad hoc networks, called TMPO (topology management by priority ordering), which is another application of the NCR algorithm. TMPO computes the priorities of nodes using the corresponding node identifiers and the "willingness" values for the nodes. The willingness value of a node is a function of the mobility and battery life of the node. Using the node priorities, TMPO elects the clusterheads that comprise the minimal dominating set (MDS) of the network, and consequently determines the nodes for connecting the clusterheads to obtain the connected dominating set (CDS) of the network. TMPO offers three key advantages over prior approaches in computation of the connected dominating set of a network. First, TMPO obtains the MDS and CDS of the network without any negotiation stage; only two-hop neighbor information is needed. Second, TMPO allows nodes in the network to periodically and asynchronously recompute their priorities, so as to balance the clusterhead role between nodes and prolong the battery life of each node. Third, TMPO introduces the willingness value of a node, which decides the probability of the node being elected into the MDS according to its battery life and mobility.

7.2 Future Work

NCR is a fundamental technique that can be utilized in different ways for various problems. To utilize NCR in control protocols, it is important that the problems are modeled into time-variant contention situations. Based on this modeling, we have explored the channel access scheduling problem according to the flow information in the networks. A flow involves a source and a destination that communicate over a multi-hop path. If we represent each flow as a contender to the channel, the NCR algorithm can easily handle the channel access scheduling problem according to the flow information, therefore supporting per-flow allocation of the channel resource. Depending on the physical technologies, such as omnidirectional or directional antennas, different algorithms may be derived.

The neighbor protocol for propagating neighbor update information can be used as a channel access scheduling method as well. The independence on destination address and reception acknowledgment during the message transmissions makes the channel access scheme ideal for wireless sensor networks. Wireless sensor networks have tight constraints in computation, storage and energy resources. Channel access in sensor networks must be energy efficient, and should allow fair bandwidth allocation to all nodes in the multihop network scenario. In contrast, the traditional approaches, such as CSMA and collision-avoidance protocols, may incur control overhead and congest the network when a large amount of data are being transferred through the sensor networks. The throughput and delay attributes of the various approaches are worth evaluating.

In addition, we expect to improve our research in the following areas:

- The effects of network mobility on the performance of the proposed channel access protocols and the neighbor protocol need to be studied. The simulations with regard to this aspect have been mostly limited to static network topologies for comparison with other true channel access scheduling algorithms. The comparisons with other channel access schemes, such as CSMA and collision-avoidance, need to be carried out in more realistic scenarios.

- The transmission ranges of directional antennas in directional mode and omnidirectional mode have been assumed the same. However, it may not be true in real implementations. The effects of having different transmission ranges for different transmission modes need to be studied.

- The work on the topology management by priority ordering (TMPO) is far from complete. The connectedness and dominating feature of the overlay network derived from TMPO are very attractive features needed by the routing and medium access control functionalities. The integration of TMPO with routing control protocols or channel access protocols may result in very efficient control mechanisms in ad hoc networks.

Bibliography

[1] IEEE Std 802.11. Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. Technical report, IEEE, 1997.

[2] Y. Afek and E. Gafni. Distributed Algorithm for Unidirectional Networks. *SIAM J. Comput.*, 23(6):1152–1178, Dec. 1994.

[3] A. Amis, R. Prakash, T. Vuong, and D.T. Huynh. MaxMin D-Cluster Formation in Wireless Ad Hoc Networks. In *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, Mar. 1999.

[4] A.D. Amis and R. Prakash. Load-balancing clusters in wireless ad hoc networks. In *Proceedings 3rd IEEE Symposium on Application-Specific Systems and Software Engineering Technology*, pages 25–32, Los Alamitos, CA, Mar. 24-25 2000.

[5] I.D. Aron and S.K.S. Gupta. A witness-aided routing protocol for mobile ad-hoc networks with unidirectional links. In *Mobile Data Access. First International Conference, MDA'99*, pages 24–33, Hong Kong, China, Dec. 16-17 1999.

[6] D.J. Baker and A. Ephremides. The architectural organization of a mobile radio network via a distributed algorithm. *IEEE Transactions on Communications*, COM-29(11):1694–701, Nov. 1981.

[7] S. Banerjee and S. Khuller. A Clustering Scheme for Hierarchical Control in Multi-hop Wireless Networks. In *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, Anchorage, Alaska, Apr. 2001.

[8] L. Bao and J.J. Garcia-Luna-Aceves. Link-state routing in networks with unidirectional links. In *Proceedings Eight International Conference on Computer Communications and Networks*, pages 358–63, Boston, MA, USA, Oct. 11-13 1999.

[9] L. Bao and J.J. Garcia-Luna-Aceves. A New Collision-Free Medium Access Control Protocol. In *Proc. IEEE MILCOM 2000*, Los Angeles, California, Oct. 22-25 2000.

[10] L. Bao and J.J. Garcia-Luna-Aceves. Unidirectional Link-State Routing with Propagation Control. In *Proc. IEEE Mobile Multimedia Communications (MoMuC) 2000*, Tokyo, Japan, Nov. 2000.

[11] L. Bao and J.J. Garcia-Luna-Aceves. A New Approach to Channel Access Scheduling for Ad Hoc Networks. In *Proc. ACM Seventh Annual International Conference on Mobile Computing and networking*, Rome, Italy, Jul. 16-21 2001.

[12] L. Bao and J.J. Garcia-Luna-Aceves. Channel Access Scheduling in Ad Hoc Networks with Unidirectional Links. In *Proc. ACM DialM 2001 – Fifth International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, Rome, Italy, Jul. 21 2001.

[13] L. Bao and J.J. Garcia-Luna-Aceves. Distributed Dynamic Channel Access Scheduling for Ad Hoc Networks. *Journal of Parallel and Distributed Computing, Special Issue on Wireless and Mobile Ad Hoc Networking and Computing*, accepted for publication 2002.

[14] L. Bao and J.J. Garcia-Luna-Aceves. Distributed Transmission Scheduling Using Code-Division Channelization. In *Proc. IFIP-TC6 Networking 2002*, Pisa, Italy, May 19-24 2002.

[15] L. Bao and J.J. Garcia-Luna-Aceves. Hybrid Channel Access Scheduling in Ad Hoc Networks. In *Proc. IEEE Tenth International Conference on Network Protocols (ICNP)*, Paris, France, Nov. 12-15 2002.

[16] L. Bao and J.J. Garcia-Luna-Aceves. Receiver-Oriented Multiple Access in Ad Hoc Networks with Directional Antennas. *ACM/Kluwer Mobile Networks and Applications (MONET)*, Dec. 2002.

[17] L. Bao and J.J. Garcia-Luna-Aceves. Transmission Scheduling in Ad Hoc Networks with Directional Antennas. In *Proc. ACM Eighth Annual International Conference on Mobile Computing and networking*, Atlanta, Georgia, USA, Sep. 23-28 2002.

[18] P. Basu, N. Khan, and T. D.C. Little. A Mobility Based Metric for Clustering in Mobile Ad Hoc Networks. In *International Workshop on Wireless Networks and Mobile Computing (WNMC2001)*, Scottsdale, Arizona, Apr. 16-19 2001.

[19] D. Bertsekas and R. Gallager. *Data Networks, 2nd edition.* Prentice Hall, Englewood Cliffs, NJ, 1992.

[20] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris. Span: an Energy-Efficient Coordination Algorithm for Topology Maintenance in Ad Hoc Wireless Networks. In *Proc. 7th ACM MOBICOM*, Rome, Italy, Jul. 2001.

[21] C.C. Chiang, H.K. Wu, W. Liu, and M. Gerla. Routing in clustered multihop, mobile wireless networks with fading channel. In *IEEE Singapore International Conference on Networks SICON'97*, pages 197–211, Singapore, Apr. 14-17 1997.

[22] I. Chlamtac, A. Farago, and H. Zhang. Time-spread multiple-access (TSMA) protocols for multihop mobile radio networks. *IEEE/ACM Transactions on Networking*, 6(5):804–12, Dec. 1997.

[23] I. Cidon and M. Sidi. Distributed assignment algorithms for multihop packet radio networks. *IEEE Transactions on Computers*, 38(10):1353–61, Oct 1989.

[24] T. Clausen, P. Jacquet, A. Laouiti, P. Muhlethaler, a. Qayyum, and L. Viennot. Optimized Link State Routing Protocol. In *IEEE INMIC*, Pakistan, 2001.

[25] M. Cooper and M. Goldburg. Intelligent Antennas: Spatial Division Multiple Access. *Annual Review of Communications*, pages 999–1002, 1996.

[26] B.P. Crow, I. Widjaja, L.G. Kim, and P.T. Sakai. IEEE 802.11 Wireless Local Area Networks. *IEEE Communications Magazine*, 35(9):116–26, Sept 1997.

[27] W. Dabbous, E. Duros, and T. Ernst. Dynamic Routing in Networks with Unidirectional Links. In *Proceedings of the Second International Workshop on Satellite-based Information Services*, Budapest, Hungary, Oct. 1997.

[28] W. Dabbous, Y. Zhang, D. Oran, and R. Coltun. A Link Layer Tunneling Mechanism for Unidirectional Links, Apr. 2000. Internet-Draft, Network Working Group.

[29] E. Duros and W. Dabbous. Supporting Unidirectional Links in the Internet. In *Proceedings of WOSBIS '96*, Rye, New York, Oct. 1996.

[30] E. Duros and W.Dabbous. Supporting Unidirectional Links in the Internet. In *Proceedings of the First International Workshop on Satellite-based Information Services*, Rye, New York, U.S.A., Nov 96.

[31] A. Ephremides and T.V. Truong. Scheduling broadcasts in multihop radio networks. *IEEE Transactions on Communications*, 38(4):456–60, Apr. 1990.

[32] A. Ephremides, J. E. Wieselthier, and D. J. Baker. A design concept for reliable mobile radio networks with frequency hopping signaling. *Proc. of IEEE*, 75(1):56–73, Jan. 1987.

[33] T. Ernst and W. Dabbous. A Circuit-based Approach for Routing in Unidirectional Links Networks. Technical report, Inria, Institut National de Recherche en Informatique et en Automatique, Oct. 1997. ercim.inria.publications/RR-3292.

[34] S. Even, O. Goldreich, S. Moran, and P. Tong. On the NP-completeness of certain network testing problems. *Networks*, 14(1):1–24, Mar. 1984.

[35] W. Feller. *An introduction to probability theory and its applications*, volume 1. New York: John Wiley, 2nd edition, 1957.

[36] M.R. Garey and D.S. Johnson. *Computers and intractability. A guide to the theory of NP-completeness*. Freeman, Oxford, UK, 1979.

[37] M. Gerla, L. Kleinrock, and Y. Afek. A distributed routing algorithm for unidirectional networks. In *GLOBECOM '83. IEEE Global Telecommunications Conference*, volume 2(3), pages 654–8, San Diego, CA, USA, 28 Nov.-1 Dec. 1983.

[38] M. Gerla and J.T.C. Tsai. Multicluster, mobile, multimedia radio network. *Wireless Networks*, 1(3):255–65, 1995.

[39] S. Guha and S. Khuller. Approximation algorithms for connected dominating sets. *Algorithmica*, 20, (no.4):374–87, Apr. 1998. Springer-Verlag.

[40] L. Hu. Topology control for multihop packet radio networks. *IEEE Transactions on Communications*, 41(10):1474–81, Oct. 1993.

[41] L. Jia, R. Rajaraman, and T. Suel. An Efficient Distributed Algorithm for Constructing Small Dominating Sets. In *Twentieth ACM Symposium on Principles of Distributed Computing PODC'01*, Newport, Rhode Island, Aug. 26-29 2001.

[42] M. Joa-Ng and I.T. Lu. Spread spectrum medium access protocol with collision avoidance in mobile ad-hoc wireless network. In *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, pages 776–83, New York, NY, USA, Mar. 21-25 1999.

[43] D.B. Johnson. Routing in ad hoc networks of mobile hosts. In *Workshop on Mobile Computing Systems and Applications*, pages 158–63, Santa Cruz, CA, USA, Dec. 8-9 1994.

[44] J.H. Ju and V.O.K. Li. An optimal topology-transparent scheduling method in multihop packet radio networks. *IEEE/ACM Transactions on Networking*, 6(3):298–306, Jun. 1998.

[45] W. Kishaba, G. Vardakas, J.J. Garcia-Luna-Aceves, L. Bao, and Y. Kang. Ad Hoc Networking with Beam Forming Antennas. In *Proc. IEEE MILCOM 2001*, Washington, D.C., Oct. 28-31 2001.

[46] L. Kleinrock and F.A. Tobagi. Packet switching in radio channels. I. Carrier sense multiple-access modes and their throughput-delay characteristics. *IEEE Transactions on Communications*, COM-23(12):1400–16, Dec 1975.

[47] Y.B. Ko, V. Shankarkumar, and N.H. Vaidya. Medium access control protocols using directional antennas in ad hoc networks. In *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, volume 1(3), pages 13–21, Tel Aviv, Israel, Mar. 26-30 2000.

[48] P. Krishna, N. Vaidya, M. Chatterjee, and D. Pradhan. A cluster-based approach for routing in dynamic networks. *ACM SIGCOMM Computer Communication Review*, pages 49–65, Apr. 1997.

[49] R. Krishnan and J.P.G. Sterbenz. An Evaluation of the TSMA Protocol as a Control Channel Mechanism in MMWN. Technical report, BBN Technical Memorandum No. 1279, 2000.

[50] L. Lamport. Time, clocks, and the ordering of events in a distributed system. *Communications of the ACM*, 21(7):558–65, Jul. 1978.

[51] R.J. Larsen and M.L. Marx. *An introduction to probability and its applications.* Englewood Cliffs, N.J. : Prentice-Hall, 1985.

[52] F.C.M. Lau, G. Chen, H. Huang, and L. Xie. A distance-vector routing protocol for networks with unidirectional links. *Computer Communications*, 23(4):418–24, Feb. 15 2000.

[53] L. Li, V. Bahl, Y.M. Wang, and R. Wattenhofer. Distributed Topology Control for Power Efficient Operation in Multihop Wireless Ad Hoc Networks. In *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, Apr. 2001.

[54] C.R. Lin and M. Gerla. Adaptive Clustering for Mobile Wireless Networks. *IEEE Journal on Selected Areas in Communications*, 15(7):1265–75, Sep. 1997.

[55] T. Makansi. Trasmitter-Oriented Code Assignment for Multihop Radio Networks. *IEEE Transactions on Communications*, 35(12):1379–82, Dec. 1987.

[56] R. McCurley and F. B. Schneider. Derivation of a Distributed Algorithm for Finding Paths in Directed Networks. *Science of Computer Programming*, 6(1):1–9, 1986.

[57] S. Narayanaswamy, V. Kawadia, R. S. Sreenivas, and P. R. Kumar. Power Control in Ad-Hoc Networks: Theory, Architecture, Algorithm and Implementation of the COMPOW Protocol. In *Proceedings of the European Wireless Conference – Next Generation Wireless Networks: Technologies, Protocols, Services and Applications*, pages 156–162, Florence, Italy, Feb. 25-28 2002.

[58] A. Nasipuri, S. Ye, J. You, and R.E. Hiromoto. A MAC Protocol for Mobile Ad Hoc Networks Using Directional Antennas. In *Proceedings of IEEE Wireless*

Communications and Networking Conference (WCNC), Chicago, IL, Sep. 23-28 2000.

[59] S. Nesargi and R. Prakash. A tunneling approach to routing with unidirectional links in mobile ad-hoc networks. In *Proceedings Ninth International Conference on Computer Communications and Networks*, pages 522–7, Piscataway, NJ, USA, 2000.

[60] M. Nishida, H. Kusumoto, and J. Murai. Network architecture using network address translation mechanism for network with unidirectional links. *Transactions of the Institute of Electronics, Information and Communication Engineers B-II*, J81B-II(5):458–67, May 1998.

[61] M. Nishizawa, H. Hagino, T. Hara, M. Tsukamoto, and S. Nishio. A routing method considering uni-directional links in ad-hoc networks. *Transactions of the Information Processing Society of Japan*, 41(3):783–91, Mar. 2000.

[62] C.A. Pomalaza-Raez. A distributed routing algorithm for multihop packet radio networks with uni- and bi-directional links. *IEEE Transactions on Vehicular Technology*, 44(3):579–85, Aug. 1995.

[63] R. Prakash. Unidirectional links prove costly in Wireless Ad-Hoc Networks. In *Proceedings of the Discrete Algorithms and Methods for Mobile Computing and Communications - DialM*, Seattle, WA, Aug. 20 1999.

[64] S.L. Preston, D.V. Thiel, T.A. Smith, S.G. O'Keefe, and Jun Wei Lu. Base-station tracking in mobile communications using a switched parasitic antenna array. *IEEE Transactions on Antennas and Propagation*, 46(6):841–4, Jun. 1998.

[65] R. Ramanathan. A unified framework and algorithm for channel assignment in wireless networks. *Wireless Networks*, 5(2):81–94, 1999.

[66] R. Ramanathan. On the Performance of Ad Hoc Networks with Beamforming Antennas. In *Proc. ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC)*, Long Beach, California, USA, Oct. 4-5 2001.

[67] R. Ramanathan and R. Rosales-Hain. Topology Control of Multihop Wireless Networks using Transmit Power Adjustment. In *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, page N.A. IEEE, Mar. 26-30 2000.

[68] R. Ramanathan and M. Steenstrup. Hierarchically-organized, multihop mobile wireless networks for quality-of-service support. *Mobile Networks and Applications*, 3(1):101–19, 1998.

[69] R. Ramaswami and K.K. Parhi. Distributed scheduling of broadcasts in a radio network. In *Proceedings of IEEE Conference on Computer Communications*

(INFOCOM), volume 2, pages 497–504, Ottawa, Ont., Canada, Apr. 23-27 1989. IEEE Comput. Soc. Press.

[70] J. Razavilar, F. Rashid-Farrokhi, and K.J.R. Liu. Software radio architecture with smart antennas: a tutorial on algorithms and complexity. *IEEE Journal on Selected Areas in Communications*, pages 662–76, Apr. 1999.

[71] P. Sinha, R. Sivakumar, and V. Bharghavan. Enhancing ad hoc routing with dynamic virtual infrastructures. In *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, pages 1763–72, Anchorage, AK, USA, Apr. 22-26 2001.

[72] R. Sivakumar, P. Sinha, and V. Bharghavan. CEDAR: a core-extraction distributed ad hoc routing algorithm. *IEEE Journal on Selected Areas in Communications*, 17(8):1454–65, Aug. 1999.

[73] H. Takagi and L. Kleinrock. Optimal transmission ranges for randomly distributed packet radio terminals. *IEEE Transactions on Communications*, 32(3):246–57, Mar. 1984.

[74] M. Takai, J. Martin, and R. Bagrodia. Directional Virtual Carrier Sensing for Directional Antennas in Mobile Ad Hoc Networks. In *Proc. ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC)*, Lausanne, Switzerland, Jun. 9-11 2002.

[75] V. Voipio and P. Vainikainen. Narrowbeam cylindrical antenna array with sparse antenna spacing. In *48th IEEE Vehicular Technology Conference*, volume 1, pages 465–9, Ottawa, Ont., Canada, May 18-21 1998.

[76] Y. Wang and J.J. Garcia-Luna-Aceves. Performance of Collision Avoidance Protocols in Single-Channel Ad Hoc Networks. In *Proc. of IEEE International Conference on Network Protocols (ICNP)*, Paris, France, Nov. 12-15 2002.

[77] J. Ward and Jr. Compton, R.T. High throughput slotted ALOHA packet radio networks with adaptive arrays. *IEEE Transactions on Communications*, 41(3):460–70, Mar. 1993.

[78] J.H. Winters. Smart antennas for wireless systems. *IEEE Personal Communications*, 5(1):23–7, Feb. 1998.

[79] J. Wu and H. Li. Domination and its applications in ad hoc wireless networks with unidirectional links. In *Proceedings 2000 International Conference on Parallel Processing*, pages 189–97, Los Alamitos, CA, USA, 2000.

[80] L. Wu and P. K. Varshney. Performance Analysis of CSMA and BTMA Protocols

in Multihop Networks (I), Single Channel Case. *Information Sciences*, 120(1-4):159–177, 1999.

[81] J. Zander. Slotted ALOHA multihop packet radio networks with directional antennas. *Electronics Letters*, 26(25):2098–100, Dec. 6 1990.

[82] C. Zhu and M.S. Corson. A five-phase reservation protocol (FPRP) for mobile ad hoc networks. In *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, volume 1, pages 322–31, San Francisco, CA, USA, Mar. 29-Apr. 2 1998.